Mobile MedioVis.

Konzeption und Entwicklung eines visuellen Suchsystems für die Mediothek der Bibliothek Konstanz auf einem Pocket PC.

Wissenschaftliche Arbeit zur Erlangung des Grades eines Bachelor Information Engineering im Fachbereich Informatik & Informationswissenschaft der Universität Konstanz

Verfasser:
Carsten Marx
Wollmatingerstr. 110
78467 Konstanz
Matr.Nr. 01 / 496473
Carsten.Marx@uni-konstanz.de

1. Gutachter: Prof. Dr. Harald Reiterer
2. Gutachter: Prof. Dr. Daniel A. Keim

Konstanz, den 19.04.2005

Zusammenfassung (deutsch)

In dieser Bachelor-Arbeit wird die Konzeption und Entwicklung eines visuellen Suchsystems für die Mediothek der Bibliothek Konstanz auf einem Pocket PC beschrieben. Bei der Mediothek handelt es sich um den Teilbereich der Bibliothek, in dem vor allem multimediale Titel (DVDs, Videos, CDs, usw.) untergebracht sind. Das visuelle Suchsystem, genannt MOBILE MEDIOVIS, ermöglicht dem Benutzer in einer geleiteten Suche diesen Informationsraum zu filtern. Dies geschieht durch die Pocket SieveMap, in der definierte visuelle Cluster der Mediothek thematisch in Kategorien angeordnet sind. Durch direkte Manipulation mit den Clustern wird eine Suchanfrage dynamisch erstellt. Anschließend werden die gefilterten Titel in einer tabellenbasierten Ansicht, der Mobile Media Grid, dargestellt und der Benutzer hat die Möglichkeit direkt in dieser Ansicht Detailinformationen zu einzelnen Mediothekstiteln zu erhalten ohne den Kontext zu verlassen. MOBILE MEDIOVIS ist als voll funktionsfähige Anwendung im .NET Compact Framework von Microsoft verfügbar und beruht auf dem Projekt MEDIOVIS, welches ebenfalls in der Arbeitsgruppe Mensch Computer Interaktion im Fachbereich Informatik & Informationswissenschaft entstanden ist.

Abstract (english)

This Bachelor Thesis illustrates the conception and development of a visual information seeking system for the Mediothek of the library of the University of Konstanz. The Mediothek is a subset of the library catalogue containing multimedia-based items like DVDs, viedos, CDs and so on. The visual information seeking system, called MOBILE MEDIOVIS, offers a managed search for the Mediothek. This is provided by the Pocket SieveMap with predefined visual clusters arranged in categories. A dynamic query is built via direct manipulation. After execution the filtered items are displayed in a table-based view in which the user can get details on demand without losing the overview. MOBILE MEDIOVIS is available as a full functional application, written in C# in the .NET Compact Framework. The idea for this project relies on the project MEDIOVIS which was also developed in the work group Human Computer Interaction at the department of computer and information science.

Inhaltsverzeichnis

1 Einleitung

Das richtige Medium aus einer Fülle von Gleichartigen zu finden spielt in heutigen digitalen Bibliotheken eine immer bedeutendere Rolle. Die Menge von Medien wie DVDs, CDs und anderen digitalen wie auch analogen Medien nimmt immer mehr zu. Durch die Katalogisierung dieser Medien in digitalen Bibliotheken besteht ein effizienter Zugriff auf die Medien bzw. deren Metadaten. Doch allein der effiziente Zugriff auf diese Daten reicht nicht aus um auch effizient und vor allem effektiv in diesem Informationsraum zu suchen. Es besteht also der Bedarf nach effizienten Suchsystemen. Mit MEDIOVIS wurde von der Arbeitsgruppe Mensch Computer Interaktion bereits ein effektives visuelles Suchsystem umgesetzt [Grün2004]. Durch die schnelle Verbreitung von immer leistungsfähigeren mobilen Geräten wird der Aspekt, überall zu jeder Zeit die gewünschten Informationen zu haben und nach Informationen zu suchen, immer populärer. Mobile Geräte finden eine immer größere Verbreitung. Sie bringen aber auch zusätzliche Rahmenbedingungen mit, die bei der Entwicklung visueller Suchsysteme berücksichtigt werden müssen.

Benutzer mit unterschiedlichstem Vorwissen, aber vor allen solche, die nicht als Experten bezeichnet werden können, haben immer wieder Probleme mit herkömmlichen visuellen Suchsystemen, die über eine Suchmaske die Suchparameter festlegen und dann als Ergebnis eine Trefferliste bieten. Oftmals dauert der Prozess die richtige Suchanfrage zu formulieren lange und der Benutzer bekommt keinen Überblick über seinen eigentlichen Informationsraum und die darin enthaltenen Medien. Gerade der unerfahrene Benutzer wird dadurch verwirrt und findet in vielen Fällen nur über Umwege und mit erheblichem Zeiteinsatz die richtigen Medien. Aber auch Expertenbenutzer sind in traditionellen Suchsystemen oftmals dazu gezwungen komplexe Suchanfragen mit einem erheblichen Aufwand in der Suchmaske zu definieren.

In dieser Arbeit wird ein kompletter Entwicklungsprozess für ein visuelles Suchsystem beschrieben, welches auf einem mobilen Gerät durch Kombination zweier Visualisierungskomponenten den Suchprozess, insbesondere den Suchanfrageformulierungsprozess, beschleunigt. Dabei wird dem Benutzer ein Gefühl für den Informationsraum, in dem er sich befindet, gegeben, und dieses visuelle Suchsystem wurde konzipiert um auf einem in der Größe beschränktem Bildschirm gut bedienbar zu sein.

1.1 Aufbau der Arbeit

Nach der Einleitung werden nun im weiteren Verlauf des ersten Kapitels zuerst weitere Grundlagen und Begrifflichkeiten eingeführt, auf die immer wieder in der Arbeit zurückgegriffen wird. Zunächst wird MEDIOVIS als bereits bestehendes Projekt und Ausgangspunkt beschrieben. Aufbauend darauf wird die Idee, die hinter dem Projekt MOBILE MEDIOVIS steht, erläutert.

Im Kapitel *Technische Rahmenbedingungen* werden die Grundlagen und die daraus resultierenden Einschränkungen und Bedingungen, die an ein visuelles Suchsystem für die Mediothek gestellt werden müssen, erörtert. So wird hier auf die Besonderheiten von mobilen Geräten bei der Hardware sowie bei der Software eingegangen. Des Weiteren wird die zur Umsetzung des Projektes verwendete Art der mobilen Geräte, die so genannten Pocket PCs, beschrieben. Als Resumée dieses Kapitels wird die Auswahl der verwendeten Implementierungsplattform begründet.

Im Kapitel *Visualisierung* wird zu Beginn eine Auswahl bestehender Visualisierungen für kleine Bildschirme vorgestellt. Anschließend wird das Konzept, welches hinter dem visuellen Suchsystem steht, erläutert. Dabei wird besonders auf die notwendigen Anpassungen und Änderungen der Visualisierungen bzgl. der kleinen Bildschirmgröße des Pocket PC eingegangen.

Die nächsten beiden Kapitel beschäftigen sich dann mit der Umsetzung der Visualisierung. Auf den Portierungsprozess des Datenmodells wird zum Anfang des Kapitels *Implementierung* eingegangen. Des Weiteren wird dann im Kapitel *MOBILE MEDIOVIS* die Interaktion des Benutzers mit der Anwendung MOBILE MEDIOVIS im Detail beschrieben.

Abschließend werden in der *Schlussbetrachtung* die Erfahrungen mit der benutzten Implementierungsplattform, dem .NET Compact Framework, berichtet, und es werden weitere mögliche Funktionen von MOBILE MEDIOVIS vorgestellt.

1.2 MedioVis

Um nach der Motivation und der Einführung in die Arbeit das Projekt MOBILE MEDIOVIS als visuelles Suchsystem einzuordnen, soll zuerst das Projekt MEDIOVIS[1] erklärt und beschrieben werden.

Bei MEDIOVIS handelt es sich um ein Projekt der Arbeitsgruppe Mensch Computer Interaktion im Fachbereich Informatik und Informationswissenschaft an der Universität Konstanz. Ziel des Projektes MEDIOVIS ist die Realisierung einer visuellen Benutzerschnittstelle zur freien sowie zielgerichteten Exploration des Bestands von multimedialen Titeln (Videos, DVDs, CD-ROM, usw.) der Universitätsbibliothek Konstanz. Dieser Bereich der Bibliothek wird als Mediothek bezeichnet. Mit MEDIOVIS soll ein wesentlicher Beitrag zur Verbesserung und Erweiterung der nutzerorientierten Dienstleistungen der Bibliothek Konstanz erzielt werden, da MEDIOVIS dem Bibliotheksnutzer ein effizientes Suchen und Browsen durch innovative Visualisierungen und Interaktionskonzepte ermöglicht und ihn von der ersten Eingrenzung des Gesamtkatalogs bis zur Selektion des gewünschten Titels begleiten wird. MEDIOVIS wird fortlaufend, in Kooperation mit der Bibliothek Konstanz, weiterentwickelt und ist bereits testweise in der Mediothek zum Einsatz gekommen [Grün2004].

Es werden immer wieder neue, innovative Visualisierungen in MEDIOVIS integriert und getestet. Abbildung 1,2 und 3 zeigen eine Auswahl dieser Visualisierungen und deren Kombination.

