Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

Vorlesung Objektorientierte
Komponenten-Architekturen
(OOKA)
SS 2023

Prof. Dr. Sascha Alda

26. Juni 2023

Übungen zur Vorlesung
Objektorientierte Komponenten-Architekturen

SS 2023

Übungsblatt Nr. 7

Informationen zum Semesterprojekt:

Für die anstehenden Prüfungen im September 2023 müssen sie ein Semesterprojekt entwickeln. Dabei können sie aus drei Fallstudien auswählen, die im Wesentlichen aus den Übungen hergeleitet und motiviert werden:

- Semesterprojekt „Laufzeitumgebung“: Dazu gelten die Anforderungen aus den Übungsblättern 2 und 3.

- Semesterprojekt „Microservices“: Dazu gelten die Anforderungen der Übungsblätter 4, 5 und 6-1 (also: Aufgabe 1 aus dem 6. Übungsblatt).

- Semesterprojekt „Migration Java EE Anwendung“: Dazu gelten die Anforderungen aus der Übungsaufgabe 6-2. Weitere Anforderungen zu der Fallstudie finden sie in dieser Übung (siehe Ende).

Präsentieren sie die Ergebnisse des Semesterprojekts im Rahmen eines Vortrags. Dieser Vortrag sollte folgendes umfassen:

- Präsentation der betreffenden fachlichen Anforderungen (Use Case etc.)
- Präsentation der betreffenden Teile der Software-Architektur, die analog zu dem 4-Sichtenmodell aufgebaut sein soll. Verwenden sie dazu die Modellierungssprache UML (z.B. Komponenten-, Verteilungs-, Klassendiagramm). Wesentliche Entwurfsentscheidungen sollten sie auf 1-2 Folien kurz kommentieren (vgl. das arc42-Template von Gernot Starke)
- Kurzer Code Walkthrough auf relevante Code-Passagen
- Kurze Demonstration der prototypischen Entwicklung
- Kurzes Fazit, „Lessons Learned“, Ausblick, aktuelle Restriktionen

Ihre gesamte Entwicklung *sollten* sie anhand des arc42-Templates dokumentieren, um auf dieser Basis einen Vortrag abzuleiten. Das arc42-Template dient somit als solide Grundlage für die Gestaltung des Vortrags und des Semesterprojekts! Das Template können sie auf ihre Bedürfnisse reduzieren (Hinweis: siehe auch die gut dokumentierten

Beispiele im Anhang bei (Starke, 2018)). Das arc42-Template *müssen* sie jedoch weder bei der Prüfung vorzeigen noch im LEA hochladen (siehe Anmerkung unten).

Bei einem Semesterprojekt mit einem Teilnehmer beträgt die Dauer des Vortrags 10-13 Minuten. Bei einer team-basierten Bearbeitung beträgt die Dauer 18-20 Minuten, wobei die Redeanteile klar zwischen den Studierenden aufgeteilt sein sollte (ca. jeweils 50% per Student/-in). Die Vorträge und die Prüfungen finden in Präsenz statt. Bitte laden sie das Handout (PDF der Folien) auf LEA bis spätestens eine Stunde vor der Prüfung hoch (Assignment „Übung Nr. 8 (Handout)“). Für die Zusammensetzung der finalen Note für OOKA verweise ich auf Kapitel 1 der Vorlesung.

Bitte teilen sie mir bis zum 10.9.2023, *22:00* per E-Mail mit, welches Thema sie bearbeiten und in welcher Konstellation (Vortrag alleine oder im Team). Ein Terminplan wird ca. 1 Woche vorher auf Basis der Prüfungsanmeldungen bekannt gegeben.

Für Rückfragen zu den Semesterprojekten können sie mich während meiner Sprechstunde (Do. 12-14 Uhr) regelmäßig (außer: 01.08. bis 25.8.23) kontaktieren.

Weitere Infos zum Semesterprojekt „Microservices“:

Für eine vollständige und solide Implementierung sollten sie *mindestens eine* von den vier Technologien bzw. technischen Anforderungen aus der Aufgabe 6-1 einsetzen:

- *MS_TA1:* Verwendung des Framework Apache Kafka für eine Integration der Microservices zur Übermittlung *insbesondere* der „Bearbeitungs-Status-Nachrichten“.

