Vermittler (Mediator)

Sabine Müller - Sven Richter - Jens Wagenbreth 03IN2-P-D

1. Einleitung

Vermittler (Mediator)
objektbasiertes Verhaltensmuster

Verhaltensmuster beschreiben Strukturen, die am Kontrollfluß innerhalb der Anwendung beteiligt sind. Sie konzentrieren sich also auf Algorithmen und die Delegation von Zuständigkeiten.
Die von ihnen verwendeten Strukturen entstehen zur Laufzeit und sind dynamisch (objektbasiert)

2. Zweck

Es wird das Zusammenspiel einer Menge von Objekten durch ein Objekt gekapselt.
Der Vermittler fördert die lose Kopplung von Objekten, indem er die Kommunikation zwischen den Objekten übernimmt. Dadurch kann das Zusammenspiel der Objekte variiert werden, unabhängig von den Objekten selbst.

3. Motivation

Objektorientierter Entwurf fördert die Verteilung von Verhalten zwischen Objekten. Das Ergebnis einer solchen Verteilung kann eine Objektstruktur mit vielen Beziehungen zwischen den Objekten sein; im schlimmsten Fall bedeutet dies, dass jedes Objekt jedes andere Objekt kennen muß.

Stellen Sie sich als Beispiel die dargestellte ‚Dialogbox in einer grafischen Benutzungsschnittstelle vor. Die Dialogbox verwendet ein Fenster, um eine Ansammlung von Widgets wie Knöpfe, Menüs und Eingabefelder zu präsentieren.

Es gibt oftmals Abhängigkeiten zwischen den Widgets in der Dialogbox. So muß z.B. ein Knopf deaktiviert sein, wenn ein bestimmtes Texteingabefeld leer ist. Die Auswahl eines Eintrags in einer Auswahlliste namens Listbox ändert möglicherweise den Inhalt eines Eingabefelds. Andererseits kann die Eingabe eines Textes in das Eingabefeld automatisch zur Auswahl eines oder mehrerer entsprechender Einträge in der Listbox führen.

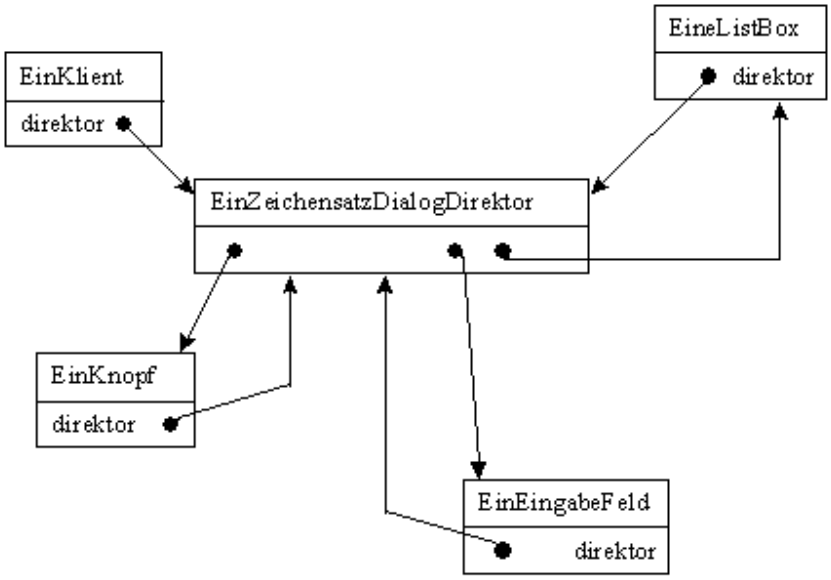
Sobald einmal ein Text im Eingabefeld erscheint, können andere Knöpfe aktiviert werden, die es dem Benutzer ermöglichen, etwas mit dem Text zu tun, so zum Beispiel das von ihm referenzierte Objekt zu ändern oder zu löschen.

