Übung Nr. 1

Jan Kessel und Sophia Nowicki

Antworten zu Fragen in der Aufgabenstellung:

Im Kontext eines Komponentendiagramms dienen Ports dafür, eine Klasse von der Umgebung abzukapseln und einen Zugangspunkt zwischen internen Strukturen und der Umgebung herzustellen. Anstatt mit der eigentlichen Klasse zu kommunizieren, werden die Operationen des Ports benutzt. Die Ports bzw. der Port können mit Schnittstellen versehen werden, die Operationen des Ports definieren. Der Port kann dabei Querschnittsfunktionalitäten wie Caching oder Protokollüberwachung realisieren (vgl. Rupp, C., & Queins, S. (2012). UML 2 glasklar: Praxiswissen für die UML-Modellierung. Carl Hanser Verlag GmbH Co KG.)

Interfaces könnten zentralisiert über ein Facade-Pattern injiziert bzw. entnommen werden. Des Weiteren könnte über Frameworks wie SpringBoot o. ä. eine Dependency-Injection vorgenommen werden.

FA0: Für eine Delegation zwischen internen und externen Verhalten eignet sich das Proxy-Pattern, wo ein Proxy ein mit dem zu abgekapselten Subjekt gemeinsames öffentliches Interface implementiert. Der Port tritt als Kommunikationspartner im Auftrag der abgekapselten Komponenten für die Außenwelt dar.

FA1: Eine dedizierte Reihenfolge des Aufrufs der Methoden des HotelSearch-Interface ist durch eine erforderlichen Verbindungsaufbau zur Datenbank nötig. Eine entsprechende Logik wird in der Klasse HotelSearchPort implementiert.

FA2: Die Schnittstelle Caching sollte zudem eine Operation zum Abruf gecachter Objekte wie get(key: String, value: List) sowie eine Operation zum Leeren des Cache-Speichers exponieren.

Abgabe des Source Codes: Siehe .jar-File im "target"-Ordner des GitHub-Repositories unter https://github.com/KsslJan/OOKA.