



DAGMAR DE HAAN

SOFTWARE ARCHITEKTIN

SENIOR ENTWICKLERIN



mail@dagmardehaan.de



+49 (0) 15678 382588



<https://dagmardehaan.de>



Hofwiesenweg 2, 55130 Mainz



Geb. 08.07.1979

ÜBER MICH

Ausdauer, systematisches Vorgehen und der Ehrgeiz zur ständigen Verbesserung – das verbindet meine Leidenschaft für Sport mit meiner Expertise in der Softwareentwicklung. Als ehemalige 800m-Läuferin und aktive Gleitschirmfliegerin weiß ich, was es bedeutet, sich systematisch auf Herausforderungen vorzubereiten.

Diese Erfahrung nutze ich als Backend-Expertin für die Entwicklung anspruchsvoller Softwarelösungen. Nach meinem Informatikstudium in Bonn habe ich mich auf Softwarearchitektur spezialisiert – mit dem Fokus auf Codequalität, langfristig wartbarer Systeme und dem Abbau technischer Schulden.

Ich bin die richtige Ansprechpartnerin, wenn Sie eine Entwicklerin suchen, die nicht nur technisch versiert ist, sondern auch die Ausdauer und Zielstrebigkeit mitbringt, um komplexe Projekte erfolgreich umzusetzen. Dabei lerne ich kontinuierlich dazu und erschließe mir neue Technologien mit der gleichen Neugier, mit der ich meine sportlichen Grenzen erweitere.

AUSBILDUNG

Diplom-Informatik, Rheinische Friedrich-Wilhelms-Universität in Bonn, 2006

SPRACHKENNTNISSE

Deutsch (Muttersprache)

Englisch (Professionell)