In Abbildung 1 ist die Standard-Ansicht dargestellt, welche dem Benutzer nach Eingabe eines Suchbegriffes erscheint. Hierbei findet sich der Benutzer von MEDIOVIS in einer sortierbaren Tabelle der Treffer wieder und beim Überfahren eines Titels mit der Maus wird dieser farblich hervorgehoben und es erscheinen die vollständigen Detailinformationen im linken unteren Bereich. Der nun fokussierte Titel kann durch Mausklick selektiert werden, worauf er in der Liste *Markierte Titel* im rechten unteren Bereich erscheint. Diese Titel können per Mail versendet, als Textdatei gespeichert, ausgedruckt oder direkt reserviert werden.

---

[1] Aktuelle Informationen über das Projekt MEDIOVIS finden Sie unter: http://hci.uni-konstanz.de/research/projects/mediovis

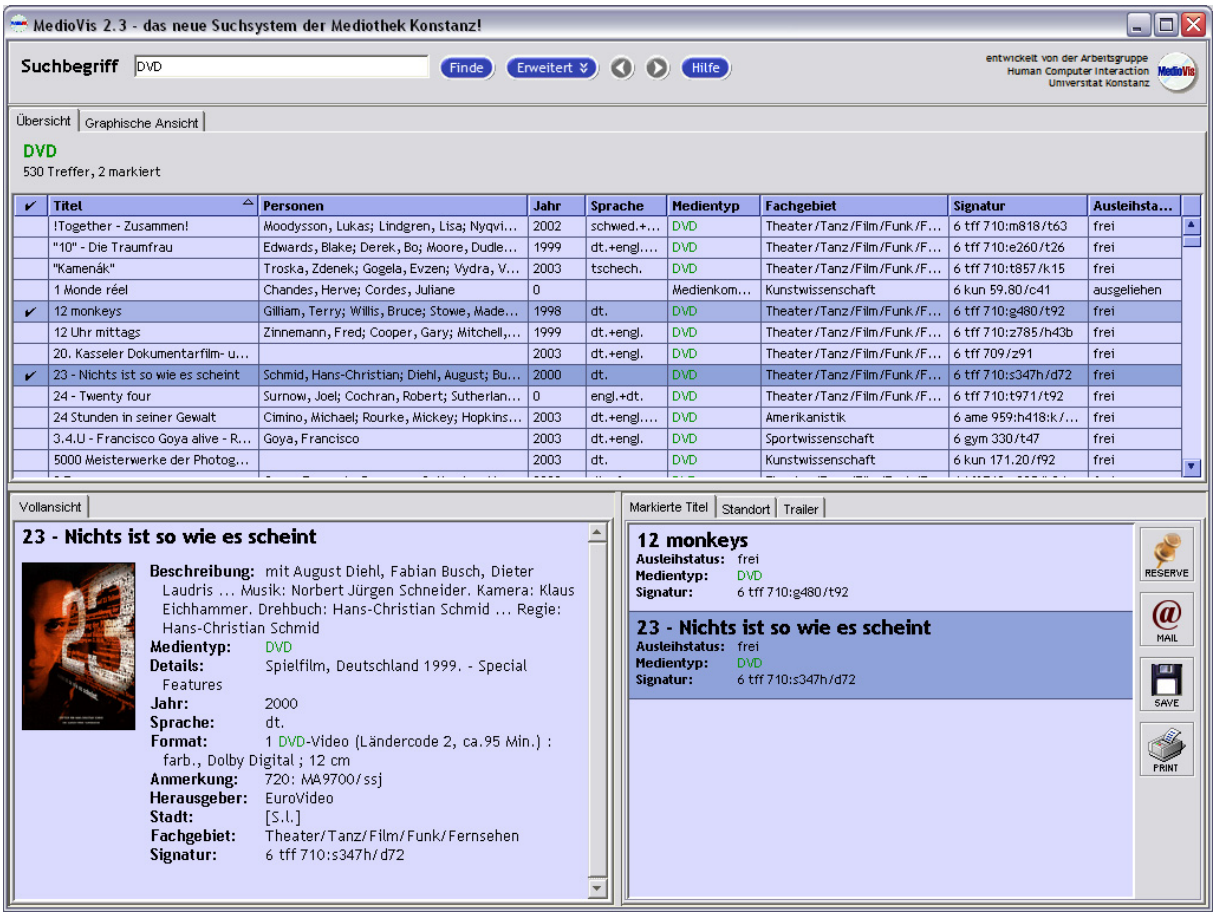

Abbildung 1: MedioVis Standardansicht

Parallel zur Tabellenansicht hat der Benutzer die Möglichkeit eine Graphische Ansicht zu wählen, wie in Abbildung 2 zu sehen ist. In dieser Graphischen Ansicht werden die Treffer anhand von zwei vom Benutzer wählbaren Attributen auf einem zweidimensionalen Scatterplot angeordnet. Dadurch können Zusammenhänge zwischen Titeln besser erkannt werden [Grün2004]. Ebenso wie in der Tabellenansicht können hier Titel durch Überfahren fokussiert werden und Detailinformationen werden im linken unteren Bereich der Anwendung sichtbar. Durch Mausklick lassen sich Titel aus- und abwählen. Diese Ansicht eignet sich vor allem um so genannte Ausreißer zu erkennen. Somit ist diese Art der Visualisierung sowohl für Experten zur Datenkontrolle, als auch für Mediotheksnutzer hilfreich.

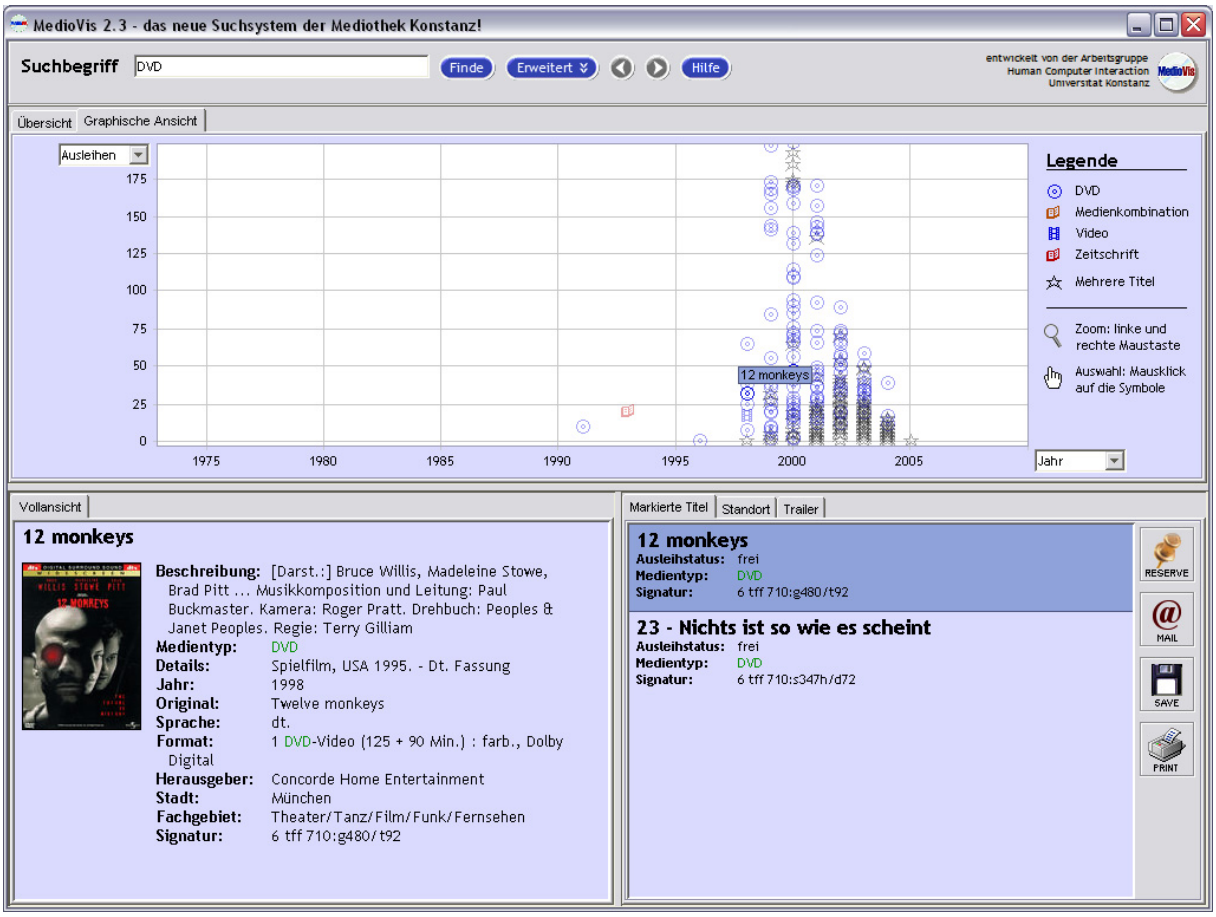

Abbildung 2: MedioVis Graphische Ansicht

Abbildung 3 zeigt eine neue innovative Visualisierung, die für MEDIOVIS entstanden ist. Das Hypergrid [Jetter2005] erlaubt es dem Benutzer, neben den in der Mediothek vorhandenen Metadaten über die Titel noch weitere externe Daten, wie zum Beispiel Informationen über Schauspieler von Filmen, sich darstellen zu lassen, ohne dabei den Kontext zu verlieren oder die Ansicht wechseln zu müssen. Des Weiteren werden bei dem Hypergrid die Attribute der Titel nicht mehr in vielen einzelnen Spalten angezeigt, sondern, thematisch geordnet, in wenigen Spalten. Die jeweiligen Spalten können dann, mittels Interaktion mit der rechten und linken Maustaste, durch den Benutzer beliebig vergrößert oder verkleinert werden. Dabei wird dem Benutzer pro linken Mausklick ein Attribut mehr und pro rechten ein Attribut weniger in der jeweiligen Zelle angezeigt. Der Kontext der Tabellenansicht wird dabei nie verlassen und dadurch kann der Benutzer sich Details von Titeln anzeigen lassen ohne dabei den Überblick zu verlieren. Des Weiteren lassen sich die einzelnen Themenspalten anhand des obersten Attributs sortieren, und der Benutzer hat die

Möglichkeit eine einstellbare Spalte mit einem von ihm gewünschten Attribut zu belegen. Nach diesem Attribut kann die Tabelle ebenfalls sortiert werden.

