Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

Vorlesung Software Engineering I
WS 2021 / 2022

Prof. Dr. Sascha Alda
Robert Hartmann (Dipl.-Inform.)

15. Dezember 2021

Übungen zur Vorlesung Software Engineering I
WS 2021 / 2022

Übungsblatt Nr. 9

(Abgabe bis: Mittwoch, den 22. Dezember 2021, 09:00 Uhr (*na kommt: auf zur letzten Meile in diesem Jahr! Im Anschluss ist dann (quasi) Weihnachten* ;-)).

Aufgabe 1 (Verifikationen von Rechnungen, 15 Punkte)

Lösung: Strategy Pattern

Der Betreiber des Legacy-Systems Reise-Anbieter aus der Aufgabe 9-2 bittet um ihre Mithilfe! Das System (repräsentiert durch eine Klasse `ReiseAnbieterController`, *= Client* welche das Interface `ReiseAnbieter` implementiert) soll über eine Möglichkeit zur Verifikation von Buchungen gemäß internationaler Reporting-Standards erweitert werden. Ihr Auftraggeber sieht dabei aktuell drei verschiedene Verifikationen vor, die jeweils als eigenständige Algorithmen im System bereitgestellt werden sollen und von einer Klasse *Policy =* `GlobalConfig` flexibel *gesetzt* werden kann:

- Swiss GAAP FER
- IFRS *= die konkreten Strategien (= Algorithmen)*
- US-GAAP

Da das Unternehmen in Zukunft plant, in weitere Länder zu expandieren, möchte man zudem offen sein für die Integration weiterer Algorithmen zur Anbindung weiterer länderspezifischer Standards. Alle Algorithmen verfügen über ein gemeinsames Interface, wobei das Interface über eine Methode verfügt: *= die abstrakte Strategien (mit der Methode verifyBooking))*

```
verfiyBooking ( b : Buchung ) : Status
```
= ContextInterface (= die eine Methode der Context-Klasse

Die Klasse `ReiseAnbieterController` soll von den Algorithmen entkoppelt sein, d.h. die Klasse darf die Klassen der Algorithmen nicht kennen. *= ein Indiz für die Context-Klasse*

Ihre Aufgabe:

a.)

Welches Entwurfsmuster (*Singular*!) würden sie für dieses Entwurfsproblem verwenden? Modellieren sie mit Hilfe des identifizierten Entwurfsmusters das Klassendiagramm entsprechend. Modellieren sie die Signaturen der Methoden ihrer identifizierten Klassen mit Hilfe der Notationsmöglichkeiten durch die UML. Modellieren sie auch die wichtigsten Attribute. Die Klassen `Buchung` und `Status` brauchen sie nicht zu berücksichtigen. Die Abhängigkeiten ausgehend von der Klasse `GlobalConfig` sollten sie explizit berücksichtigen. *Hinweis: zu den konkreten Strategien!!*

b.)
Modellieren sie ein konkretes Szenario mit Hilfe eines UML-basierten Sequenzdiagram. Verwenden sie dazu *asynchrone* Interaktionen. Gehen sie davon aus, dass die Policy zu Beginn des Szenarios den Algorithmus „IFRS“ erzeugt und in den Context einsetzt. Darauf kann der Client (hier: `ReiseAnbieterController`) die weiteren Interaktionen initiieren. Bei Callbacks können sie eine Callback-Funktion `return` annehmen, welche z.B. den Status übergibt (Beispiel: `return ( s : Status )`). Die Objekte `Buchung` und `Status` brauchen sie nicht explizit erzeugen, können sie aber als Argumente auf den Nachrichten annehmen.

Aufgabe 2 (Benachrichtigung von Benutzereingaben, 15 Punkte)

In einem Web-Framework gibt es eine Reihe von UI-Elementen (Components), welche in einer graphischen Benutzeroberfläche eingebunden werden können:

- Button
- Choice *= Concrete Publisher*
- Table

Component = Publisher

All diese Klassen sind von der Klasse `Component` abgeleitet. Jedes UI-Element ist von einem Benutzer „klickbar“, wodurch ein Ereignis ausgelöst werden kann. Über dieses Ereignis sollen Objekte aus der Anwendungslogik („AL-Objekte“) zur Laufzeit benachrichtigt werden können. Die Benachrichtigung soll *flexibel* durchgeführt werden: Nur „AL-Objekte“, die aktuell bei den UI-Elementen *eingetragen* sind, sollen benachrichtigt werden. „AL-Objekte“ müssen für eine Benachrichtigung folgendes Interface `ActionListener` implementieren, das folgende Methode enthält:

Observer Pattern

= Subscriber

```
update( c : ClickEvent )
```
= Methode des Subscribers

Die Informationen über das vom Benutzer ausgelöste Ereignis werden in einem Objekt vom Typ `ClickEvent` beschrieben; eine Abholung von weiteren Informationen bzw. Daten bei Bedarf durch die „AL-Objekte“ ist nicht notwendig bzw. nicht vorgesehen.

Factory Method Pattern

Die Erzeugung von einem Objekt vom Typ `ClickEvent` soll konsistent über eine separate Klasse erfolgen. Die Erzeugung von dem Click-Event soll abhängig von dem Class-Typ des UI-Elements erfolgen. Folgende Methode soll dabei in der separaten Klasse vorgesehen werden: *Klasse „Factory“ wird verwendet durch die Klasse Component*

```
createClickEvent( class : Class ) : ClickEvent
```

Ihre Aufgabe:
Welche Entwurfsmuster (Plural!) sind bei der Lösung dieser Problemstellung notwendig? Modellieren sie ein UML-basierten Klassendiagramm zur Lösung dieser Problemstellung unter der Berücksichtigung der identifizierten Muster. Modellieren sie alle Methoden und Attribute exakt. Als Beispiel eines „AL-Objekts“ können sie die Klasse „RegistrationControl“ annehmen. Modellieren sie auch die Klasse `ClickEvent`.

= ConcreteSubscriber