#26 TECHARTIKEL

Architekturdokumentation mit dem arc42-Template – Ein Bootstrap Approach

26.01.2026

Christian Hummel

Inhalt

Architekturdokumentation mit dem arc42-Template – Ein Bootstrap Approach

Willkommen zum Blogbeitrag über Architekturdokumentation mit Hilfe des Dokumentation-Templates arc42. Dieser Beitrag verwendet die Gliederung dieses Patterns für Dokumentation von Software-Architektur. Daher der Untertitel: Ein Bootstrap Approach. Einführung und Ziele ist bereits der 1. Abschnitt des arc42-Templates. Im Rahmen des Blogs wird hier kurz die Notwendigkeit von Architekturdokumentation hergeleitet.

1. Einführung und Ziele

Wenn es zum Thema Dokumentation kommt, wird gerne argumentiert:

„Wir sind agil, da wird keine Dokumentation benötigt." oder

„Wir arbeiten nach dem ‚Clean Code' Prinzip, unser Code ist lesbar und letztlich die einzige Wahrheit."

Solche Einwände greifen aber zu kurz. Denn Code allein kann weder übergeordnete Architekturentscheidungen noch deren Begründungen transparent machen. Auch modulübergreifende Aspekte wie Security, Performance oder Integrationsszenarien bleiben darin unsichtbar.

Das Manifest für Agile Softwareentwicklung wird dabei gerne missverstanden. Es stellt nicht in Frage, dass Dokumentation notwendig ist, sondern betont lediglich, dass „funktionierende Software wichtiger ist als umfassende Dokumentation". Damit wird bereits eine wichtige Qualitätsanforderung an Architekturdokumentation deutlich: Sie muss angemessen sein – leichtgewichtig, praxisnah und nutzbar.

Architekturdokumentation ist kein Widerspruch zu agilen Arbeitsweisen, sondern eine notwendige Ergänzung. Sie unterstützt die mündliche Kommunikation, hält deren Ergebnisse fest und schafft eine gemeinsame, weiterentwickelbare Wissensbasis. Im Agilen Manifest ist dies unter dem Punkt ‚Zusammenarbeit mit dem Kunden mehr als Vertragsverhandlungen' zu finden. Für die Architekturdokumentation wird dadurch das Qualitätsmerkmal ‚Verständlichkeit' bedient.

Hinter dem Clean-Code-Gedanke steht auch, dass veraltete und dadurch falsche Dokumentation schlechter ist als keine Dokumentation. Aus diesem Grund muss die Architekturdokumentation korrekt und wartbar sein.

Vor diesem Hintergrund stellt sich die Frage: Wie gelingt gute Architekturdokumentation in der Praxis?

In diesem Beitrag wird das arc42-Template zur Dokumentation der Softwarearchitektur vorgestellt. Der Einstieg scheint zunächst abrupt, die Bedeutung der einzelnen Abschnitte wird genauer in der **5. Bausteinsicht** erklärt.

1.1. Aufgabenstellung

Software-Architekturdokumentation durch die Verwendung des arc42-Templates.

1.2. Qualitätsziele

nach iSAQB-Lehrplan [1] werden folgende Qualitätsziele an eine Architekturdokumentation gestellt.

- **Verständlichkeit** – Verständlichkeit kann nur von den Lesern des Dokuments beurteilt werden.
- **Angemessenheit** – Weniger ist mehr. Zu viel Dokumentation verschlechtert z.B. die Wartbarkeit.
- **Korrektheit** – Eine falsche Dokumentation ist schlechter als keine Dokumentation.
- **Effizienz** – Schnelle Erstellung und Wartbarkeit verringern den Ressourcenverbrauch.
- **Wartbarkeit** – Ist Voraussetzung für Korrektheit und Effizienz.