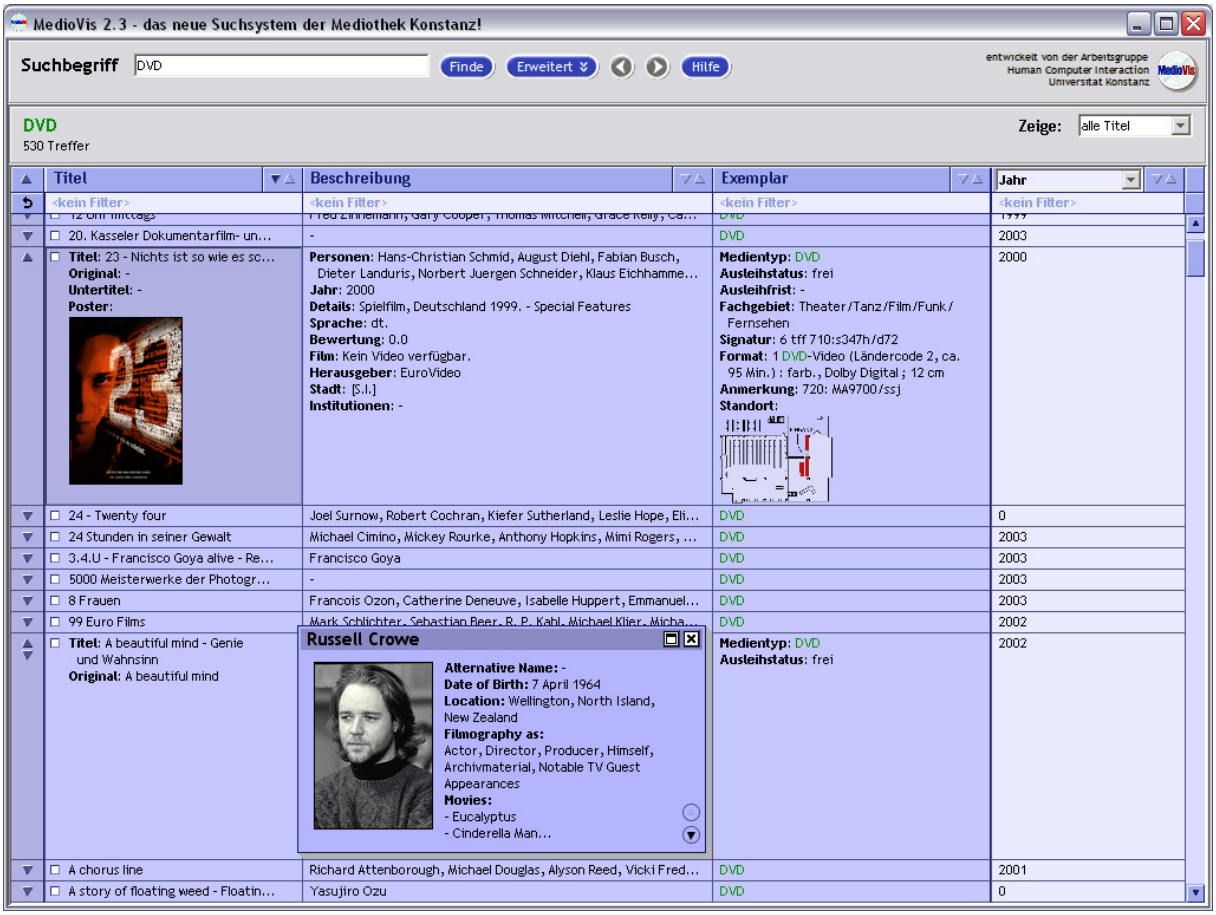

Abbildung 3: MedioVis Hypergrid

Aktuell wird im Projekt MEDIOVIS versucht, dem Hypergrid visuelle Möglichkeiten zur Filterung und Selektierung vorzuschalten um den Benutzern noch bessere Möglichkeiten zur Exploration zu geben, bevor die eigentliche Trefferansicht dem Benutzer geboten wird.

Die Akzeptanz von MEDIOVIS ist bei den Benutzern sehr hoch und ein durchgeführter quantitativer Benutzertest mit 24 Testpersonen, bei dem MEDIOVIS mit der von der Bibliothek Konstanz angebotenen Suchmaske KOALA[2] verglichen wurde, hat gezeigt, dass MEDIOVIS in nahezu allen getesteten Aspekten KOALA überlegen ist

[2] *KOALA* ist die Online-Suchmöglichkeit der Bibliothek Konstanz, http://www.ub.uni-konstanz.de/koala/

[Grün2005]. Dadurch steht für das in dieser Arbeit beschriebene Projekt eine breite Basis an Erfahrung bei der Visualisierung von digitalen Bibliotheken und eine sich im realen Betrieb bewährte Anwendung zur Verfügung.

1.3 Idee von Mobile MedioVis

MOBILE MEDIOVIS ist die konsequente Weiterführung der Idee und der Forschungsthemen, die hinter dem Projekt MEDIOVIS stehen. Mobile Geräte erfreuen sich schon seit geraumer Zeit großer Beliebtheit und bieten, unabhängig davon, wo sich der Benutzer gerade befindet, die Möglichkeit Tätigkeiten, wie zum Beispiel eine Suche in der Mediothek, auszuführen. Mit der neuen Generation von Pocket PCs, die in Kapitel 2 näher beschrieben werden, ist es nun möglich mit VGA-Auflösung[3] auf diesen Geräten zu arbeiten. Außerdem besteht durch die Integration von Bluetooth[4] und Wireless Lan[5] in diese Geräte die Möglichkeit an vielen Orten drahtlos auf aktuelle Daten zugreifen zu können. Mit diesen Möglichkeiten in einem Pocket PC können auch anspruchsvolle Anwendungen umgesetzt werden. Um mit dem begrenzten Bildschirmplatz, der zur Verfügung steht, effektiv umzugehen, stellen sich folgende Fragen:

- Wie unterstützt man den Benutzer visuell bei seiner Suche oder Exploration?
- Wie stellt man eine große Ergebnismenge auf einem kleinen Bildschirm dar?
- Wie verhindert man, dass überhaupt große Ergebnismengen dargestellt werden müssen?

Neben diesen Fragen, die zum größten Teil ihren Ursprung in der Tatsache des geringen verfügbaren Bildschirmplatzes haben, ist es ebenso wichtig dem Benutzer ein Gefühl zu vermitteln, wie groß sein gerade durchsuchter Datenbestand ist. Dies ist sinnvoll, um den Benutzer effektives Suchen anzutrainieren. All dies sind Aspekte,

---

[3] *VGA*, Abkürzung für Video Graphics Array. Standard mit einer Auflösung von 640x480 Bildpunkten

[4] *Bluetooth* ist ein Standard für die drahtlose Übermittlung von Sprache und Daten.

[5] *Wireless Lan* (WLAN) ist ein kabelloses Netzwerk, auch Funknetzwerk genannt.

die hinter der Idee von MOBILE MEDIOVIS stehen und die zu folgenden Punkten führen, die im Projekt MOBILE MEDIOVIS umgesetzt werden:

- Portierung der Java Anwendung MEDIOVIS auf ein mobiles Gerät.
- Visuelle Unterstützung der Suche und Exploration auf Geräten mit kleinem Bildschirm.
- Effektive Darstellung der Ergebnismenge auf einem kleinen Bildschirm.

Um effizient und effektiv auf einem Pocket PC zu entwickeln und um eine große Abdeckung der sich im Umlauf befindlichen Geräte zu haben, war der erste Schritt die Auswahl der richtigen Implementierungsplattform. Bei der Auswahl spielte die Portierung des in MEDIOVIS bestehenden Datenmodells eine wichtige Rolle. In Kapitel 2.3 wird die Auswahl der Implementierungsplattform und das Ergebnis beschrieben. Kapitel 4.1 beschäftigt sich mit der Portierung des Datenmodells.
Um den Benutzer bei der Exploration und Suche in der Mediothek visuell bestmöglich zu unterstützen, wird eine visuelle Repräsentation der Ergebnismenge benötigt, die zum einen dem Benutzer erlaubt mit ihr zu interagieren und zum anderen gleichzeitig eine Gesamtübersicht über den durchsuchten Informationsraum (in diesem Projekt die Mediothek) bietet. Auf das Konzept und den Aufbau der vorgeschlagenen und umgesetzten Visualisierung wird in Kapitel 3.2 näher eingegangen.
Die effektive Darstellung der Ergebnismenge auf kleinen Bildschirmen ist vor allem von dem Standpunkt aus interessant, dass verhindert werden soll viele Treffer darstellen zu müssen. So soll die Visualisierung verhindern, dass der Benutzer in einer großen Ergebnismenge mühsam nach dem Gewünschten suchen muss. Vielmehr soll dem Benutzer ein visuelles Instrument gegeben werden, mit dem es möglich ist seine Ergebnismenge vor der Detailpräsentation soweit zu filtern, dass nur die tatsächlich Erwünschten erscheinen. Gleichzeitig soll die Visualisierung der Ergebnisse ebenfalls effektiv mit Bildschirmplatz umgehen um dem Benutzer einen Vergleich mehrerer Titel der Mediothek bequem zu ermöglichen. Auf die Umsetzung dieses Punktes wird in Kapitel 3.2.6 näher eingegangen.

