PES
Technische Universität Berlin
Software Engineering for Embedded Systems Group – Prof. Dr. Sabine Glesner
www.pes.tu-berlin.de Secr. TEL 12-4 Ernst-Reuter-Platz 7 10587 Berlin

Softwaretechnik und Programmierparadigmen WiSe 2014/2015

Prof. Dr. Sabine Glesner
Joachim Fellmuth joachim.fellmuth@tu-berlin.de
Dr. Thomas Göthel thomas.goethel@tu-berlin.de
Lydia Mattick lydia.mattick@tu-berlin.de
Tutoren

Übungsblatt 12 **Ausgabe: 15.01. (Besprechung: 19.01. und 20.01.)**

1. Entwurfsmuster

Eure Firma arbeitet an einem Qualitätsanalysetool, welches Modelle aus einer XML Datei auslesen und im Speicher darstellen soll. Anfangs sollen Kripke-Modelle eingelesen werden, zukünftig soll es möglich sein, auch komplexere Modelle mit weiteren Gestaltungselementen zu parsen. Es ist bereits ein XML-Parser geschrieben worden, welcher die XML-Tags der Datei in eine interne XML-Tag-Klasse abbildet. Aus diesen Tags soll nun das komplette Modell abgebildet werden. Die XML-Tags sollen jeweils als eigene Klassen und die Attribute der Tags als Attribute dieser Klassen abgebildet werden. Beim Einlesen soll für jeden XML-Tag ein entsprechendes Objekt im Speicher erzeugt werden.
Die XML-Tags sollen zusätzlich in unterschiedlichen Sprachen unterstützt werden. Die Einführung einer neuen Sprache soll möglich sein, ohne die bestehenden Klassen zu ändern.

XML-Datei eines Kripke-Modells

```
<Zustand name="s0" initial="true">
    <Value val="q"></Value>
</Zustand>
<Zustand name="s1">
    <Value val="p"></Value>
    <Value val="r"></Value>
</Zustand>
<Zustand name="s2">
    <Value val="r"></Value>
```

```
</Zustand>
<Zustand name="s3">
    <Value val="p"></Value>
    <Value val="q"></Value>
</Zustand>
<Transition from="s0" to="s0"></Transition>
<Transition from="s0" to="s1"></Transition>
<Transition from="s1" to="s2"></Transition>
<Transition from="s1" to="s3"></Transition>
<Transition from="s2" to="s1"></Transition>
<Transition from="s3" to="s3"></Transition>

/**
 * An XmlTag object represents a parsed tag of an xml file
 * The original Tag, all given Attributes and the subTags
 * are saved and accessible with getters
 */
public class XmlTag {
    String tag;
    HashMap<String,String> attributes;
    ArrayList<XmlTag> subTags;

    /**
     * @return the tag of this xml object
     */
    public String getTag(){
        return this.tag;
    }

    /**
     * @return all xml tags, which are sub tags of this xml tag
     */
    public ArrayList<XmlTag> getSubTags(){
        return this.subTags;
    }

    /**
     * @return a HashMap with all attributes of this xml tag
     */
    public HashMap<String,String> getAttributes(){
        return this.attributes;
    }
}
```

Entwurfsmuster = Design Patterns
gute, elegante Beispiellösungen für (Design) Probleme

a) Welches der in der Vorlesung vorgestellten Entwurfsmuster eignet sich für die Umsetzung?
factory method

b) Warum ist es sinnvoll dieses Entwurfsmuster zu nutzen?
für verschiedene Sprachen sollen die gleichen Methoden verfügbar sein, wir wissen aber noch nicht konkret wie viele Sprachen es sein werden

c) Erarbeitet eine Umsetzung in Java.

d) Eure Firma hat ein spezielles Qualitätsmodell zur Modellanalyse erarbeitet. Das Qualitätsmodell ist das Herzstück des Programms und die darin definierten Metriken sollen auf jedenfall genutzt werden. Aus diesem Grund soll es nur möglich sein eine Instanz der vorhandenen Qualitätsmodell-Klasse zu erzeugen. Wie lässt sich das implementieren?
Singleton: globale, statische Variable für die einzige erlaubte Instanz des Qualitätsmodells

```
public class qualityModel {
    public static model;
    private qualityModel(){
        // Konstruktur private, nicht aufrufbar}
    public qualityModel getInstance() {
        if (model == null) model = new qualityModel()
        return model;
```

