4. OOHDM

In diesem Kapitel wird ein Überblick über die verwendete Designmethode OOHDM gegeben. Es werden die einzelnen Designphasen besprochen und die Notation und deren Semantik erläutert.

4.1 Einleitung

Die Methode OOHDM (Object Oriented Hypermedia Design Method) ist ein Modell-basierter Ansatz, um große Hypermediaanwendungen zu entwerfen. OOHDM wurde von Gustavo Rossi und Daniel Schwabe entwickelt [1],[2],[3]. Ziel ist, eine Softwareentwicklungsmethode für Hypermediaanwendungen bereit zu stellen. Herausragend ist die weitreichende Form der Navigation, welche das Paradigma Hypertext zur Verfügung stellt. Sie ermöglicht dem Benutzer über eine bestimmte Informationsbasis zu navigieren und dabei komplexe Aufgaben zu erledigen. Die Erstellung von derartigen Applikationen ist ein schwieriges Unterfangen. Das erste Ziel einer solchen Anwendung ist eine gut funktionierende und durchdachte Navigation zu erstellen. Weiterhin unterscheiden sich das User-Interface einer normalen Applikation stark von dem einer Hypermediaapplikation. Es muß z.B. festgelegt werden, welche Events vom Benutzer in den Anzeigenelementen nur lokale Effekte in der Benutzerschnittstelle bewirken und welche Events eine Navigation auslösen. Der Blick auf die Entwicklung von herkömmlichen Applikationen zeigt, daß sich das objektorien-

tierte Softwareengineering bewährt hat. Demnach sollte die Methode für die Erstellung von Hypermediaapplikationen objektorientiert sein und die oben genannten Problembereiche berücksichtigen.

OOHDM gliedert sich in die vier nachfolgend beschriebenen Phasen:

- **Conceptual Design**: Im Conceptual Model wird ein Domänenmodell des Problembereichs erstellt, in dem alle Datenobjekte mit ihren unterschiedlichen Beziehungen modelliert werden. OOHDM benutzt dazu die Syntax von den gängigen objektorientierten Designmethoden wie UML oder OMT.

- **Navigational Design:** Hier werden verschiedene Ansichten auf die Applikation, wie sie im Conceptual Design dargestellt wurden, modelliert. Zusätzlich werden die Navigationsmöglichkeiten, die der Benutzer hat, graphisch modelliert. OOHDM definiert zwei verschiedene Diagramme für das Navigational Model: das Navigational Schema und das Contextual Schema.

- **Abstract Interface Design:** Hier wird spezifiziert welche Benutzerschnittstellenelemente dem Benutzer angezeigt werden. Zusätzlich werden die Auswirkungen von Ereignissen (Benutzerinteraktionen), die nicht der Navigation dienen, festgelegt. Die Datenobjekte aus dem Conceptual Design werden mit den Abstract Data Views verknüpft, um die Relation zu der Datenbasis herzustellen.

- **Implementierung:** Die Implementierung ist in der Methode direkt als letzter Schritt spezifiziert.

Demnach bietet OOHDM für die obengenannten Problembereiche einer Hypermediaapplikation genügend Unterstützung. OOHDM unterstützt die Navigation in zwei Schemata und die Bildung von der Benutzerschnittstelle im Abstract Interface Design. Allem voran wird eine Phase der Definition des Anwendungsbereich gestellt. Die Grundsteine von OOHDM sind nachfolgend beschrieben.[7]

- Navigationsobjekte sind spezielle Ansichten von den Objekten des Conceptual Designs

- Navigationskontexte beschreiben und strukturieren auf einem angemessenen Abstraktionsniveau den Navigationsraum der Anwendung.

- Das Design der Navigation wird vom Design der Benutzerschnittstelle getrennt.
- Es gibt spezifische Designentscheidungen, die erst während der Implementierungsphase gemacht werden müssen und deswegen nicht in der Abstract Interface Designphase realisiert werden.

Die weiteren Unterkapitel gehen näher auf die einzelnen Phasen ein.