1.3. Stakeholder

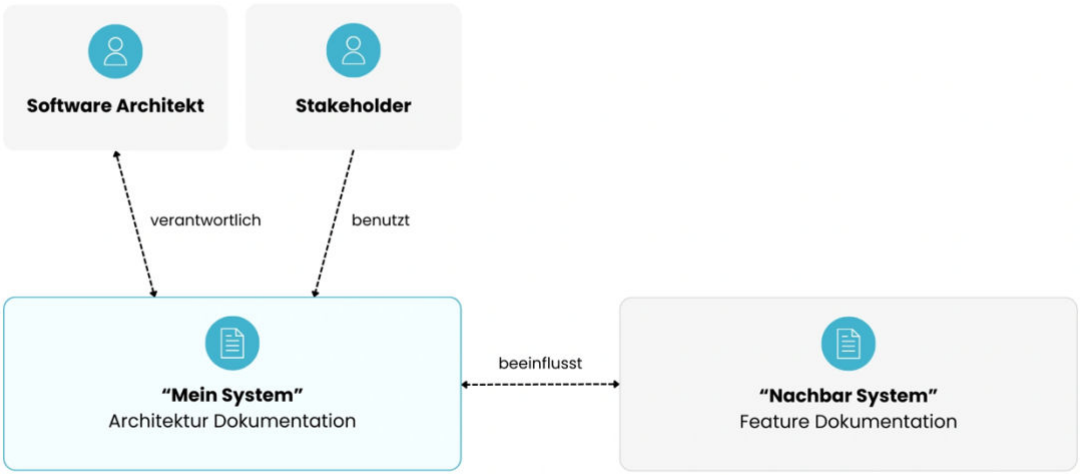
Die hier aufgeführten Rollen sind Stakeholder der Architekturdokumentation.

Rolle	Kontakt	Erwartungshaltung
Projekt Manager	Mike Money	Einhaltung des Zeitplans und des Budgets
Entwickler	Kai Board	Abgrenzung der Aufgaben, Definierte Schnittstellen
Fachbereich	Bea S. Ness	Umsetzbarkeit der Anordnungen
Infrastruktur	Susi Server	Verteilung der Komponenten, Notwendige Hardware

2. Randbedingungen

Notwendigkeit von Dokumentation wird nicht von allen Stakeholdern geteilt. Für die Dokumentation müssen Ressourcen bereitgestellt/eingeplant werden.

3. Kontext & Abgrenzung



Die Architekturdokumentation beschreibt vorrangig Qualitätsanforderungen und die Aufgaben, um diese zu erreichen. Sie beschreibt die Struktur des Systems weniger die Funktion des Systems, welche in der Feature-Dokumentation zu finden ist. Beide Dokumente sind nicht gänzlich unabhängig, da Entscheidungen in einem Bereich Auswirkungen auf den anderen Bereich haben können.

Der Softwarearchitekt ist für die Erstellung und Wartung der Architekturdokumentation verantwortlich.

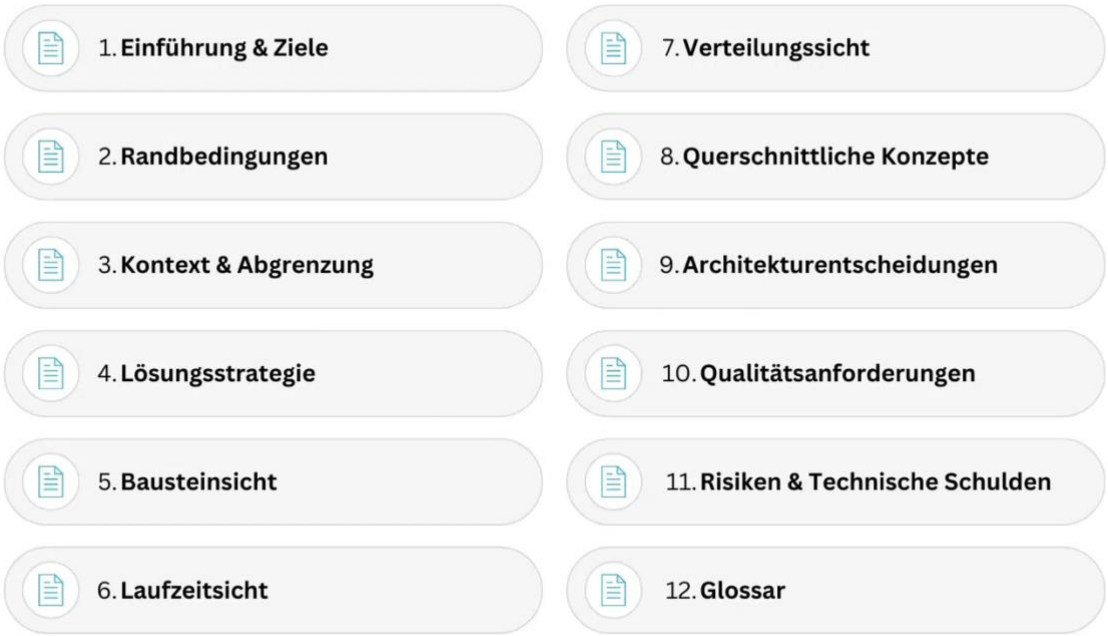
4. Lösungsstrategie

Verwendung des arc42-Templates, einer Mustervorgabe zur Architekturdokumentation. Ein Template bietet eine einheitliche Struktur und vereinfacht den Einstieg in die erste Iteration der Dokumentation. Es werden feste Abschnitte vorgegeben, die nicht alle ausgefüllt werden müssen. Leere Abschnitte verbleiben im Dokument und können später bei Bedarf ergänzt werden.