2 Technische Rahmenbedingungen

Nach der Idee folgt die Umsetzung. Vor der Umsetzung müssen die technischen Rahmenbedingungen erörtert und ausgewertet werden. Dies ist bei Pocket PCs nicht einfach. Sicherlich sollte sich ein Konzept eines neuen visuellen Suchsystems, das gut auf kleinen Displays umgesetzt werden kann, nicht von implementierungstechnischen Details beeinflussen lassen. Dennoch ist es auf dieser Art von Plattform[6] wichtig sich vor dem Beginn eines Konzeptes mit den technischen Rahmenbedingungen vertraut zu machen. Ohne einen Überblick über die Besonderheiten in Bezug auf Interaktion mit dem Gerät, Hardwareeigenschaften und Softwarerestriktionen zu haben ist es schwer, effektive und effiziente Visualisierungen für Pocket PCs zu entwickeln. Zu Beginn soll ein Pocket PC vorgestellt und auf die technischen Besonderheiten dieses mobilen Gerätes eingegangen werden.

2.1 Was ist ein Pocket PC?

In der Tat definiert nicht jeder den Begriff Pocket PC gleich. Die meisten werden wohl unter einem Pocket PC ein PDA[7]-ähnliches Gerät erwarten. Viele können sich darunter nichts Konkretes vorstellen. Pocket PC bezeichnet sowohl ein Computer-Betriebssystem als auch eine Reihe von persönlichen digitalen Assistenten (PDAs), die mit dem Betriebssystem Pocket PC von Microsoft betrieben werden [Wikipedia2005]. In Abbildung 4 sind einige aktuelle Geräte mit der neueren Microsoft Pocket PC Version, mittlerweile in Microsoft Windows Mobile umbenannt, abgebildet. Ein Pocket PC wird hier definiert als ein mobiles Gerät mit dem aktuellen Betriebssystem Windows Mobile 2003 SE von Microsoft, das speziell auf mobile Anwendungen zugeschnitten ist. Im Folgenden wird auf die Besonderheiten bei Pocket PCs und auf die Unterschiede zu herkömmlichen Desktop PCs[8] eingegangen.

---

[6] Eine *Plattform* bezeichnet eine Kombination von Betriebssystem und einem bestimmten Computertyp.

[7] *PDA*, Personal Digital Assistant, Kleinstcomputer um Termine und Adressen zu verwalten.

[8] *Desktop PC*, Personal Computer für den Schreibtisch.

Abbildung 4: Aktuelle Pocket PCs

2.2 Besonderheiten bei Pocket PCs

Was unterscheidet einen Pocket PC von einem Desktop PC? Diese Frage scheint nicht schwer zu beantworten zu sein. Ein Pocket PC ist kleiner, leichter, verfügt über keine Maus und Tastatur, hat nur beschränkte Ressourcen. Diese Liste könnte man noch um einige, sehr schnell auffallende Merkmale, bei denen sich ein Pocket PC und ein Desktop PC unterscheiden, weiterführen. Einige Unterschiede, welche für die Konzeption und Entwicklung eines visuellen Suchsystems auf einem Pocket PC relevant sind, erfährt man erst bei genauerer Betrachtung.

2.2.1 Hardware

Hier wird auf die Unterschiede in der Hardware zwischen Pocket PCs und Desktop PCs, die bezüglich der Entwicklung eines visuellen Suchsystems relevant sind, eingegangen. Ein bereits mehrfach erwähnter, wichtiger Unterschied ist die verfügbare Bildschirmgröße, exakter ausgedrückt die Bildschirmauflösung. Diese liegt typischerweise bei der neuesten Generation der Pocket PCs bei 640 x 480 Pixel und bei zurzeit aktuellen Desktop PCs bei mindestens 1024 x 768 Pixel. Um den Unterschied zu verdeutlichen folgende kleine Rechnung: Bildschirmauflösung Pocket

PC = 640 * 480 = 307.200 Pixel; Bildschirmauflösung Desktop PC >= 1024 * 768 >= 786.432 Pixel. Dies bedeutet, dass man auf den zurzeit aktuellen Bildschirmen von Desktop PCs mindestens 2,5-mal so viele Pixel für ein visuelles Suchsystem zur Verfügung hat wie auf einem Pocket PC. Die Prozessorgeschwindigkeit muss ebenfalls berücksichtigt werden. Während sich bei aktuellen Desktop PCs die Taktraten jenseits der 3 GHz-Marke bewegen, müssen sich aktuelle Pocket PCs mit Taktraten, die sich typischerweise in der Region 400 bis 620 MHz bewegen, begnügen. Dies muss bei der Umsetzung eines visuellen Suchsystems, gerade in Bezug auf Animationen und Speicherung von Daten, berücksichtigt werden. Einen nicht zu vernachlässigenden Einfluss auf die Interaktion mit dem visuellen Suchsystem, der in Kapitel 5.2 erklärt wird, haben die verschiedenen Eingabegeräte der beiden Geräteklassen. Desktop PCs bedienen sich hier im Regelfall der Maus und Tastatur zur Interaktion. Bei Pocket PCs findet die komplette Interaktion mittels eines Stylus[9] statt. Mit dem Stylus kann auf dem drucksensitiven Bildschirm durch Klicken ein dargestelltes Objekt ausgewählt werden, oder durch längeres Drücken des Objektes ein, sofern für dieses Objekt vorhanden, Kontextmenü aufgerufen werden, über das definierte Operationen mit dem Objekt ausgeführt werden können. Texteingaben können bei einem Pocket PC über das Soft Input Panel (SIP) durchgeführt werden. Hierbei gibt es mehrere Möglichkeiten die Texteingabe durchzuführen. Zum einen ist es möglich, Texteingaben direkt mit dem Stylus auf das Display in Schreibschrift zu schreiben, und das SIP übersetzt das Geschriebene mittels einer Schrifterkennung in Text, andererseits kann aber auch eine Tastatur am unteren Rand des Displays eingeblendet werden, über die der Benutzer mittels des Stylus die Texteingabe durchführen kann. Des Weiteren verfügt jeder Pocket PC über eine bestimmte Anzahl an Knöpfen, die an dem Gehäuse direkt angebracht sind. Da sie aber bei den verschiedenen Geräten auf unterschiedliche Weise angeordnet sind, kann es für den Benutzer verwirrend sein, wenn diese Knöpfe in einer Anwendung für eine bestimmte Tätigkeit definiert sind. Selbstverständlich ist bei Desktop PCs und auch bei Pocket PCs eine Vielzahl anderer Eingabearten möglich und auch durchaus denkbar für die Interaktion mit der Anwendung bzw. dem

[9] Als *Stylus* bezeichnet man Stifte, die zur Bedienung von Touchscreens an Computern, Pocket PCs, Graphic Tablets usw. verwendet werden.

Gerät. So gibt es zum Beispiel externe Tastaturen für Pocket PCs, die über Verbindungskabel oder drahtlos über Bluetooth mit dem Gerät verbunden werden können. Allerdings ist dann der Vorteil der Mobilität eines Pocket PCs nicht mehr gegeben, und diese Art von Eingabegeräten ist für dieses Projekt nicht nötig, wie in Kapitel 3.2 gezeigt wird.
In Tabelle 1 sind die wichtigsten Unterschiede bezüglich der Hardware noch einmal zusammengefasst.

Hardware	Pocket PC	Desktop PC
Bildschirm Auflösung Größe	TFT[10] ~640 x 480 Pixel Bis zu 4“	TFT >= 1024 x 768 >= 17“
Prozessor	Bis zu 620 MHz	Über 3 GHz
Eingabegeräte	Stylus auf drucksensitivem Bildschirm	Maus, Tastatur

Tabelle 1: Unterschiede Hardware Pocket PC und Desktop PC

Neben den erwähnten Unterschieden gibt es auch einige Gemeinsamkeiten zwischen aktuellen Pocket und Desktop PCs. Durch die Verfügbarkeit von Bluetooth und Wireless LAN in Pocket PCs ist es möglich auf ein bestehendes Netz mit Verbindung zum Internet zuzugreifen, und somit die volle Konnektivität, die auch bei einem Desktop PC mit aktiver Verbindung zum Internet vorhanden ist, zu erreichen. Dies ist gerade in Bezug auf die Datenmenge und die Aktualität von Informationen bei einem visuellen Suchsystem für Mediotheksdaten von Bedeutung.

2.2.2 Software

Sowohl bei Pocket als auch bei Desktop PCs ist es jeweils das aktuelle Betriebssystem von Microsoft, welches am stärksten am Markt vertreten ist. Neben

[10] *TFT*, Abkürzung für "Thin Film Transistor" - kurz für Flachbildschirme.

den vorgestellten Unterschieden in der Hardware gibt es auch nennenswerte Unterschiede bezüglich der Software. Bevor auf diese Unterschiede zwischen der Desktop PC Version, in diesem Falle Windows XP, und der Pocket PC Version, hier Windows Mobile 2003 SE, eingegangen wird, zunächst eine Erklärung bzw. Definition von Microsofts Betriebssystem Pocket PC.