Unterschiedliche Dialogboxen besitzen unterschiedliche Abhängigkeiten zwischen den Widgets. Obwohl also Dialoge dieselbe Art von Widgets anzeigen werden, können sie nicht einfach existierende Widgetklassen wieder verwenden; sie müssen auch dahingehend angepasst werden, dass sie dialogspezifische Abhängigkeiten wiedergeben. Sie durch

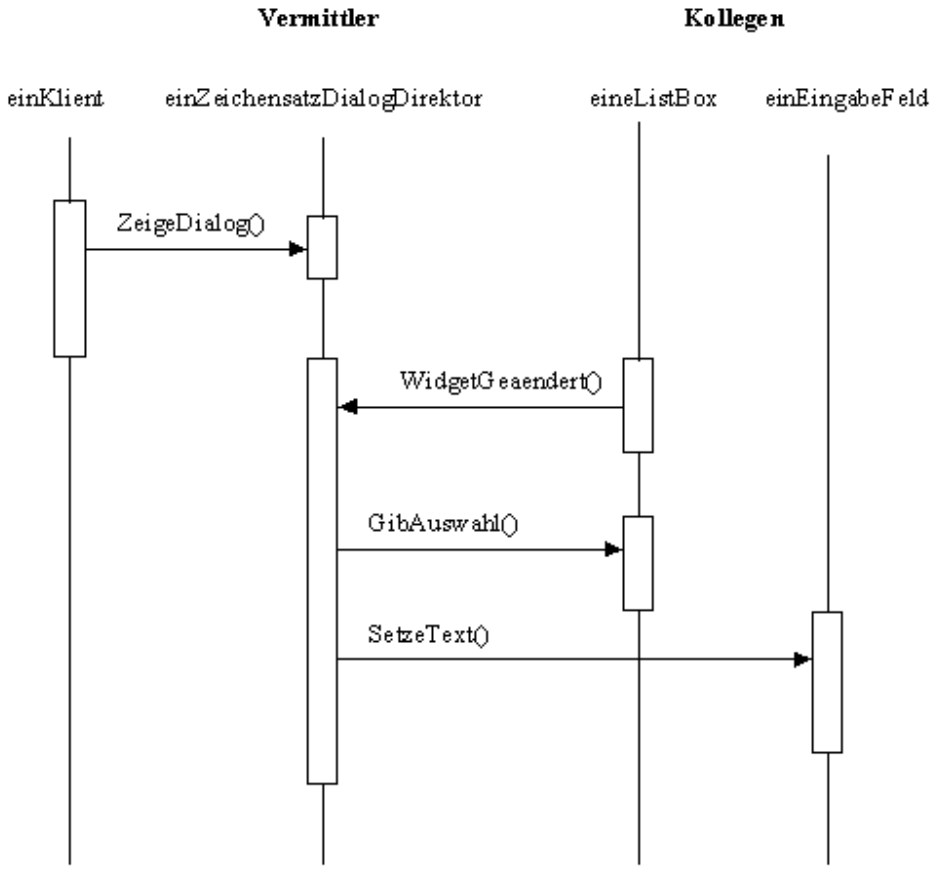
Unterklassenbildung maßzuschneidern ist sehr mühsam, da viele Klassen einbezogen werden müssen.

Sie können diese Probleme vermeiden, indem sie das Gesamtverhalten in einem separaten Vermittlerobjekt kapseln. Ein Vermittler ist für die Kontrolle und Koordination der Interaktion innerhalb einer Gruppe von Objekten zuständig. Der Vermittler hält die Objekte in einer Gruppe davon ab, direkt aufeinander Bezug zu nehmen. Die Objekte kennen nur den Vermittler und reduzieren dadurch die Anzahl ihrer Verbindungen.

So kann beispielsweise ein ZeichensatzdialogDirektor als Vermittler zwischen den Widgets in einer Dialogbox fungieren. Ein ZeichensatzDialogDirektor-Objekt kennt die Widgets in einem Dialog und koordiniert ihr Zusammenspiel. Es fungiert als die Nabe der Kommunikation zwischen den Widgets.