4.2 Conceptual Design

Während der Conceptual Design Phase wird ein statisches Objektmodell des Anwendungsgebiets aufgebaut. Zur Erstellung werden die geläufigen Schritte angewandt, wie sie beispielsweise in OMT vorgeschlagen werden. [44] OOHDM legt sich auf keine Beschreibungssprache für das Conceptual Design fest. Es kommen alle gängigen Methoden, wie UML oder OMT, für die Objektmodellierung in Betracht[6]. Für das Objektmodell wird das statische Klassendiagramm benutzt. Als einzige Erweiterung sieht OOHDM die Überführung von nicht-gerichteten Relationen in gerichtete Relationen vor. Diese werden mit Pfeilen gekennzeichnet und entsprechen der Berücksichtigung der Navigationsrichtung zwischen den Objekten beim Navigational Design.

Es soll hier nicht weiter auf das Conceptual Design eingegangen werden, da es den klassischen Objektmodellen entspricht. Es wird hier auf die Dokumentation der Methode OMT [45] oder der Beschreibungssprache UML ([28],[29], [30] und [31]) verwiesen.

Im Conceptual Design wurde festgehalten, aus welchen Objekten die Applikationsgebiet besteht und in welcher statischen Beziehung sie zueinander stehen.

4.3 Navigational Design

Während das Conceptual Design noch mit traditionellen Methoden erstellt wird, geht das Navigational Design auf die spezielle Thematik “Navigation” in einer Hypermediaapplikation ein. Der Schritt beschreibt losgelöst von der Beschrei-

bung der Oberfläche die möglichen Navigationsobjekte, basierend auf den Klassen des Conceptual Designs. Das Navigational Design repräsentiert demnach die verschiedenen Sichten auf die Klassen des Conceptual Design.

Eine spezielle Klasse, der Kontext, beschreibt eine Menge von Objekten, welche durch eine Einschränkung (Query) über die gesamten Objekte spezifiziert wird. OOHDM führt dazu eine eigene Anfragesprache ein. Kontexte können auch ineinander geschachtelt werden. Die verschiedenen Kontextarten, die OOHDM spezifiziert, sind weiter unten beschrieben.

Während der Entwicklung des Navigational Designs werden folgende Aspekte untersucht:

- Welche Objekte spielen bei der Navigation eine Rolle, welche Attribute haben sie und wie sind die Relationen zwischen den Objekten?
- In welchem Kontext wird der Benutzer navigieren?
- Hat das Objekt unterschiedliche Attribute je nach dem in welchem Kontext der Benutzer sich befindet?
- Von welchem Typ sind die Verbindungen zwischen den Objekten und über welche Zugriffsstruktur werden sie realisiert? (Links, Indizes etc.)
- Was passiert, wenn der Benutzer zwischen Objekten hin- und herspringt, d.h. den Kontext wechselt?

Wie schon weiter oben in den Grundbausteinen von OOHDM beschrieben, werden im Navigational Design bestimmte Ansichten auf Klassen des Conceptual Designs erstellt. Dies geschieht im Navigational Schema. Ein weiterer Grundstein, die Navigational Kontexte, werden im Contextual Schema definiert. Das Navigational Design wird demnach in zwei Schemata spezifiziert:

- Das **Navigational Schema** stellt eine Sicht vom Standpunkt der Navigation auf die Klassen des Conceptual Schemas dar. Es bedient sich der gleichen Notation und der gleichen Konzepte. Zusätzlich werden aber spezielle Navigationsklassen eingeführt wie Indizes, Links und Knoten. Weitere Änderungen können das Wegfallen von Klassen bzw. das Anreichern von Klassen durch fremde Attribute (welche durch z.B. 1-1 Relationen zugreifbar sind) sein. Navigationsklassen können ein Subset der

Attribute ihrer entsprechenden Klassen im Conceptual Schema enthalten. Sie repräsentieren somit ein spezielle Sichtweise auf die Klasse des Conceptual Schemas, welche sich in Abhängigkeit von einem Kontext ändern kann (siehe dazu den nächsten Punkt).

- Das **Contextual Schema** erläutert und erklärt den Navigationsraum mit Hilfe von Kontextklassen. Ein Kontext ähnelt einer Sammlung von Elementen. Weiterhin kann ein Kontext geschachtelt werden.