Microsoft Pocket PC basiert auf dem Betriebssystemkern Windows CE, der speziell für die Verwendung in Klein- und Kleinstcomputern entwickelt wurde. Windows CE stellt also die Basis für das Betriebssystem Pocket PC dar, ist dem aber nicht gleichzusetzen. Windows CE kann auf unterschiedlichsten Plattformen mit verschiedensten Features laufen. Ein Entwickler nimmt hierzu den Microsoft Platform-Builder und stellt sein individuelles Windows CE zusammen: Mit oder ohne grafische Oberfläche, Kommandozeile, Bluetooth-Unterstützung usw. Die für die Entwicklung von Anwendungen für die verschiedenen Windows CE-Plattformen benötigten Werkzeuge und SDKs[11] stellt Microsoft kostenlos zur Verfügung. Microsoft Pocket PC, die aktuelle Version wird auch als Windows Mobile 2003 SE bezeichnet, erweitert die Funktionalität von Windows CE um typische Anwendungen für Taschencomputer wie Terminkalender oder Adressverwaltung. Die Benutzeroberfläche orientiert sich dabei an derjenigen von Microsoft Windows, ist allerdings speziell für die Verwendung auf Pocket PCs angepasst worden [Wikipedia2005].

Diese erwähnten Anpassungen der Benutzeroberfläche lassen sich am besten anhand einiger Bildschirmfotos zeigen. Abbildung 5 zeigt in (a) den Startbildschirm von Windows Mobile 2003 SE, der sich dem Benutzer präsentiert, wenn das Gerät eingeschaltet wird. Der Aufbau weist deutliche Parallelen zu Microsoft Windows auf, welches ebenfalls, zwar typischerweise am unteren Bildschirmrand, über ein Startmenü verfügt. Gleichzeitig werden, ebenso wie bei Windows, die wichtigsten Statusinformationen über aktive Verbindungen sowie Lautstärkeeinstellung und Uhrzeit eingeblendet. Abbildung 5 (b) zeigt das aufgeklappte Startmenü, welches über einen Klick mit dem Stylus auf das Startsymbol aufgerufen wird. Über das Startmenü können alle wichtigen Funktionen bzw. Bereiche des Betriebssystems

[11] Ein Software Development Kit, kurz *SDK*, ist eine Sammlung von Programmen und Dokumentation zu einer bestimmten Software.

erreicht werden. Was man bei einem Pocket PC im Startmenü vermisst, ist die Möglichkeit den Rechner herunterzufahren. Dies ist nicht vorgesehen, da sich der Pocket PC nach einer einstellbaren Zeit automatisch in einen Stand-By-Modus begibt.

(a) (b) (c)

Abbildung 5: Pocket PC Benutzeroberfläche

Abbildung 5 (c) zeigt die geöffnete Anwendung Internet Explorer für Pocket PC, in der eine Webseite aufgerufen wurde. Gleichzeitig ist das Soft Input Panel sichtbar, welches benötigt wird, um Texteingaben wie zum Beispiel eine URL einzugeben. Hier wird ein Problem bei Pocket PCs sichtbar: Der begrenzt verfügbare Bildschirmplatz. Eine weitere Besonderheit bei der Benutzeroberfläche von Pocket PCs ist, dass nicht mit Fenstern gearbeitet wird. Alle Anwendungen laufen im Vollbildmodus. Dadurch ergeben sich weitere Einschränkungen und Vorgaben, die bei einer Visualisierung beachtet werden müssen. Weitere Besonderheiten werden, wenn benötigt, in Kapitel 3.2 beim Konzept der Visualisierung und in Kapitel 4.2, in dem die Umsetzung der Visualisierung beschrieben wird, aufgezeigt.

2.3 Auswahl der Implementierungsplattform

Im Rahmen des Projektes MEDIOVIS ist eine voll funktionale Anwendung in Java entstanden, die durch die Plattformunabhängigkeit von Java keine Voraussetzungen an das Betriebssystem stellt außer eine laufende Java Virtual Machine (JVM). Die

JVM ist eine Laufzeitumgebung und virtuelle Maschine für Software. Die JVM führt den Java-Bytecode aus, der bei der Kompilierung von Java-Quellcode entsteht. Jeder PC, der über ein JVM, unabhängig vom Betriebssystem, in der benötigen Version verfügt, kann diesen Bytecode ausführen. Das gleiche sollte in Bezug auf wenige Anforderungen an das Betriebssystem auch für das Projekt MOBILE MEDIOVIS gelten. Des Weiteren sind, wie bereits in Kapitel 2.2.1 beschrieben, die von der jeweiligen Implementierungsplattform benötigten Ressourcen ein nicht zu vernachlässigender Punkt, der mit in die Auswahl einfließt. Begleitend zu ersten Ideen und Mock-ups[12] für die Visualisierung, die zu Beginn des Kapitels 3 vorgestellt werden, war es nötig eine Aufstellung und einen Vergleich der am meisten verbreiteten Implementierungsplattformen für die Entwicklung auf Pocket PCs zu machen. Die Auswahl stand vor allem unter dem Gesichtspunkt, dass MOBILE MEDIOVIS nicht nur ein funktionaler Prototyp bleiben sollte, sondern vielmehr eine unter realen Bedingungen laufende Anwendung, die über nahezu dieselben Funktionalitäten wie die Java Anwendung MEDIOVIS verfügt. Im Folgenden werden vier bekannte Vertreter im Detail vorgestellt und dabei jeweils auf die Voraussetzungen sowohl im Hardware- als auch im Softwarebereich, die verfügbaren Entwicklungsumgebungen und auf die Vor- und Nachteile eingegangen. Abschließend wird die getroffene Auswahl begründet.

2.3.1 Java

Der Ausdruck Plattformunabhängigkeit, welcher bei Desktop PCs die große Stärke von Java ist, endet bei Pocket PCs sehr schnell. Dies hat mehrere Gründe. Der Hauptgrund ist sicherlich die fehlende Unterstützung Microsofts, dem Hersteller des meist verwendeten Betriebssystems auf Pocket PCs: Windows Mobile 2003 SE.
Die Micro Edition der Java 2 Plattform von Sun ist dafür ausgelegt auf mobilen Geräten zu entwickeln [J2ME]. Die mobilen Geräte sind in Klassen eingeteilt und verschiedene Konfigurationen und Profile definiert, um die verschiedenen

[12] Ein *Mock-up* in der Softwareentwicklung bezeichnet einen rudimentären Wegwerfprototyp der Benutzeroberfläche einer zu erstellenden Software.

Hardwareressourcen bestmöglich zu unterstützen und auszunutzen. Abbildung 6 zeigt die von Sun vorgenommene Einteilung der Konfigurationen und Profile.

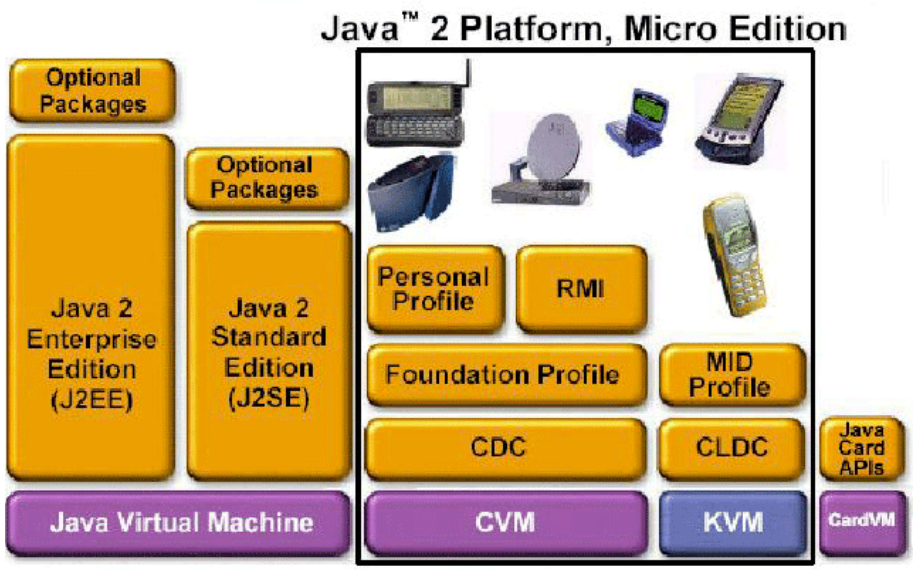

Abbildung 6: Java 2 Platform, Micro Edition [J2ME]

Mit der Connected Device Configuration (CDC) ist es möglich für High-end PDAs zu entwickeln. Vorteil hier ist, dass viele Teile der bestehenden Java Anwendung aus dem Projekt MEDIOVIS ohne großen Aufwand direkt übernommen werden könnten. Die Realität spricht eine andere Sprache. So gibt es, wie bereits erwähnt, keine Unterstützung auf Pocket PCs und es existiert derzeit auch keine frei verfügbare Java Virtual Machine, die auf Pocket PCs sinnvoll eingesetzt werden könnte [JavaFAQ]. Es existieren einige kommerzielle Produkte. Das bekannteste ist die Everywhere Micro Enviroment[13] von IBM (früher J9 genannt). Allerdings muss für jedes Gerät eine Lizenz erworben werden, und auch die Entwicklungsumgebung ist nicht frei verfügbar. Es gibt noch einige weitere Produkte, wie zum Beispiel die Jeode JVM oder das Produkt Waba, die hier nicht näher ausgeführt werden, da auch mit diesen Produkten die aufgeführten Nachteile vorhanden sind. Ein weiteres Kriterium, welches gegen die Implementierungsplattform Java spricht, ist die fehlende Unterstützung der Oberflächenbibliothek Swing. Bei Swing handelt es sich um eine

[13] Everywhere Micro Enviroment, URL: http://www-306.ibm.com/software/wireless/weme/

von Sun Microsystems entwickelte Grafikbibliothek, die in Java geschrieben wurde. Ohne die Unterstützung dieser Grafikbibliothek ist es nahezu unmöglich effizient ein visuelles Suchsystem in Java umzusetzen.