- *MS_TA2:* Verwendung der Technologien Docker und Docker Compose zur Bereitstellung und Komposition der Microservices in einsetzbare Docker Container.

- *MS_TA3:* Verwendung des Spring-Netflix-Stacks, um wichtige Pattern wie Service Registry oder Circuit Breaker *sinnvoll* zu integrieren (Alternative: Resilience4j)

- *MS_TA4:* Integration von KNative zur Realisierung einer skalierbaren Funktion

Optional können Sie zudem KNative Functions verwenden, um die Ergebnisse der Analysen abzuspeichern und ggf. wieder abzurufen. Falls Sie sich für diese Lösung entscheiden, finden unter dem nachfolgenden Link eine Anleitung für das Setup einer lokalen Knative Plattform:

https://git.inf.h-brs.de/lringh2m/knative-functions-dev-setup-guide

Die Analyse-Services sollen die Function über HTTP aufrufen. An dieser Stelle müssen Sie Apache Kafka nicht verwenden, da eine Anbindung via Knative Kafka Broker ziemlich komplex ist. Die Ergebnisse der Analysen sollten z.B. als JSON oder XML formatiert an die Speicher-Function übertragen werden. Diese soll die Daten in einer Datenbank (beispielsweise PostgreSQL) ablegen.

Weitere Infos zum Semesterprojekt „LZU“:

Neben den Anforderungen aus dem Übungsblatt Nr. 2 kommt eine Anforderung hinzu:

LZU_TA1:
Entwickeln sie ein UI-basiertes Frontend für ihre Laufzeitumgebung, mit der sie wesentliche Befehle ausführen können. Verwenden sie dazu ein Web-Framework ihrer Wahl (z.B. Vaadin, JSF, Angular) oder aber ein anderes UI-Framework (z.B. JavaFX).

Weitere Infos zum Semesterprojekt „Migration Java EE Anwendung“:

Die Firma YourConf KG plant die Migration der Anwendung „TheConfSystem“ hin zu einer Microservice-basierten Software-Architektur als mittelfristiges Ziel. Ihre Aufgaben:

- Wenden sie ihre Kenntnisse über Java EE (Kapitel 6, Übung Nr. 6) an und installieren sie die Anwendung lokal auf ihrem Rechner. Falls sie eine Datenbank (PostgreSQL) aus der H-BRS benötigen, so schreiben sie mir eine E-Mail.

- Modellieren sie die gegebene IST-Architektur anhand des 4-Sichten-Modells nach Starke. Auf eine Laufzeitsicht können sie verzichten.

- Führen sie eine *partielle* Migration der Anwendung unter der Verwendung ihrer Erkenntnisse aus dem Kapitel 6 durch.

- Fügen sie im Rahmen der partiellen Migration an einer frei wählbaren Stelle eine Sicherheitsschranke unter der Verwendung des Musters „Feature Toggle“ ein. Das Feature Toggle sollte auch in einer Demo vorgeführt werden.

- Dokumentieren sie die Migrationsschritte auf Architektur-Ebene. Modellieren sie die Phasen der Migration auf Basis einer geeigneten Baustein-Sicht sowie unter der Verwendung einer geeigneten Markierungs-Methode, mit der sie ihre Anpassungen oder Erweiterungen kennzeichnen (vgl. Kapitel 3, Abschnitt 2).

- Empfehlung für eine Migration: Migrieren sie zunächst die EJB-Komponenten (siehe Package `service.user`) auf unabhängige Microservices, zu denen es zum Teil schon eine öffentliche API z.B. in Form eines REST-Service gibt (vgl. Package `network.rest`). Betrachten sie auch die abhängigen EJB bzw. Services, die im Rahmen *einer* Transaktion aufgerufen werden.

- Entwickeln sie einen ersten Prototyp der resultierenden Software-Architektur auf Grundlage der Technologien Spring Boot und (ggf.) Docker.

- Modellieren sie die resultierende ZIEL-Architektur nach der partiellen Migration anhand des 4-Sichten-Modells nach Starke (ohne Laufzeitsicht).