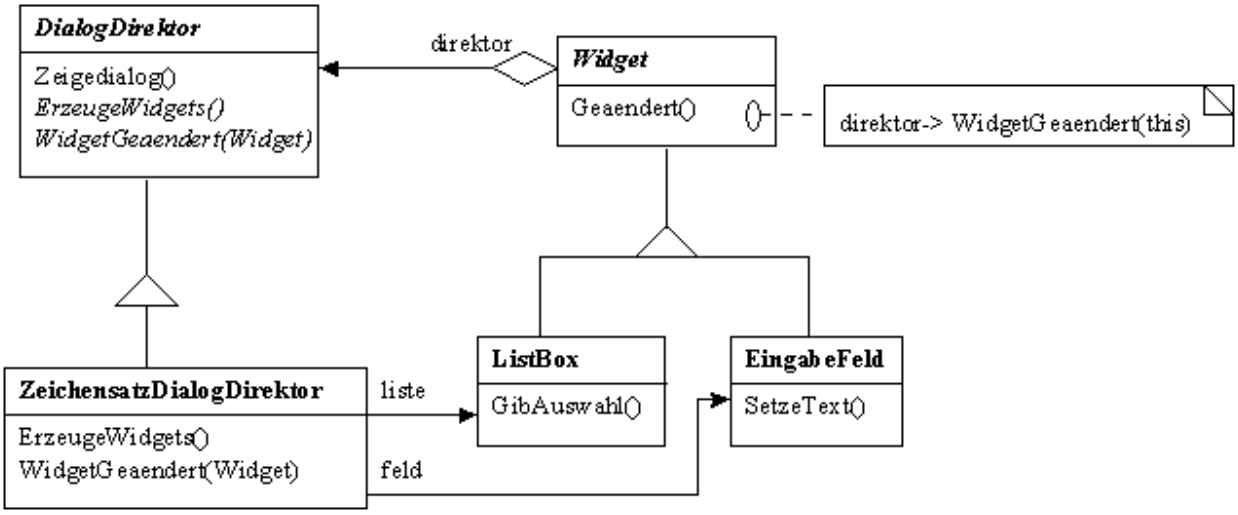


Anstatt daß sich die Listbox, das Eingabefeld und der Knopf direkt kennen und Bezug zueinander haben, wird hier ein ZeichensatzDialogDirektor eingesetzt. Der ZeichensatzDialogDirektor kennt als einziger alle Widgetobjekte und "vermittelt" zwischen ihnen, ohne daß die Widgetobjekte voneinander wissen. Die Widgetobjekte kennen nur den ZeichensatzDialogDirektor.
Sie wollen z.B. wenn sich die Auswahl in der ListBox ändert, soll der ausgewählte Eintrag im Eingabefeld erscheinen. Folgende Interaktionen werden im Vermittlermuster durchgeführt:



1. Die Listbox meldet ihrem Direktor, daß sie sich geändert hat.
2. Der Direktor holt sich die Auswahl von der Listbox.
3. Der Direktor gibt die Auswahl an das Eingabefeld weiter.

Nun können noch weitere Aktionen ausgelöst werden, weil das Eingabefeld geändert wurde. Die folgende Struktur hat das ebengenannte Beispiel:

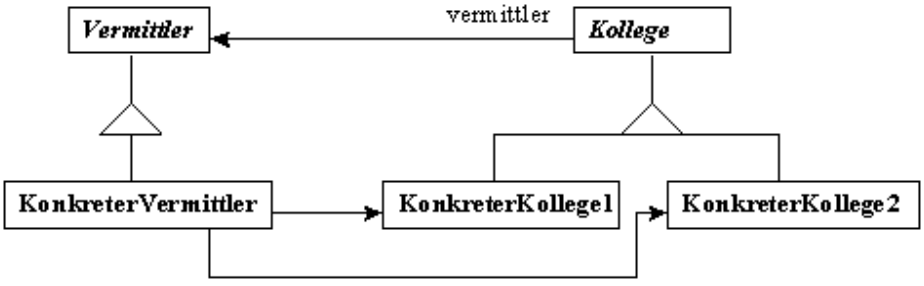


4. Anwendbarkeit

Das Vermittlermuster sollten Sie in den folgenden Situationen verwenden:

- Sie haben eine Menge von Objekten, die in komplexer Weise miteinander zusammenarbeiten. Die Abhängigkeiten zwischen den Objekten sind unstrukturiert und schwer nachvollziehbar.
- Ein Objekt bezieht sich auf viele andere Objekte und arbeitet mit denen zusammen. Dadurch ist es schwierig diese Objekt wiederzuverwenden.
- Ein auf mehrere Klassen verteiltes Verhalten soll, ohne viele Unterklassen zu bilden, maßgeschneidert werden.

5. Struktur



6. Teilnehmer

- Vermittler
  Definiert ein Schnittstelle für den Umgang mit den Kollegenobjekten.
- KonkreterVermittler
  - kennt und verwaltet seine Kollegenobjekte.
  - implementiert durch Koordination der Kollegenobjekte das Gesamtverhalten.
- KollegenKlassen
  - Jede Kollegenklasse kennt ihre Vermittlerklasse.
  - Jedes Kollegenobjekt arbeite mit seinem Vermittler zusammen und nicht mit seinen Kollegenobjekten.

7. Interaktion

Die Kollegen senden und empfangen Anfragen von/zu einem Vermittlerobjekt.
Der Vermittler leitet diese Anfragen zwischen den richtigen Kollegenobjekten weiter. Dadurch wird das Gesamtverhalten implementiert.