ZERTIFIKATE

iSAQB®-Modul FLEX - Flexible Architekturmodelle, 2025

iSAQB® Certified Professional for Software Architecture - Foundation Level, 2024

AWS Certified Cloud Practitioner, 2021

AWS Certified Alexa Skill Builder, 2021

Certified Scrum Master (CSM), 2016

SKILLS

Methoden & Konzepte

Objektorientierte Programmierung ●●●

Domain-Driven Design ●●●

Test-Driven Development ●●●

Agile, Scrum, Kanban ●●●

Scrum of Scrum, SAFe ●●

Design Pattern ●●●

Clean Code ●●●

Onion/Hexagonale Architektur ●●●

Architekturdokumentation, Arc42 ●●●

IT-Sanierung ●●●

Refactorings ●●●

Microservices ●●●

Event-Driven Architecture ●●●

Event Sourcing ●●●

CQRS ●●

Pair/Mob Programming ●●●

Code Review ●●●

Datenbanken

Oracle ●●●

MySQL/MariaDB ●●

MongoDB ●●●

PostgreSQL ●●

DB2 ●●

SQL ●●

Hibernate ●●

Flyway ●●

JPA ●●

IDEs

IntelliJ IDEA ●●●

Eclipse ●●

Cloud

Azure ●●

AWS ●●

Backend

Java ●●●

Kotlin ●●●

Python ●●

Frontend

JavaScript ●●

TypeScript ●●

React ●●

Frameworks & Libraries

Spring Framework ●●●

Spring Boot ●●●

Spring Web, Cloud, Actuator ●●●

Spring Data, GraphQL ●●●

Java EE/Jakarta EE ●●

Kafka ●●●

Amazon Alexa ●●●

Lombok ●●●

Feign ●●

Log4J ●●●

OpenAPI/Swagger ●●●

Shell, Bash ●●

DevOps

Kubernetes ●●●

Docker ●●●

Instana ●●●

Grafana ●●●

Kibana ●●

Elasticsearch ●●

Prometheus ●●●

JMeter ●●

Micrometer ●●●

Nginx ●●

Tomcat ●●

Flux ●●

Terraform ●●

Keycloak ●●

Software Qualität

JUnit ●●●

Mockito ●●●

Sonar ●●●

Testcontainers ●●●

Wiremock ●●●

REST Assured ●●●

Cucumber ●●●

TestNG ●●

JMockit ●●●

Tools

Gradle ●●●

Maven ●●●

npm ●●

Git ●●●

GitLab ●●●

GitHub ●●●

Bitbucket ●●●

SVN ●●

Jira ●●●

Confluence ●●●

Postman ●●●

Formate

REST ●●●

JSON ●●●

XML/XSD ●●●

SOAP ●●●

WSDL ●●●

YAML ●●●

MQTT ●●

CI/CD

GitHub Actions ●●

Gitlab CI ●●

Hudson/Jenkins ●●●

Nexus ●●

Artifactory ●●

BISHERIGE PROJEKTE

Schott AG

(17.000 Mitarbeiter)
05/2024 – 05/2025

Qualitätstracking für Fertigungsprozesse

Kotlin, Spring Boot, Kubernetes, Docker, Microservices, REST, Azure, Helm, Instana, PostgreSQL, Scrum, JUnit, Swagger, MockK, Postman, Python, JavaScript/TypeScript, MQTT, Playwright, Bitbucket, Flyway, OPC UA, Micrometer, Grafana, Prometheus, Flux, Terraform, Keycloak, Flux, Arc42, Maven

Das Material Tracking System ermöglicht lückenlose Qualitätsverfolgung in der Fertigung. Alle Arbeitsschritte und Materialqualitäten werden nachvollziehbar dokumentiert.

Datenerfassung:

- Android-App für manuelle Qualitätsdatenerfassung durch Fabrikarbeiter
- OPC UA-Maschinenschnittstelle für automatisierte Datenübertragung

System-Architektur: Microservice-Backend als Vermittlungsschicht zwischen Datenquellen und Manufacturing Integration Plattform (MIP).

Performance-Anforderungen: Entwicklung für Massenproduktions-Szenarien mit strikten Last- und Performance-Vorgaben. Optimierung der Datenverarbeitung und -weiterleitung für kontinuierlichen hohen Durchsatz.

Vaillant Group

(16.000 Mitarbeiter)
08/2022 – 03/2024

Partner-API für Gerätesteuerung und -überwachung

Java, Spring Boot, Kubernetes, Kafka, Docker, Microservices, REST, Azure, IoT, github, Prometheus, Grafana, Flux, Terraform, Instana, MongoDB, Scrum, Scrum of Scrums, JUnit, Mockito, Lombok, Postman, Keycloak, Gradle

Das API Developer Programme stellt Vaillant-Partnern eine REST-API zur Heizgeräte-Steuerung und -Überwachung bereit. Die API ermöglicht u.a. die nahtlose Integration in Smart Home Systeme. Das System verarbeitet große Mengen an Datenpunkten der Heizgeräte, wobei eine präzise Zeitstempel-Verarbeitung für akkurate Gerätezustandsverfolgung kritisch ist.

Technische Umsetzung: Microservice-Architektur mit Apache Kafka für asynchrone Service-Kommunikation und reaktive Verarbeitung der High-Throughput-Datenströme vernetzter Geräte. DevOps-Ansatz mit End-to-End-Verantwortung für Development und Operations.

Technische Achievements:

- Migration auf aktuelle Java- und Spring Boot-Versionen
- Systematischer Abbau technischer Schulden
- Reaktive Datenverarbeitungs-Streams für skalierbare IoT-Verarbeitung
- Service-Redesign mit Context Mapping aus Domain-driven Design

Deutsche Bahn AG

(235.000 Mitarbeiter)
04/2021 – 05/2022

Mobile App-Integration für DB Fernverkehr-Systeme

Java, Kotlin, Spring Boot, Hibernate, Kubernetes, Docker, Microservices, REST, MQTT, AWS, Onion Architecture, gitlab, Prometheus, MariaDB, Scrum, Gradle, JUnit, Mockito, ArchUnit, Lombok

Entwicklung des Microservice-Backend für die RIS Communicator Android-App, eine zentrale Integrationsplattform für DB Fernverkehr-Mitarbeiter. Das System aggregiert Daten aus verschiedenen Legacy-Backends und stellt eine einheitliche API für das Frontend bereit.