5. Bausteinsicht

Das arc42-Template verwendet folgende Bausteine, die Erklärung zu den Bausteinen wurde direkt oder sinngemäß aus [2] übernommen.

5.1. Whitebox Gesamtsystem



Blackbox 1. Einführung und Ziele

Die maßgeblichen Anforderungen, die Hauptaufgaben, der wesentliche Zweck des Systems. In diesem Kapitel fassen Sie zentrale Teile der Anforderungsdokumentation zusammen.

Blackbox 2. Randbedingungen

Organisatorische und technische Randbedingungen, die Auswirkungen auf Architekturentscheidungen besitzen können. Enthält die wichtigsten Einschränkungen der Entwurfsfreiheit. Insbesondere technische, organisatorische und rechtliche Randbedingungen sowie einzuhaltende Standards werden hier aufgeführt.

Blackbox 3. Kontext & Abgrenzung

Sicht aus der „Vogelperspektive“, zeigt das Gesamtsystem als Blackbox und den Zusammenhang mit Nachbarsystemen, wichtigen Stakeholdern sowie der technischen Infrastruktur. Es können hier sowohl eine rein fachliche wie auch eine sehr technische Perspektive eingenommen werden.

Blackbox 4. Lösungsstrategie

Die Kernidee der Lösung: Was sind die zentralen Lösungsansätze, Gestaltungskriterien oder Herangehensweisen? Dieses Kapitel sollte kompakt gehalten werden, die ausführliche Darstellung erfolgt in den zugehörigen Abschnitten.

Blackbox 5. Bausteinsicht

In der Bausteinsicht werden die einzelnen Teile des Systems und deren Beziehungen zueinander beschrieben. Die Abstraktionsebenen werden durch Blackboxes und Whiteboxes dargestellt. Je nach Abstraktionsgrad werden die Blackboxes genauer durch Whiteboxes erklärt. Die Flughöhe kann dabei in den Iterationen der Dokumentation stetig verringert werden.

Blackbox 6. Laufzeitsicht

Zeigt das Zusammenspiel der Architekturbausteine in Laufzeitszenarien. Hier sollten mindestens die Erfolgsszenarien der zentralen Use Cases sowie weitere wichtige Abläufe beschrieben werden.

Blackbox 7. Verteilungssicht

Diese Sichten zeigen, in welcher Umgebung das System abläuft, sowohl Hardware (Rechner, Netze etc.) als auch Software (Betriebssysteme, Datenbanken, Middleware etc.).

Blackbox 8. Querschnittliche Konzepte

Zu übergreifenden oder querschnittlichen Konzepten zählen beispielsweise Persistenz, Ausnahme- und Fehlerbehandlung, Logging und Protokollierung, Transaktions- und Session-Behandlung, der Aufbau der grafischen Oberfläche, Ergonomie sowie Integration oder Verteilung des Systems.

Blackbox 9. Architekturentscheidungen

Dokumentation der wichtigen übergreifenden Architekturentscheidungen und deren Gründe, die an keiner anderen Stelle von arc42 einen sinnvollen Platz finden. Architekturentscheidungen können z.B. durch Architecture Decision Records (ADR) dokumentiert werden. ADRs sind nicht Bestandteil des arc 42-Templates und werden hier auch nicht im Detail beschrieben.

Blackbox 10. Qualitätsanforderungen

Die einzelnen Qualitätsanforderungen stehen immer im Kontext zu einem Szenario, indem sie anwendbar und überprüfbar sind. Dieser Zusammenhang wird hier beschrieben.

Blackbox 11. Risiken & Technische Schulden

Technische Risiken mit ihren möglichen Auswirkungen und Abhilfemaßnahmen. Offenlegung der bekannten Defizite (technische Schulden) des Systems.