8. Konsequenzen

1. Es schränkt die Unterklassenbildung ein.
   Um das Verhalten des ganzen Systems zu ändern, muß lediglich die Vermittlerklasse abgeleitet werden. Die Kollegenklassen verwenden Sie ohne Änderung wieder.
2. Es entkoppelt Kollegenobjekte.
   Ein Vermittler unterstützt die lose Kopplung von Kollegenobjekten. Die Kollegen- und Vermittlerklassen sind unabhängig voneinander.
3. Es vereinfacht die Struktur eines Objekts.
   Die n-zu-n Beziehungen zwischen den Kollegnobjekten wird durch 1-zu-n Beziehungen zwischen dem Vermittler und den Kollegenobjekten ersetzt. 1-zu-n Beziehungen sind einfachen zu er- und bearbeiten.
4. Es abstrahiert davon, wie Objekte zusammenarbeiten.
   Sie können durch die Anwendung des Vermittlermusters sich auf die Interaktionen zwischen den Kollegenobjekten konzentrieren. Dies kann Ihnen dabei helfen, die Zusammenarbeit ihrer Objekte mit dem System zu erklären.
5. Es zentralisiert die Steuerung.
   Die Komplexität der Interaktion zwischen Kollegenobjekten wird durch die Komplexität des Vermittlers getauscht. Dadurch kann der Vermittler ein sehr kompliziertes und schwer zu wartendes Objekt werden.

9. Implementierung

1. Weglassen der abstrakten Vermittlerklasse.
   Eine abstrakte Vermittlerklasse ist nur dann Notwendig, wenn die Kollegenobjekte mit mehr als einem Vermittler zusammenarbeiten sollen. Das gleiche gilt auch für die abstrakte Kollegenklasse.
2. Die Interaktion von Kollegen- und Vermittlerklassen
   Die Benachrichtigung von Kollegenklassen zur Vermittlerklasse kann als Beobachtermuster implementiert werden.
   Ein anderer Ansatz ist, daß die Kollegenobjekte eine spezialisierte Schnittstelle zum Vermittler benutzen und sich selbst übergeben. So kann der Vermittler feststellen von wem er benachrichtigt wurde. Smalltalk/V für Windows benutzt diese Art der Delegation. Java nutzt eine Ähnliche.

10. Beispielcode

Java benutzt auch das Vermittlermuster. Java hat Listboxen, Eingabefelder und auch Buttons. Die Vermittlerklasse ist entweder eine von Applet abgeleitet Klasse oder eine von Frame abgeleitet Klasse.
Die Kollegenobjekte benachrichtigen den Vermittler unter Zuhilfenahme der ListenerInterfaces, dieses Interface wird durch Mehrfachvererbung in den Vermittler mitaufgenommen. Der Vermittler reagiert dann darauf und koordiniert.

```
public class ZeichensatzDialogDirektor extends Applet implements ItemListener{

      private Button      m_button;  // Das sind die Kollegenobjekte
              private List        m_list;
              private TextField  m_texfeld;

              public void Init(){
        m_button = new Button("OK");
        m_list = new List(4, false);
        m_list.addItem("chicago");
        m_list.addItem("courier");
        m_list.addItem("palatino");
        m_list.addItem("times roman");
        m_textfeld = new TextField(20);
        add(m_button);
        add(m_list);
        add(m_textfeld);
        addItemListener(this);
      }

              public void itemStateChanged(ItemEvent e)      // die Vermittlung zwischen
              {                                              // Listbox und TextField
                        String cmd = e.getItemSelectable();
              m_textfeld.setText(cmd);
              }

}
```

11. Bekannte Verwendungen

- Sowohl ET++ als auch die THINK-C-Klassen, sowie auch Borland Delphi benutzten die Vermittler in Dialogen.
- Smalltalk/V für Windows Anwendungsarchitektur basiert auf der Vermittlerstruktur.

12. Verwandte Muster

- Das Fassadenmuster gibt eine vereinfachte Schnittstelle zu einem komplexen System. Sein Protokoll ist unidirektional. Der Vermittler ermöglicht eine Zusammenarbeit von Objekten. Sein Protokoll ist multidirektional.
- Kollegenobjekte können mit Hilfe des Beobachtermuster den Vermittler benachrichtigen.

13. Quellen

http://www.fh-wedel.de/~si/seminare/ws97/Ausarbeitung/index.html

Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J. *Entwurfsmuster*. Addison-Wesley, 2001
.