2.3.2 Macromedia Flash

Macromedia Flash ist sowohl ein beliebtes Instrument, um einen Webauftritt mit einem so genannten Eye-Catcher[14] zu versehen, als auch ein Tool, welches sich in vielen Bereichen zu einem oft benutzten Instrument für Prototyping etabliert hat. Flash besteht aus einer lizenzpflichtigen Entwicklungsumgebung, welche direkt von Macromedia vertrieben wird, und einem Flash Player, welcher kostenlos für nahezu jede Plattform verfügbar ist. Die Entwicklungsumgebung zeichnet sich vor allem dadurch aus, dass man in einem grafischen Editor sehr schnell Benutzeroberflächen und Animationen erstellen kann. Diese können mit einer Scriptsprache, genannt ActionScript, die sich am ECMA[15] Script 2 Standard orientiert, mit Funktionen versehen werden. Mit Flash ist es möglich vollwertige Anwendungen zu schreiben, oder einfach prototypisch Funktionen oder Designvorschläge zu generieren. Um Flash Anwendungen auf einem Pocket PC zu betreiben, wird der Macromedia Flash Player für Pocket PC 2003 benötigt. Technisch gesehen gibt es bei Pocket PCs hier Einschränkungen. So kann der Flash Player für Pocket PC nur Flash Filme innerhalb des Internet Explorers abspielen. Dies hat zur Folge, dass Anwendungen nicht im Vollbildmodus betrieben werden können. Es gibt kostenpflichtige Versionen direkt von Macromedia als auch von Drittherstellern, die hier Abhilfe schaffen können. Für das Projekt MOBILE MEDIOVIS gibt es hier zwei entscheidende Nachteile. Bei Verwendung von Macromedia Flash wäre es notwendig, das komplette Datenmodell neu zu implementieren. Das Projekt ZuiScat [Büring2005], ebenfalls in der Arbeitsgruppe Mensch Computer Interaktion entstanden, hat zudem gezeigt, dass es

[14] Ein *Eye-Catcher* (Deutsch: Blickfang) ist meist ein grafisches Element welches die Aufmerksamkeit des Betrachters auf sich ziehen soll.

[15] Die *European Computer Manufacturers Association (ECMA)* ist eine private Normungsorganisation und wurde 1961 gegründet, um Computersysteme in Europa zu standardisieren.

einige Schwierigkeiten bezüglich der Performanz von Flash auf Pocket PCs gibt. In diesem Projekt wurde mittels Flash ein auf einem Scatterplot basierendes Konzept entwickelt um große Informationsräume zu durchsuchen. Dabei wurde ein Prototyp dieses Suchsystems auf einem Pocket PC mit Macromedia Flash umgesetzt. Auf dieses Projekt wird in Kapitel 3.1.3 noch einmal zurückgekommen.

2.3.3 Embedded C++

Bei Embedded C++[16] handelt es sich um ein Projekt, das im Jahr 1996 auf der Embedded Systems Conference in San Jose vorgestellt wurde. In dem Projekt geht es darum, die mächtige Sprache C++ standardisiert für mobile Geräte verfügbar zu machen. Bei EC++ handelt es sich um eine Teilmenge von C++. Um die Komplexität aus der Sprache zu nehmen und nur das bereitzustellen, was auch wirklich von mobilen Geräten benötigt wird, wurden einige Konstrukte ausgelassen. So wurden Bereiche wie Mehrfachvererbung und Exception Handling aus dem Sprachumfang herausgenommen, um sicherzustellen nur diejenigen Sprachanteile zu benutzen, welche alle C++ Compiler unterstützen. Viele professionelle und effiziente Anwendungen für Pocket PCs entstehen mit Embedded C++. Wenn mit Embedded C++ für Pocket PCs entwickelt wird, ist folgendes zu beachten. Benötigt wird hierfür eine spezielle Entwicklungsumgebung, wie zum Beispiel eMbedded Visual C++ 4.0 von Microsoft. Mit Embedded C++ wird nativer Code direkt für das Gerät erzeugt, d.h. es ist keine Virtual Machine wie bei Java oder ein Player wie bei Macromedia Flash notwendig um die Anwendung auszuführen. Embedded C++ wird und sollte immer dann verwendet werden, wenn systemnah programmiert wird und Anwendungen sehr effizient ausführbar sein müssen [DevTools]. Für MOBILE MEDIOVIS hat diese Implementierungsplattform deswegen Nachteile, weil, wie bei Flash, große Teile der bestehenden Software MEDIOVIS nicht übernommen werden können.

[16] *Embedded C++*, Projektseite: http://caravan.net/ec2plus

2.3.4 Microsoft .NET Compact Framework

Das .NET Framework von Microsoft wird seit einigen Jahren immer populärer. Hier wird, ähnlich wie bei Java, eine Trennung von ausführbarem Code und Laufzeitumgebung vorgenommen. Hierbei übernimmt die *Common Language Runtime* (CLR) im Wesentlichen die Aufgabe, die durch die Virtual Machine bei Java übernommen wird. Durch diese Trennung erreicht Microsoft Plattformunabhängigkeit, allerdings nur in der Form, dass die von Microsoft entwickelten verschiedenen Plattformen unterstützt werden. Seit einiger Zeit existiert das Project Mono[17]. Mono ist ein Open-Source-Projekt mit dem Ziel, eine Reihe von .NET-kompatiblen Entwicklungs-Werkzeugen zu schaffen, einschließlich eines C#-Compilers und einer Common Language Runtime, für den Betrieb unter Windows, Linux, verschiedenen UNIX-Derivaten sowie Mac OS X.

Im .NET Framework wird, im Gegensatz zu Embedded C++, so genannter *managed code* erzeugt, der dynamisch kompiliert und nachgeladen wird. *Managed code* wird typischerweise für Anwendungen benutzt, bei denen es in erster Linie um die Benutzerschnittstelle geht und die schnell entwickelt werden sollen [DevTools]. Das .NET Framework unterstützt dabei über 20 verschiedene Sprachen, die durch die CLR unterstützt werden. Die CLR spielt im .NET Framework eine ähnliche Rolle wie die Virtual Machine bei Java Anwendungen. Ein schematischer Aufbau des .NET Framework ist in Abbildung 7 zu sehen.

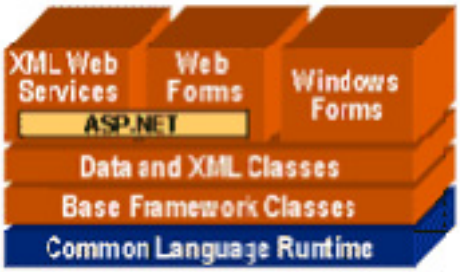

Abbildung 7: Schematischer Aufbau .NET Framework

Das .NET Compact Framework ist eine Teilmenge vom .NET Framework. Es besteht aus den Basisklassen und hat einige zusätzliche Bibliotheken, um die spezifischen

[17] *Mono Projekt*, URL: http://www.mono-project.com

Funktionen solcher Geräte ausnutzen zu können. In Abbildung 8 ist die Klassenarchitektur des .NET Compact Framework dargestellt. Das .NET Compact Framework ist komplett in die Entwicklungsumgebung Visual Studio .NET integriert, welche für den akademischen Bereich kostenfrei bezogen werden kann. Ein Nachteil des .NET Compact Framework ist, dass die Plattformunabhängigkeit, welche durch die Java Anwendung MEDIOVIS gegeben war, verloren geht. Allerdings ist aufgrund des großen Markanteils von Pocket PCs, die mit dem Betriebssystem von Microsoft ausgestattet sind, dieser Faktor eher zu vernachlässigen. Vorteile hingegen ergeben sich hier durch die Tatsache, dass mit einem Tool, welches von Microsoft kostenfrei zur Verfügung gestellt wird und im Kapitel 4.1 näher beschrieben wird, große Teile der bestehenden Anwendung MEDIOVIS von Java in die Programmiersprache C# in das .NET Compact Framework übernommen werden können.

