Objektorientierte Programmierung WS 2013/2014 - Lösung 5

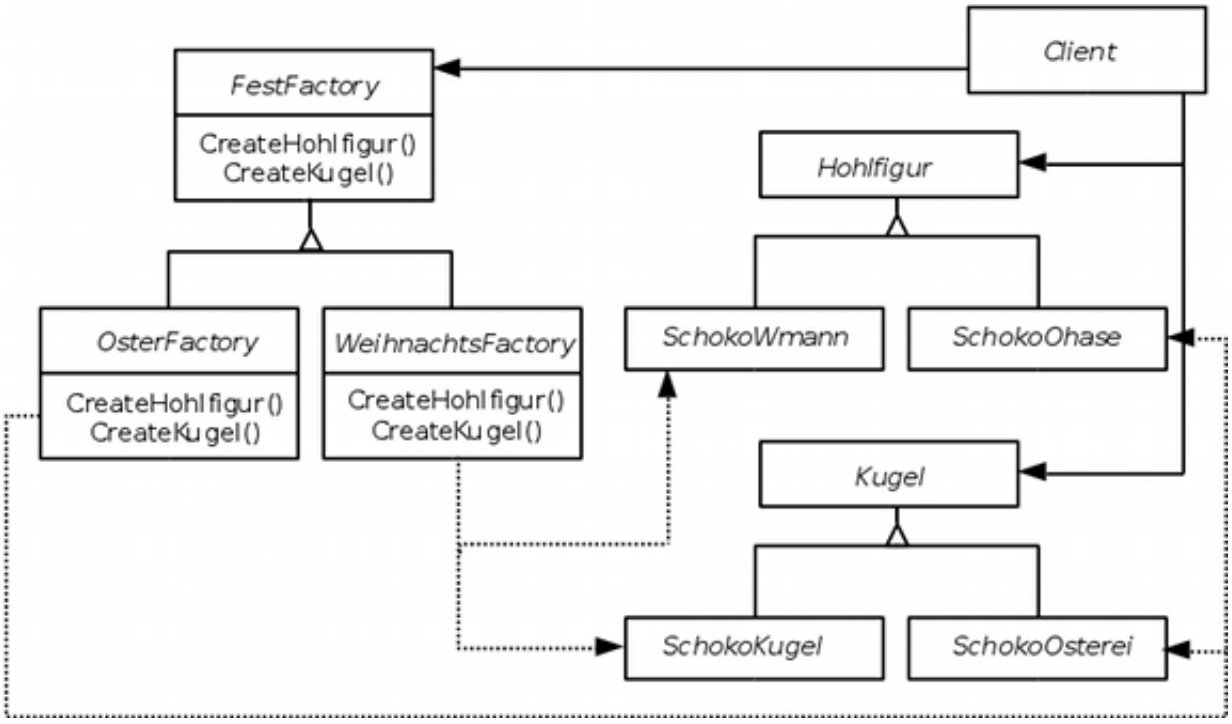
Prof. Dr. U. Kastens
Institut für Informatik, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn

Lösung zu Aufgabe 1

Die Rollen des Musters "Fabrikmethode" sind im C++-Text annotiert: `restaurant-loes.cc`

Lösung zu Aufgabe 2

Das Abstract Factory-Schema für die Festtagsfabrik sieht wie folgt aus:



Hier ist ein Lösungvorschlag für die Implementierung der Festtags-Fabrik in Java:

- `blatt5/afactory`

Lösung zu Aufgabe 3

Die Zuordnung der Rollen im Brücken-Muster zu den Klassen dieser Anwendung:

```
HTMLListPrinter      Concrete Implementor 1
ListPrinter          Refined Abstraction
PlainListPrinter     Concrete Implementor 1
PrinterImpl          Implementor
Printer              Abstraction
```

Erweiterungen des Beispiels um die charakteristischen Eigenschaften des Brücken-Musters zu demonstrieren:

- Die benutzte Implementierung kann dynamisch gewechselt werden: Dazu ändern wir die Klasse Client (`Client.java`):

```
package bridgesol;
public class Client {
   public static void main(String[] args) {
      Printer p = new ListPrinter(new HTMLListPrinter());
      String[] farben = {"rot", "grün", "blau"};
      p.print(farben);

      // Erweiterung dynamischer Wechsel
      p.setImpl(new PlainListPrinter());
      p.print(farben);
   }
}
```

- Unabhängig von der Abstraktion kann eine neue Implementierungsart ergänzt werden, zum Beispiel als Ausgabeformat eine kommaseparierte, geklammerte Liste: `CommaListPrinter.java`.
- Unabhängig von den Implementierungen kann eine verfeinerte Abstraktion hinzugefügt werden, z.B. "drucke eine einfache Folge von Strings": `SeqPrinter.java`.