Backend-Architektur: Microservice-basierte Datenintegration mit Abruf und Aggregation aus heterogenen Quellsystemen. Aufbereitung und Transformation der Daten für optimierte Frontend-Konsumierung.

Offline-Fähigkeit: Implementierung robuster Caching- und Synchronisationsmechanismen für unterbrechungsfreie Datenversorgung bei instabilen Netzwerkverbindungen. Gewährleistung der Datenkonsistenz auch bei temporären Verbindungsabbrüchen.

DevOps & Infrastructure: Vollständige Greenfield-Implementierung:

- CI/CD-Pipeline Setup und Automatisierung
- Kubernetes-basiertes Deployment und Container-Management
- Production-ready Monitoring und Alerting-System

Mercedes Benz AG
(175.000 Mitarbeiter)
10/2020 – 02/2021

Requirements Engineering für REST-API Integration

Kanban, REST, Swagger

Business-Analyse und Konzeption für die REST-API-Integration zwischen dem bestehenden System zur Presswerksplanung und einem externen Fremdsystem.

Aufgaben: Durchführung von Backlog Refinement Sessions mit Entwicklungsteam und Stakeholdern, Erstellung des technischen Integrationskonzepts und Überführung in entwicklungsreife User Stories mit Akzeptanzkriterien.

Miele & Cie. KG
(23.500 Mitarbeiter)
05/2019 – 12/2020

Alexa-Skill für Haushaltsgeräte

Java, Amazon Alexa, Custom Skill, SmartHome Skill, IoT, Kubernetes, Docker, Helm, Azure, AWS, gitlab, Kibana, Elasticsearch, Scrum, SAFe, JUnit, Mockito, Gradle

Entwicklung von Alexa-Skills für die sprachbasierte Steuerung von Miele-Haushaltsgeräten im Rahmen der Miele Cloud Services Plattform. Integration von Voice-Interfaces für Gerätefunktionen wie Statusabfrage, Programmsteuerung und Remote-Kontrolle.

Skill-Architektur: Entwicklung zweier spezialisierter Alexa-Skills:

- **Miele Custom Skill:** Vollständige Dialog-Engine mit eigenständiger Konversationslogik, Intent-Handling und Response-Management für komplexe Nutzerinteraktionen
- **Miele SmartHome Skill:** Integration in Amazons Smart Home API-Framework mit standardisierten Voice-Patterns für direkte Gerätesteuerung

Voice User Interface Design: Implementierung von Natural Language Processing für Sprach-Kommandos, Entwicklung der Konversationsflüsse und Response-Logik für intuitive Nutzerinteraktion.

Backend-Integration: Anbindung an die Miele Cloud Services Infrastruktur für Echtzeit-Gerätekommunikation und -steuerung. Entwicklung erfolgte im Multi-Team-Setup mit spezialisierten Entwicklungseinheiten.

Sonax GmbH
(< 500 Mitarbeiter)
07/2018 – 04/2019

IT-Architektur für Web-Plattform Modernisierung

Scrum, Kanban, CMS, PMS, Cloud

IT-Architektur-Verantwortung für den Aufbau einer neuen Webpräsenz mit integrierter Content- und Asset-Management-Landschaft.

Enterprise Architecture:

- Erfassung und Analyse der bestehenden IT-Landschaft
- Entwicklung der Zielarchitektur
- Ausarbeitung eines strukturierten Migrationsplans

System-Auswahl & Integration: Cloud-Provider-Evaluierung und Content Management System (CMS) Assessment. Architektur-Design für die Integration von Product Information Management (PIM) und Digital Asset Management (DAM) Systemen in die neue Web-Plattform.

