FHWS

Hochschule für angewandte Wissenschaften
Fachhochschule Würzburg-Schweinfurt

Seminararbeit

Das MVC - Entwurfsmuster und Erweiterungen

vorgelegt an der
Hochschule für angewandte Wissenschaften Würzburg-Schweinfurt
in der Fakultät Informatik und Wirtschaftsinformatik
FWPM Design Patterns

bei Professor:	Prof. Eberhard Grötsch
Abgabetermin:	23. Mai 2013
Eingereicht von:	Florian Beckh Christian Petry Johannes Krenig 5109038

Inhaltsverzeichnis

1 MVVM

1.1 WPF und MVVM-Entwurfsmuster

Zum besseren verstehen des Model View View Model (MVVM) wird im folgenden Abschnitt zuerst das Grundprinzip von Windows Presentation Foundation (WPF) erklärt um folgend auf das eigentliche Thema MVVM überzugehen.

1.1.1 WPF

Mit .NET Framework 3.0 wurde WPF als Grafikframework für Windows Anwendungen eingeführt. Das rendern und zeichnen von Inhalten findet mittels DirectX auf der Graphic Processor Unit (GPU) statt und ermöglicht es den GUI-Designern die Erstellung von optisch sehr ansehnlichen Benutzeroberflächen, bei gleichzeitig Entlastung der Central Processor Unit (CPU). In Gegensatz zur altbekannten WinForm-API trennt die WPF die Präsentations- und Geschäftslogik nach strengen Richtlinien. Während die Geschäftslogik in Programmcode geschrieben ist, wird die Präsentation mittels der Auszeichnungssprache Extensible Application Markup Language (XAML) deklariert. XAML ist wie der Name schon impliziert an XML angelehnt (siehe **??**). Wohingegen die Idee eine Auszeichnungssprache als Metasprache für die Benutzeroberfläche zu verwenden schon von HTML bekannt ist.

```
1 <xml>
2
3
4 </xml>
```

Listing 1.1: Beispiel XAML

Die wichtigsten Eigenschaften einer WPF-Anwendung im Überblick:

- Design mit der Auszeichnungssprache XAML, alternativ auch ausprogrammiert möglich.
- Unterstützung von 2D und 3D Grafiken
- Die Ausgabe ist vektorbasiert anstatt pixelbasiert. Daraus erfolgt eine bessere Skalierbarkeit der Bildschirmausgabe.
- Vielfältige Datenbindungsmöglichkeiten

- Grafikberechnungen auf der GPU anstatt CPU

Die strikte Trennung von Oberflächengestaltung und Code ermöglicht es, dass Programmentwickler und Oberflächendesigner vollkommen unabhängig voneinander Code und Benutzeroberfläche entwickeln können.

1.1.2 MVVM

Das MVVM Entwurfsmuster wurde von John Gossman als Variante des Model View Presentation (MVP) Entwurfsmuster auf seinem Blog vorgestellt. Hintergrund war die fortschreitende Entwicklung moderner Benutzerinterface-Frameworks, wie WPF und Silverlight von Microsoft. In besonders großen Anwendungen ist das MVVM sinnvoll, in kleiner Anwendungen, ist es aufgrund des Overheads nicht wirklich zu empfehlen. Die Abkürzung MVVM steht wie bei Model View Controller (MVC) und MVP für die Komponenten des Entwurfsmusters (Siehe Abbildung 1.1). Die Langform Model View ViewModel ist auf den ersten Blick ungewöhnlich und bedarf der Erklärung.

- Model: Datenzugriffsschicht für die Inhalte, dem Benutzer angezeigt und von ihm manipuliert werden. Dazu benachrichtigt es über Datenänderungen und führt eine Validierung der vom Benutzer übergebenen Daten durch. Hierdurch wird vor allem in der View der Code-Behind minimiert.

- View: Alle durch die GUI angezeigten Elemente. Es bindet sich an Eigenschaften des ViewModel, um Inhalte darzustellen und zu manipulieren sowie Benutzereingaben weiterzuleiten. Durch die Datenbindung ist die View einfach austauschbar und ihr Code-Behind gering.

- ViewModel: beinhaltet die UI-Logik (Model der View) und dient als Bindeglied zwischen View und obigem Model. Einerseits tauscht es Information mit dem Model aus, ruft also Methoden oder Dienste auf. Andererseits stellt es der View öffentliche Eigenschaften und Befehle zur Verfügung. Diese werden von der View an Steuerelemente gebunden, um Inhalte auszugeben bzw. UI-Ereignisse weiterzuleiten. Insgesamt wird CRUD ermöglicht. Das ViewModel darf dabei keinerlei Kenntnis der View besitzen.

Die vom MVVM genutzte funktionale Trennung und Datenbindung des MVC wird dazu genutzt, die bereits beschriebene lose Kopplung zu erreichen. Diese ist jedoch im MVVM um einiges stärker verwirklicht als in MVC und MVP. Ein Vorteil des MVVM in der Entwicklung von Anwendungen ist zudem die bessere Einbindung der UI-Logik in Unit Tests. Aufgrund der Tatsache, dass Sie beim MVVM die Logik und das Command in der ViewModel-Klasse haben möchten, haben sich bei diesem Entwurfsmuster Commands wie das ActionCommand gegenüber den RoutedCommands durchgesetzt. Ein ActionCommand erlaubt beim Instanziieren direkt die Angabe eines Delegates, der auf eine Methode im ViewModel zeigen kann. MVVM in Verbindung mit WPF verwendet üblicherweise eine Implementierung mit Delegates basierend auf ICommand-Interfaces.

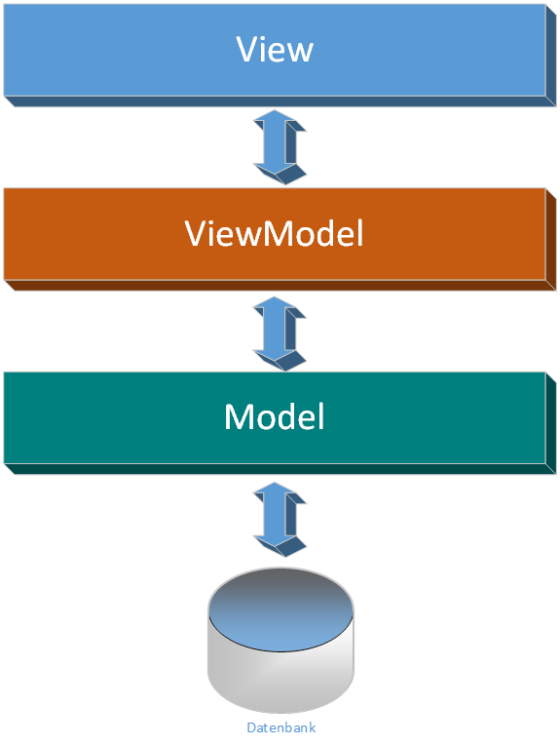

Abbildung 1.1: MVVM-Entwurfsmuster

1.2 Problem

1.3 Lösung

Eine bessere Trennung der Schichten ermöglicht größere Flexibilität

1.4 Konsequenz

1.5 Beispiel

Literaturverzeichnis

Listings

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

CPU Central Processor Unit 1

GPU Graphic Processor Unit 1

MVVM Model View View Model 1

MVC Model View Controller 2

WPF Windows Presentation Foundation 1

MVP Model View Presentation 2

XAML Extensible Application Markup Language 1