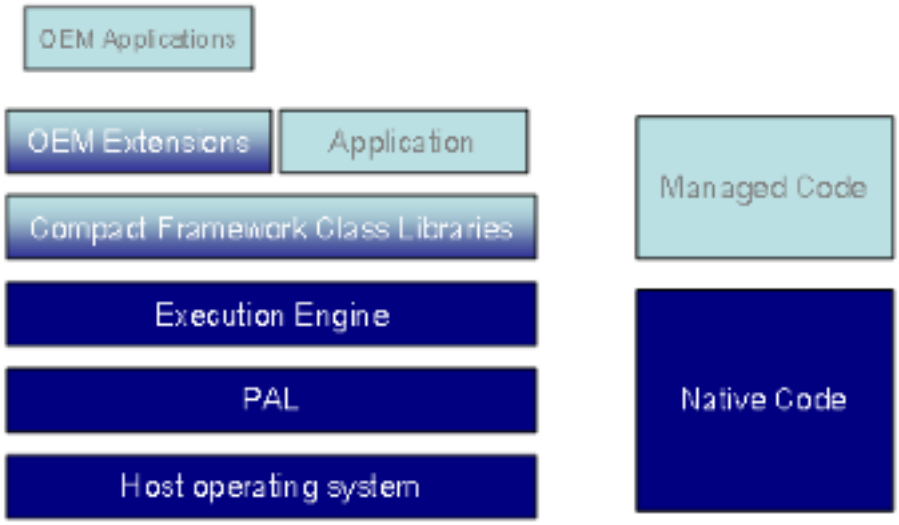

Abbildung 8: .NET Compact Framework Klassenarchitektur

2.3.5 Vergleich und Entscheidung

Nachdem die vier Vertreter vorgestellt wurden, soll kurz eine Zusammenfassung gegeben und anschließend die Auswahl begründet werden. Unter der bereits erwähnten Voraussetzung, dass die Implementierungsplattform wenig Anforderungen an den Pocket PC stellen sollte, stellte sich heraus, dass sich am besten das .NET Compact Framework für die Umsetzung des Projektes MOBILE

MEDIOVIS eignet. Dies ist folgendermaßen zu begründen. Bedingt durch die Tatsache, dass die Plattform für die Umsetzung ein Pocket PC mit dem Betriebssystem Windows Mobile 2003 ist, können Java als auch Flash nur über kostenpflichtige Wege sinnvoll zum Einsatz kommen. Des Weiteren bietet Java, egal mit welchem Produkt, keine Unterstützung für die Swing Bibliothek, welche zwangsläufig benötigt wird um ein effizientes visuelles Suchsystem zu implementieren. Flash und Embedded C++ scheiden aufgrund der Tatsache aus, dass hier jeweils große Teile der vorhandenen Klassen von MEDIOVIS nicht übernommen werden können. Somit fiel die Wahl auf das .NET Compact Framework von Microsoft, nicht zuletzt deswegen, weil eine nahezu perfekte Unterstützung der ausgewählten Plattform gegeben war.

3 Visualisierung

Nachdem im letzten Kapitel ausführlich auf technische Aspekte eingegangen wurde, die für das visuelle Suchsystem auf einem Pocket PC wichtig sind, soll auf den eigentlichen Kern dieser Arbeit, dem Konzept für das visuelle Suchsystem, in diesem Kapitel eingegangen werden. Selbstverständlich sind für die Ideen und Konzepte eines visuellen Suchsystems auf einem Pocket PC nicht nur die technischen Rahmenbedingungen oder die Auswahl der Implementierungsplattform entscheidend. Es bedarf auch einer zielgerichteten Analyse, wie ein Pocket PC typischerweise benutzt wird, und wo der Vorteil liegt ein visuelles Suchsystem auf einem Pocket PC umzusetzen. Teile dieser Analyse liegen aber in technischen Details, da auf Pocket PCs neben Hard- und Softwarerestriktionen auch Interaktionstechniken mit anderen Eingabegeräten zum Einsatz kommen. So wird auch in diesem Kapitel wenn nötig auf technische Details eingegangen. Um in das Konzept des visuellen Suchsystems, welches in dem Projekt MOBILE MEDIOVIS entwickelt wurde, einzuführen, werden zunächst einige bestehende interessante Visualisierungen vorgestellt. Anschließend wird das Konzept des visuellen Suchsystems, welches im Rahmen des Projektes MOBILE MEDIOVIS entstanden ist, im Detail erläutert.

3.1 Bestehende Visualisierungen für kleine Bildschirme

Ein guter Start um mit einem Konzept für eine innovative Visualisierung zu beginnen, ist, dass man sich zunächst einen Überblick über bereits bestehende Visualisierungen, welche im selben Problemfeld liegen, verschafft und dabei die Vor- und Nachteile für das eigene Problem herausarbeitet. Sinn und Zweck ist es dabei nicht, die beste passende Visualisierung zu finden und diese zu kopieren, sondern vielmehr aus bereits bestehenden Forschungsergebnissen neue Ideen zu entwickeln. Durch Kombinationen oder Anpassung bestehender Visualisierungen mit anderen, neuen oder bereits vorhandenen Interaktionstechniken können effektive visuelle Suchsysteme entstehen.

3.1.1 Starfield Visualisierung

Bei der *Starfield* Visualisierung [Ahlberg1993] handelt es sich um einen zweidimensionalen Scatterplot, welcher mit einer Zooming Funktionalität versehen wurde, damit eine Überladung des Bildschirms verhindert wird. Bevor ich ein die *Starfield* Visualisierung benutzendes visuelles Suchsystem, den Filmfinder [Ahlberg1994], vorstelle, will ich kurz auf den Scatterplot eingehen. Der Scatterplot kommt ursprünglich aus der Statistik. Dabei handelt es sich um eine Visualisierung, bei der die Objekte in einem zweidimensionalen Koordinatensystem anhand von zwei Attributen angeordnet werden. Ins Deutsche übersetzt bedeutet Scatterplot Punktwolke, was aufgrund des visuellen Ergebnisses ein gelungener Name dieser Visualisierung ist. Die zwei Achsen werden entweder direkt, wie beim FilmFinder, vom Entwickler festgelegt, oder die Benutzer können selbst die Attribute festlegen, nach welchen die Objekte im Scatterplot angeordnet werden. Vorteile eines Scatterplots sind die visuellen Vergleichsmöglichkeiten von Objekten anhand von zwei Attributen und das Erkennen von Beziehungen zwischen Objekten. Der FilmFinder [Ahlberg1994] bedient sich der Scatterplot Visualisierung und erweitert diese um eine Zooming Funktionalität. In Abbildung 9 sieht man die Ausgangsituation des FilmFinders. Die Filme sind auf dem Scatterplot ausgerichtet. Dabei wird auf der horizontalen Achse die Jahreszahl aufsteigend als Attribut und auf der vertikalen Achse eine Filmbewertung von 0 bis 9 aufsteigend verwendet. Die Filme werden anhand dieser zwei Attribute angeordnet. Zusätzlich wird über die Farben noch ein drittes Attribut, die Kategorie des jeweiligen Filmes, kodiert. Beim FilmFinder werden zwei Interaktionsparadigmen benutzt, die für die Interaktion auf einem Pocket PC interessant sind. Der Benutzer hat die Möglichkeit über Regler Titel, Schauspieler und Direktor einzustellen. Des Weiteren können über Knöpfe die Kategorie und über einen Bereichsregler der Jahresbereich eingestellt werden. Verschiedene Arten der Filmbewertung können zusätzlich über Kontrollboxen ausgewählt werden. Abbildung 10 zeigt die Anwendung, nachdem Einstellungen von einzelnen Optionen verändert wurden, und dadurch in den Jahreszahl-Bewertungs-Raum hineingezoomt wurde.