Mercedes Benz AG
(175.000 Mitarbeiter)
12/2014 – 06/2018

Legacy-System-Transformation für Presswerksplanung

C#, WPF, DevExpress, JavaScript, TypeScript, REST, DB2, gitlab, Scrum

Modernisierung und Weiterentwicklung des KT-DB Systems (Karosserieteile-Datenbank) für Presswerksplanung und Karosserieteiledaten-Management. Das Legacy-System wies erhebliche Wartbarkeitsprobleme auf und erforderte umfassende Sanierungsmaßnahmen.

Legacy-Modernisierung: Neuentwicklung der Anwendung bei gleichzeitiger fachlicher Weiterentwicklung für Mehrfachteilefertigung. Kompletter Neuaufbau zur Wiederherstellung der Wartbarkeit und Reduzierung zukünftiger Entwicklungskosten.

Architektur-Migration: Vollständige Ablösung der FatClient-Anwendung durch moderne Web-Applikation mit N-Tier-Architektur. Migration von Desktop-basierter zu browser-basierter Benutzeroberfläche.

Technische Herausforderungen:

- Neuentwicklung mit nahtloser Migration von Legacy-Datenbeständen
- Integration neuer fachlicher Anforderungen in moderne Systemarchitektur
- Sicherstellung des Parallelbetriebs beider Systeme während der Migration

Deutsche Bahn AG
(235.000 Mitarbeiter)
06/2014 – 08/2014

IT-Landschaftsanalyse für Reisendeninformation

IT-Sanierung, Softwarelandkarten

Enterprise-Architektur-Analyse für die Modernisierung der Reisendeninformations-Systeme der Deutschen Bahn. Strategische IT-Bewertung zur Optimierung der geschäftsfeldübergreifenden Informationsverteilung bei Abweichungen vom Leistungsversprechen.

Potentialanalyse: Vollständige Erfassung und Analyse der aktuellen IT-Systemlandschaft für Reisendeninformation. Dokumentation der System-Aufgaben, Interaktionen und Abhängigkeiten zwischen den verschiedenen Informationskanälen.

Zukunftsfähigkeits-Assessment: Bewertung der bestehenden IT-Landschaft bezüglich Skalierbarkeit, Wartbarkeit und Erweiterbarkeit für zukünftige Anforderungen.

Strategische Handlungsempfehlungen: Entwicklung von Architektur-Empfehlungen und Roadmap zur Modernisierung der Systemlandschaft für konsistente, rechtzeitige Reisendeninformation über alle Kanäle.

TRAINERTÄTIGKEIT

2025

Workshop zu hexagonaler Architektur

bei KanDDDinsky 2025

2h-Workshop, der die praktischen Grundlagen sowohl für die Einführung, als auch für die Umstellung auf hexagonale Architektur vermittelt.

2021 - 2024

Workshop zu Event Sourcing und Aggregate Implementierung

bei DDD Europe 2021, OOP 2021, DDD Europe 2022, KanDDDinsky 2022, MAD Summit 2022 und 2023, OOP 2024

Ganztages-Workshop, in dem die Teilnehmer verschiedene Implementierung Muster für ein event-sourced Aggregate kennen und anwenden lernen. Zudem werden die Unterschiede zwischen den Konzepten CQRS, Event Sourcing und EDA ausgeführt.

2019 - 2022

Clean Code Schulung

Mehrwöchige Schulung, in der junioren Entwicklern die Clean Code Prinzipien nahe gebracht werden, u.a. KISS, YAGNI, DRY, Naming, Single Responsibility Principle, Open Closed Principle, Liskov Substitution Principle, Interface Segregation Principle, Dependency Injection, Testpyramide.

2018 - 2020

IT-Sanierungsschool

Zweitägige Schulung, bei der die Teilnehmer lernen, die technischen Schulden eines Bestandssystem (Legacy System) abzubauen. Dazu gehören die Absicherung der vorhandenen Funktionen über Tests, sowie verschiedene Sanierungstechniken.