Es gibt sechs verschiedene Kontextarten:

- **Klassenbasierter Kontext**:
  Alle Objekte gehören der selben Klasse C an und können durch ein gemeinsames Attribut ausgewählt werden. Ein Beispiel wären alle Partner des Accounttyps “Händler”. Der Accounttyp ist Attribut der Klasse Partner.

- **Klassenbasierte Kontextgruppe:**
  Dies ist eine Menge von klassenbasierten Kontexten, welche dadurch gebildet wird, daß ein Parameter der Klasse alle möglichen Werte annimmt. Ein Beispiel wäre Partner nach Accounttyp. Der Accounttyp ist Attribut der Klasse Partner.

- **Link-basierter Kontext:**
  Auch hier sind die Objekte von einem Klassentyp C und werden über eine 1-n Relation selektiert. Z.B. alle Kunden von Accountmanager Herrn Schmidt. Kunde zu Accountmanager ist eine 1-n Relation.

- **Link-basierte Kontextgruppe:**
  Dies ist eine Menge von Link-basierten Kontexten, die aus einer 1-n Relation entsteht, indem nicht ein Quellobjekt festgelegt wird, sondern diese Relation für alle Quellobjekte aufgelöst wird. Z.B. Account nach Accountmanager. Kunde zu Accountmanager ist eine 1-n Relation.

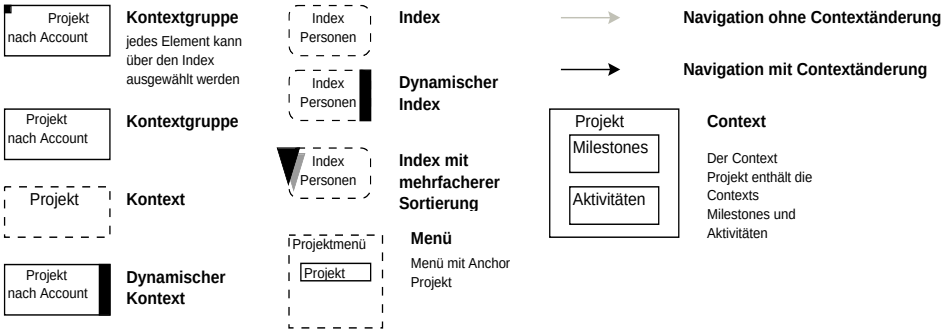
- **Enumerierte Kontextklasse:**
  Diese Kontextklasse beschränkt sich nicht nur auf Objekte einer Klasse, sondern enthält Objekte aus verschiedenen Klassen. Ein Beispiel hierfür wären alle Notizen und Erinnerungen.

- **Dynamische Kontextklasse:**
  In die dynamische Kontextklasse können während der Navigation die in ihr enthaltenen Objekte gelöscht oder neue hinzugefügt werden.

In Abbildung 4–1 ist die Syntax des Contextual Schemas angegeben. Das Schema besteht aus den Komponenten:

- **Zugriffsstrukturen:** Ein Zugriffstruktur wie z.B. ein Index oder eine Guided Tour erlaubt es dem Benutzer über eine Auswahl von mehreren Objekten zu navigieren, sie auszuwählen und damit in den Context zu übernehmen, welcher der Zugriffsstruktur nachfolgt. Der Benutzer muß immer wieder zur Zugriffsstruktur zurückkehren, um ein anderes Objekt zu selektieren.
- **Context:** die verschiedene Kontextarten wurden oben beschrieben.
- **Link:** die Verknüpfung von verschiedenen Kontexten.

Abbildung 4–1 Syntax des Contextual Schemas



Zusätzlich zu Abbildung 4–1 muß angemerkt werden, daß alle Kontexte und Kontextgruppen, die über einen Index gesteuert werden ein kleines schwarzes Rechteck links oben in der Ecke haben. Auf dem Diagram ist eine Kontextgruppe mit dieser Notation dargestellt.

Die sechs Kontextarten werden zusammengefaßt in drei verschiedene Notationen. Hierbei wird zwischen Kontext und Kontextgruppe differenziert. Die Unterscheidung der Link-basierten Kontexte und der einfachen Kontexte ist implizit. Liegt dem Kontext eine Relation zugrunde so ist er Link-basiert, ansonsten ist er Klassen-basiert.