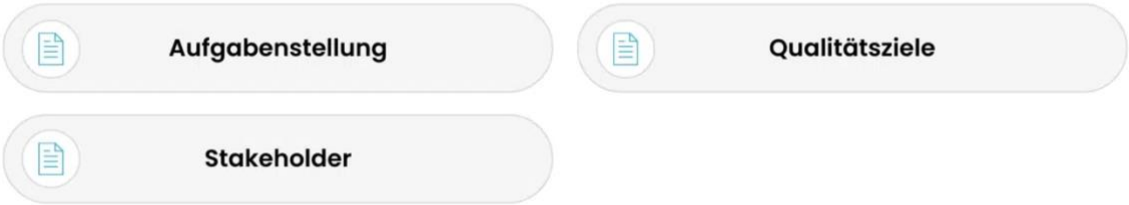
Blackbox 12. Glossar

Die wesentlichen (Fach-)Begriffe. Im Domain-Driven Design auch bekannt als Ubiquitous Language.

5.2. Ebene 2

Hier werden die Blackboxes aus dem Gesamtsystem als Whiteboxes dargestellt, falls eine weitere Aufteilung möglich ist.

Whitebox 1. Einführung und Ziele



Blackbox Aufgabenstellung

Eine kompakte Zusammenfassung des fachlichen Umfelds. Erklärt den „Grund“ für das System. Darstellung der wichtigsten vom System bearbeiteten

Geschäftsprozesse als Use Cases oder User Stories (Text und Diagramm). Verweise für Details auf die Anforderungsdokumentation.

Blackbox Qualitätsziele

Beschreibung der Qualitätsanforderungen, deren Erfüllung oder Einhaltung den maßgeblichen Stakeholdern besonders wichtig ist.

Architekturrelevante Themen wie Performance, Sicherheit, Änderbarkeit, Bedienbarkeit und Ähnliches werden hier angesprochen.

Hierzu gehören auch die Benennung und Quantifizierung der wichtigen Größen des Systems, etwa Datenaufkommen und Datenmengen, Dateigrößen, Anzahl User, Transaktionen, Geschäftsprozesse, Transfervolumen und andere.

Beschränkung auf die wichtigsten fünf bis zehn solcher Anforderungen ist empfohlen.

Stakeholder

Eine Liste der wichtigsten Personen oder Organisationen im Umfeld des Systems, zusammen mit deren wesentlichen Zielen und Erwartungen an das System inklusive dessen Konstruktion und Entwicklung

Whitebox 10. Qualitätsanforderungen

Blackbox Qualitätsbaum

Zuordnung der einzelnen Qualitätsanforderungen zu einzelnen Szenarien.

Blackbox Qualitätsszenarien

Auflistung der Qualitätsszenarien.

6. Laufzeitsicht

Dieser Abschnitt wird nicht benötigt, da keine Laufzeitsicht zur Verfügung steht. Das arc42-Template sieht jedoch vor, dass alle Abschnitte zumindest mit der Überschrift in der Dokumentation verbleiben, damit eine einheitliche und vergleichbare Dokumentation entsteht.

7. Verteilungssicht

Wie bei 6. bleibt dieser Teil leer.

8. Querschnittliche Konzepte

In diesem Blogbeitrag werden keine Konzepte beschrieben.

9. Architekturentscheidungen

ADR-SA-1: arc42 als SA-Dokumentation

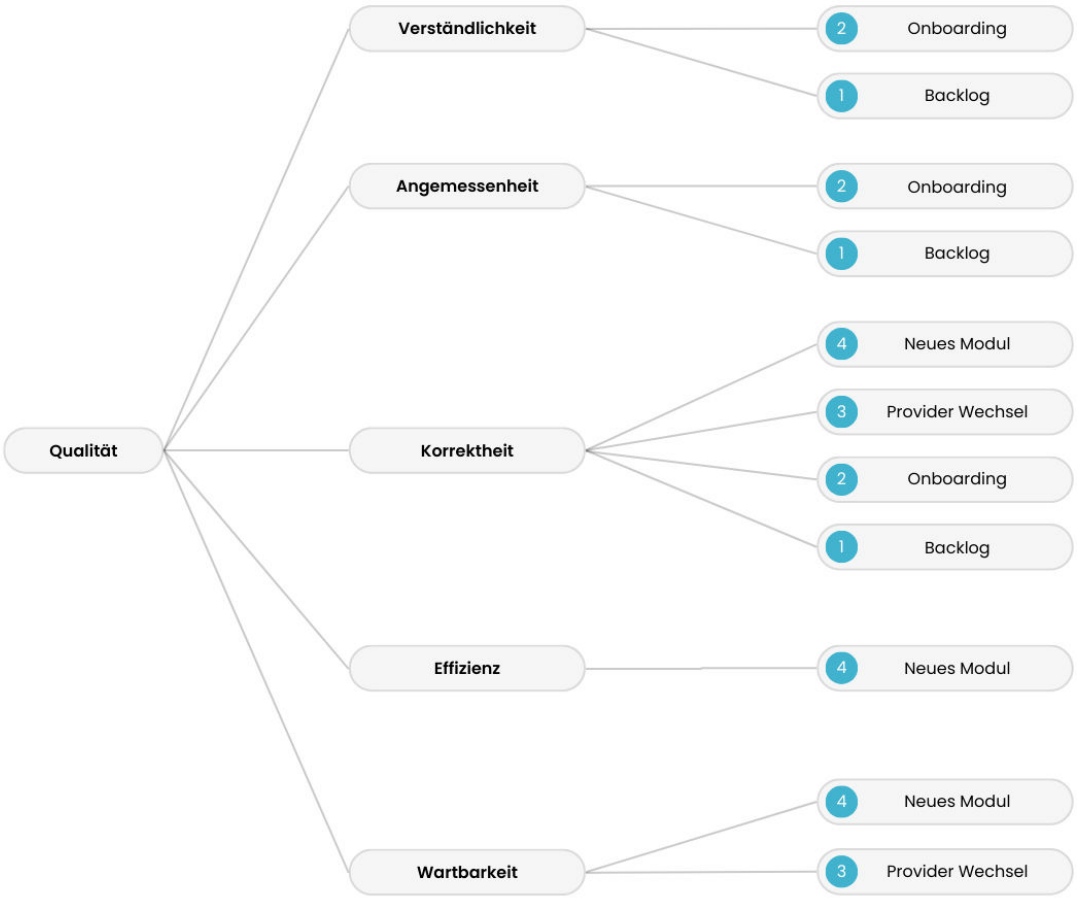
Kontext: Architekturdokumentation mit Hilfe einer Vorlage