2. Metriken

Diskutiert die folgenden Punkte.

a) Was ist eine Metrik?
weist Programmen Kenngrößen zu, die Qualität & Komplexität abbilden sollen

b) Wozu dienen Metriken in der Softwareentwicklung?
objektives, verlässliches Vergleichkriterium; Code messbar machen

c) Was versteht man unter Zeilenmetrik?
Wie viele Zeilen Code? -> wenig aussagekräftig

d) Was sagt die Halstead Metrik?
lexikalische Komplexität (im Verhältnis zu verwendeten Sprache)

e) Was ist zyklomatische Komplexität?
zählt Verzweigungen im Programm

f) Was sollte man bei der Verwendung von Metriken beachten?
keine sichere Wissenschaft; nur Hilfestellung

3. Anwenden der Halstead Metrik

Führe die Halstead Metrik am Beispiel des ggt-Codes durch!

Vereinbarung:
Operatoren: Sprachelemente (Operatoren, Schlüsselwörter..), Funktionsnamen
Operanden: Bezeichner, Literale

```
int ggt(int x, int y){
    while(x != y){
        if(x > y){
            x -= y;
        }else{
            y -=x;
        }
    }
```

Operatoren: !=, >, -=, while, if, else, ggt, int, {}, (), ;, Komma, return
Anzahl: n1 = 12 Operatoren
Vorkommen: N1 = 20

Operanden: x, y
Anzahl: n2 = 2
Vorkommen N2 = 11

Vokabular n = n1 + n2 = 14
verwendete Elemente N = N1 + N2 = 31

Schwierigkeit D = n1 / 2 * N2 / n2
-> komplexer je mehr Operatoren oder je weniger verschiedene Operanden

Volumen V = log(n) * N

```
return x;
}
```

4. McCabes zyklomatische Komplexität

Berechnet für die folgenden Beispielprogramme McCabes zyklomatische Komplexität `v(G)`. Der Kontrollflussgraph zum jeweiligen Beispielprogramm ist gegeben. Für die Berechnung gelten die folgenden Formeln:

```
v(G) = e-n+2*p
e: Anzahl Kanten im Graphen
n: Anzahl Knoten im Graphen
p: Anzahl Komponenten
```

```
v(G) = b+p
b: Anzahl Binärverzweigungen im
Graphen
p: Anzahl der Graphen
```

3 Verzweigungen (2 Schleifen und 1 if)

a)

```
public static int[] bubblesort(int[] a) {
    int temp;

    for(int i=1; i<a.length; i++) {
        for(int j=0; j<a.length-i; j++) {
            if(a[j]>a[j+1]) {
                temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
            }
        }
    }
    return a;
}
```

Berechnungsmethoden:

v(G) = Anzahl binärer Verzweigungen +1
= b + 1

ODER

v(G) = Anzahl Knoten - Anzahl Kanten +
2 * Anzahl Komponenten
= c - n + 2p

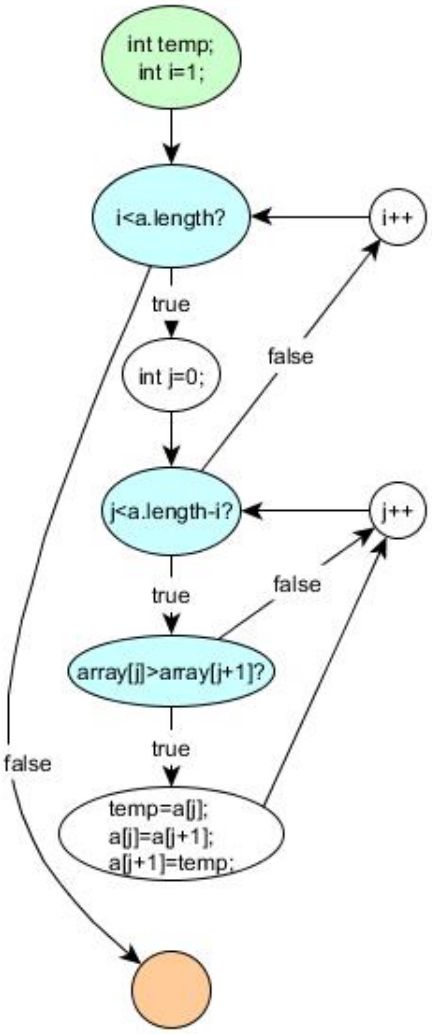
Am Beispiel:

v(G) = b + 1 = 4

ODER

v(G) = c - n + 2p = 4

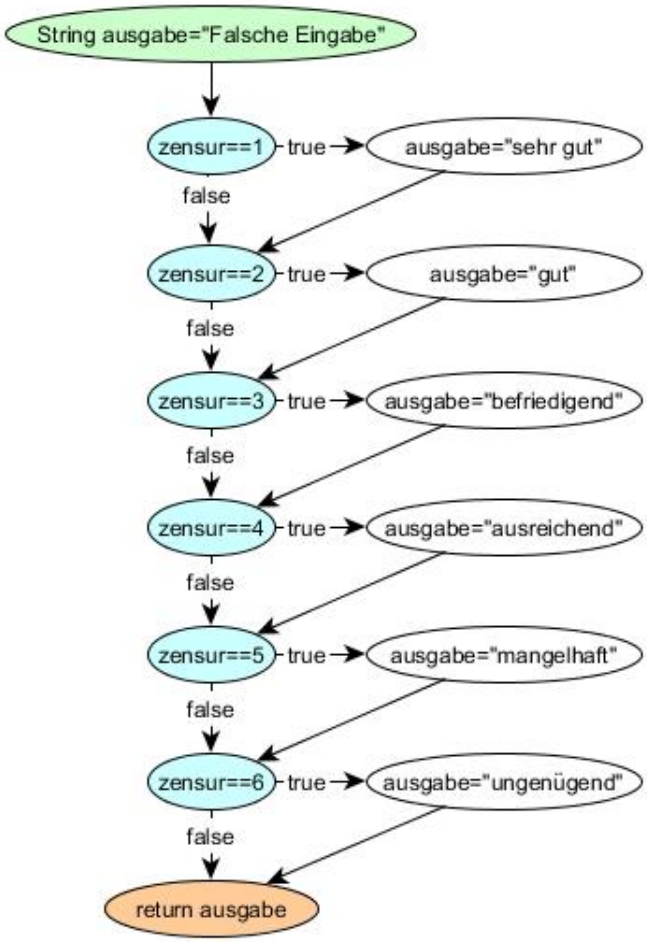
Zweite Variante ist sicherer, weil binäre Verzweigungen unübersichtlich werden; theoretisch sollten beide Formeln das zweite Ergebnis liefern
Problem: Methode 1 funktioniert nur mit einem einzigen Endknoten



b)

```
public static String zensurString(int zensur) {
    switch (zensur){
        case '1': return "sehr gut";
        case '2': return "gut";
        case '3': return "befriedigend";
        case '4': return "ausreichend";
        case '5': return "mangelhaft";
        case '6': return "ungenuegend";
        default: return "Falsche Eingabe";
    }
}
```

v(G) = 7



Design Patterns:

Erzeugung
- factory method: abstrakte Klasse, die als Schnittstelle zwischen Anwendung und noch nicht spezifizierter konkreter Klasse steht (implementiert nichts, kennt nur die später nötigen Funktionen
- Singleton: Klasse, die nur einmal instanziiert sein darf

Struktur
- composite: zusammengehörende Elemente (Knoten eines Baums) erben von der selben Klasse um alle zugleich zu verarbeiten (draw)
- facade: vereinfachter Zugang zu einem komplexen Subsystem (Verstecken der Komplexität)
- proxy: Stellvertreterklasse, der Verbindung zur richtigen Klasse kennt (wenn diese z.B. in einem anderen System liegt und Proxy die Kommunikation steuert; Klasse ist zu teuer selbst zu instanziieren (großes Bild -> Vorschau))

Verhalten
- command: Befehlsaufruf wird in Klasse gespeichert, und kann dann später ausgeführt werden
- observer: Klasse, die Beobachter sind, können sich bei zu beobachtender Klasse anmelden und werden informiert, wenn sich etwas ändert