Für jeden Navigational Kontext muß außerdem noch eine Spezifikation geschrieben werden, in der die Abfragen, welche den Kontext einschränken, spezifiziert sind. Die genaue Syntax ist in [7] beschrieben.

4.4 Abstract Interface Design

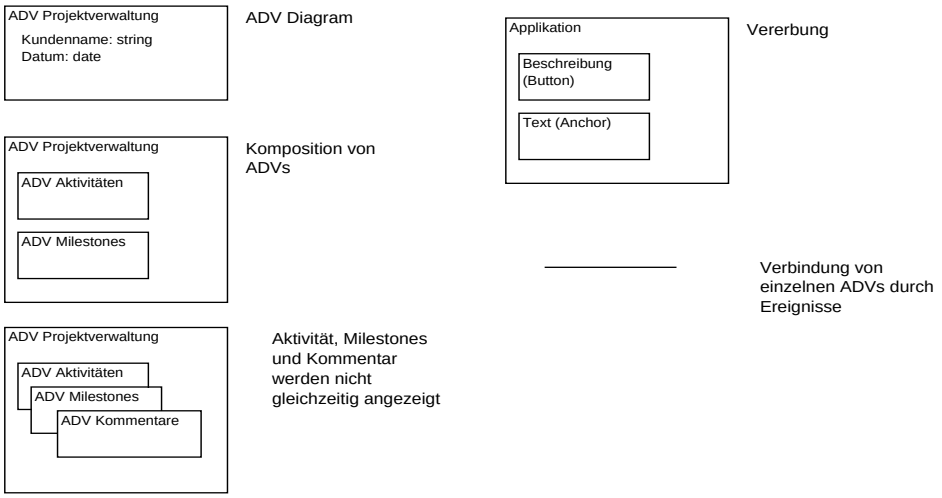
Dem Navigational Design folgt der Entwurf der Benutzerschnittstelle. Hier wird definiert wie die Navigationsobjekte dargestellt werden. Es muß unterschieden werden zwischen Ereignissen vom Benutzer, die navigationsspezifisch sind, und Ereignissen, die eine Veränderung der Darstellung von Navigationsobjekten bewirken. Die Trennung von Navigation und Benutzerschnittstelle erlaubt es, für ein Navigationsdesign mehrere Benutzerschnittstellen zu entwerfen. Es wird eine klare Trennung von Darstellung und Applikationslogik forciert.

OOHDM benutzt die Notation der Abstract Data Views (ADV) für das Design des User-Interfaces. ADVs haben einen internen Zustand und eine Schnittstelle. Sie können auf externe (vom Benutzer erzeugte) Ereignisse reagieren. Benutzerereignisse lösen entweder einen “Navigationsschritt” aus oder beeinflussen die Anzeige. Beispielsweise könnte ein Objekt in einer Grobansicht und einer Detailansicht angezeigt werden. Der Wechsel zwischen den Ansichtsarten erfordert ein Event vom Benutzer. Dieser Event löst aber keine Navigation aus, sondern beschränkt sich auf die Darstellung. Falls aber nun aus der Detailansicht ein Link zu einer anderen Klasse verfolgt wird, handelt es sich hierbei um eine “Navigationsschritt”.

Die Ereignisse, auf die ein ADV reagieren kann, sind beschränkt auf die vom Browser zur Verfügung gestellten Ereignisse. Hierzu gehören auf jeden Fall MouseClick, MouseOver, MouseUp, etc.

ADVs können durch Contains-Beziehungen oder durch Vererbung in Relation gebracht werden. Abbildung 4–2 zeigt die Notation eines ADV Diagramms.

Abbildung 4–2 Syntax von ADV Diagrammen



Die Dynamik der Benutzerschnittstelle wird mittels ADVCharts beschrieben. Sie zeigen den Einfluß den Ereignisse des Benutzers (MouseClick, etc.) auf die Darstellung der Interfaceobjekte haben. Ereignisse, welche die Navigation beeinflussen, werden vom Navigational Design abgedeckt.

Weitere Literatur bezüglich OOHDM sind [1], [2], [3], [4], [5], [6], [7], [8], [9] und [10].