Entscheidung: Das arc42-Template wird als Dokumentationsmuster verwendet, da es die Anforderung nach Verständlichkeit und Effizienz erfüllt und in Formaten für Word, Markup und Confluence zur Verfügung steht.

Konsequenzen: Die Dokumentation wird durch Verwendung des Templates einheitlicher und der Einstieg in die Dokumentation wird dadurch vereinfacht.

10. Qualitätsanforderungen

10.1. Qualitätsbaum



10.2. Qualitätsszenarien

1	Backlogerstellung durch PO – Qualitätsanforderungen müssen schnell ersichtlich sein.
2	Onboarding – Verständnis über das Gesamtsystem und die einzelnen Module.
3	Provider Wechsel- Betroffenen Module müssen erkannt werden und Dokumentation angepasst werden.
4	Neues Modul – Die neuen Anforderungen und der Kontext müssen geändert/erweitert werden.

11. Risiken und technische Schulden

Die Risiken einer Dokumentation liegen in Inkorrektheit und schlechter Wartbarkeit, beides wird durch eine zu umfangreiche Dokumentation begünstigt.

Technische Schulden sind fehlende oder fehlerhafte Dokumentation.

12. Fazit

Eine Dokumentation der Architektur eines Software-Systems ist notwendig. Das arc42-Template bietet eine Struktur und Beschreibungen, die den Einstieg in die Dokumentation erleichtert und es erlaubt, sich auf die Inhalte zu fokussieren. Das Template ist offen für Iteration, Refinement und Refactoring, also ein guter Begleiter im agilen Umfeld. Das Template war für mich ein guter Einstieg in die Erstellung dieses Blogbeitrags. Ich hoffe ich konnte euch das Template näherbringen und wünsche frohes Dokumentieren.

13.Glossar

Begriff	Definition
arc42	Template und Anleitung zur Software-Architekturdokumentation, https://arc42.org/
bootstrap	Analogie für ein selbststartendes oder selbstbeschreibendes System. https://de.wikipedia.org/wiki/Bootstrapping_(Informatik)
Word	Textverarbeitungssoftware
Confluence	Wiki-Like Dokumentationssystem von Atlassian.
ADR	Architecture Decision Record Template zur Beschreibung von Architektur Entscheidungen. https://adr.github.io/
iSAQB	Internationales Software Architecture Qualifikation Board, https://www.isaqb.org/
reg42	Templates für Requirements Management, https://req42.de/
UML	Unified Modeling Language, https://www.omg.org/uml/
PlantUml	Tool zum Erstellen von Diagrammen in Textform von https://plantuml.com/de/

14. Quellen

[1] iSAQB-Lehrplan zum „Certified Professional for Software Architecture – Foundation Level: https://isaqb-org.github.io/curriculum-foundation/

[2] Gernot Starke: Effektive Softwarearchitekturen - Ein praktischer Leitfaden 10., überarbeitete Auflage, Carl Hanser Verlag 2024.