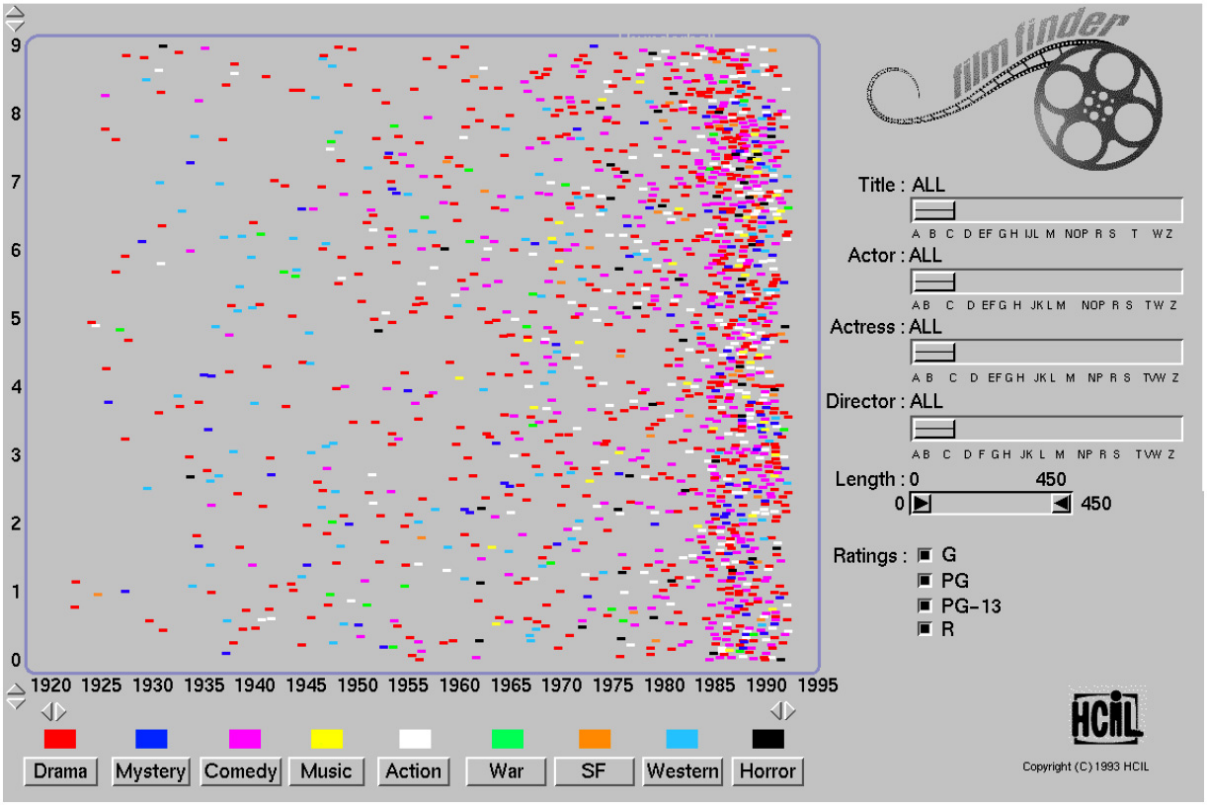

Abbildung 9: FilmFinder, Startansicht

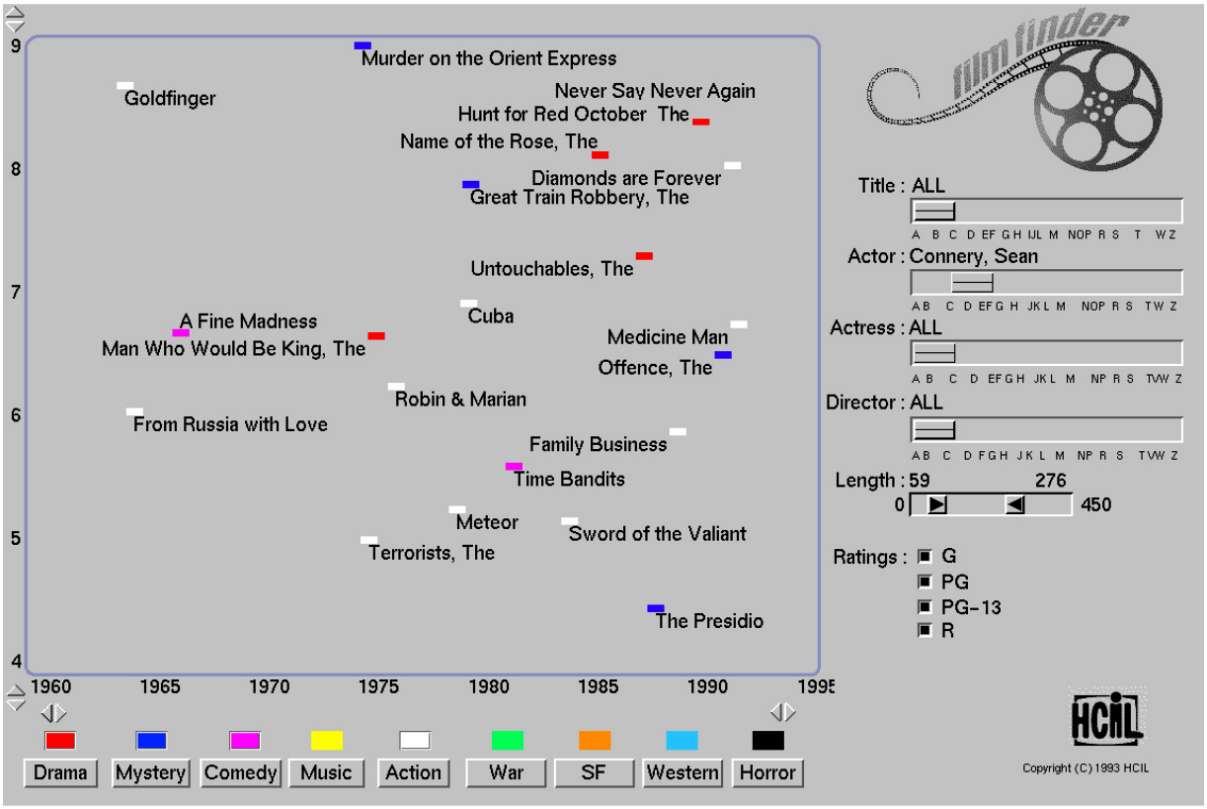

Abbildung 10: FilmFinder, Selektion und Zooming durchgeführt

Die Paradigmen, die beim FilmFinder verwendet und kombiniert werden, sind *Direct Manipulation* und *Dynamic Queries*.

Bei *Direct Manipulation* handelt es sich, wie der Name schon aussagt, um direkte Manipulation der Oberfläche [Frohlich1993]. Die Idee ist dabei, dass der Benutzer Aktionen mit Objekten, wie beispielsweise eine Datei löschen, nicht über Kommandos erledigen muss, sondern die Datei, die er löschen will, einfach in den Papierkorb zieht. Demnach ist jedes, heutzutage aktuelle, Betriebssystem bis zum einem gewissen Grad auf diesem Paradigma aufgebaut. Eine Bedienoberfläche, welche *Direct Manipulation* unterstützt, sollte folgende Eigenschaften besitzen [Shneiderman1997]:

- Durchgehende Darstellung von Objekten und Aktionen, die benötigt werden.
- Physische Aktionen oder Benutzung von Knöpfen zur Aktionsausführung, anstatt komplexer Kommandos.
- Schnelle, wieder umkehrbare Operationen, von denen die Effekte auf die Objekte sofort sichtbar sind.

Mit Hilfe der direkten Manipulation können Systeme umgesetzt werden, die folgende Vorteile haben [Shneiderman1997]:

- Unerfahrene Benutzer können Basisfunktionalitäten schnell erlernen. Normalerweise durch Erklärungen von erfahrenen Benutzern.
- Fehlermeldungen werden selten benötigt.
- Benutzer können sofort sehen, ob ihre Aktionen das von ihnen gewünschte Ergebnis erzielen, und wenn nicht, können sie schnell die Änderungen wieder rückgängig machen.
- Benutzer haben keine Angst Fehler zu machen, weil das System verständlich ist und Aktionen schnell rückgängig gemacht werden können.
- Benutzer haben das Gefühl die Anwendung im Griff zu haben, weil sie die Aktionen selbst ausführen und die volle Kontrolle über das System haben.

Das zweite Paradigma, *Dynamic Queries* [Ahlberg1992], zeichnet sich dadurch aus, dass komplexe Filterungen der Objekte oder Suchanfragen dynamisch erstellt werden. Dies bedeutet am Beispiel des FilmFinders, dass die Kombination der Einstellungen jedes Reglers oder Knopfes die jeweilige Filterung bestimmt. Filter werden nicht über Begriffe oder Zahlen definiert, sondern, dynamisch und jederzeit änderbar, über direkte Manipulation bestimmt. Somit kann der Benutzer, ohne dass er Detailinformationen über die Objekte haben muss, komplexe Suchanfragen definieren.

Sowohl *Dynamic Queries* als auch *Direct Manipulation* können durch Kombination miteinander zu einem mächtigen visuellen Suchsystem werden, wenn gleichzeitig die visuelle Präsentation der Objekte nah mit den Funktionen verbunden ist. Ahlberg und Shneiderman [Ahlberg1993] gehen bei ihrer *Starfield* Visualisierung und beim FilmFinder nicht auf Vorteile ein, welche ein solches, visuelles Suchsystem auf kleinen Bildschirmen hat. Dies liegt vielleicht daran, dass die Idee aus dem Jahre 1993 stammt. Dennoch ist es zum einen mit einem Scatterplot möglich viele Objekte auf kleinen Raum zu repräsentieren, und zum anderen sind die verwendeten Paradigmen *Direct Manipulation* und *Dynamic Queries* mächtige Werkzeuge um effektive visuelle Suchsysteme für kleine Bildschirme zu entwickeln. Beide Paradigmen spielen für das Konzept des im Rahmen dieses Projektes entwickelten visuellen Suchsystems, welches in Kapitel 3.2 beschrieben wird, eine große Rolle.

3.1.2 Mobile Liquid 2D Scatter Space

Eine sehr interessante Informationsvisualisierung, welche für mobile Geräte entwickelt wurde [Waldeck2003], stellt der Mobile Liquid 2D Scatter Space (ML2DSS) dar. Das Konzept beruht auf der *Starfield* Visualisierung [Ahlberg1994]. Der Autor Carsten Waldeck geht dabei auf die Nachteile eines 2D Scatterplots bezüglich überlappender Objekte ein. Solche Überlappungen kommen zustande, wenn mehrere Objekte die gleichen Attributwerte auf beiden Achsen haben. Gleichzeitig liegt, laut Waldeck, ein großer Vorteil dieser 2D Scatterplot Darstellung in der stufenlosen Skalierbarkeit der Darstellungsfläche, was eine sehr einfache Anpassung an verschiedenste Bildschirmgrößen ermöglicht. Um mit dem Problem

der Überlappungen umgehen zu können und Klarheit und Lesbarkeit auch bei sehr großen Informationsdichten zu gewährleisten, wurde versucht, Deckkraft und Rollover-Funktionen möglichst effektiv einzusetzen, sowie eine Entzerrungsfunktionalität hinzugefügt, die ein sehr intuitives Wühlen mit dem Look-and-Feel[18] einer öligen Flüssigkeit in dem Informationsraum ermöglicht. Diese Funktionalität wird als *Liquid Browsing* beschrieben. In Abbildung 11 wird das Problem und die Lösung grafisch dargestellt. (a) zeigt das Problem der Überlappung einzelner Elemente. In (b) wird gezeigt, wie die Lösung mittels der Fisheye-Technik aussehen würde. Bei der Fisheye-View [Furnas1986] werden die Objekte selbst beim Überfahren vergrößert. Das Problem der Überlappung besteht allerdings weiterhin. *Liquid Browsing*, mit dem das Problem gelöst wird, wird in (c) veranschaulicht. Dabei werden nicht die Objekte, sondern der Raum zwischen den Objekten vergrößert, so dass man alle Objekte visuell erfassen kann. Der Raum wird je nach Druckstärke, die auf das Display ausgeübt wird, angepasst. Der Name *Liquid Browsing* kommt durch die Tatsache zustande, dass die Verschiebung der Objekte so aussieht, als ob sie in einer Flüssigkeit schwimmen würden.

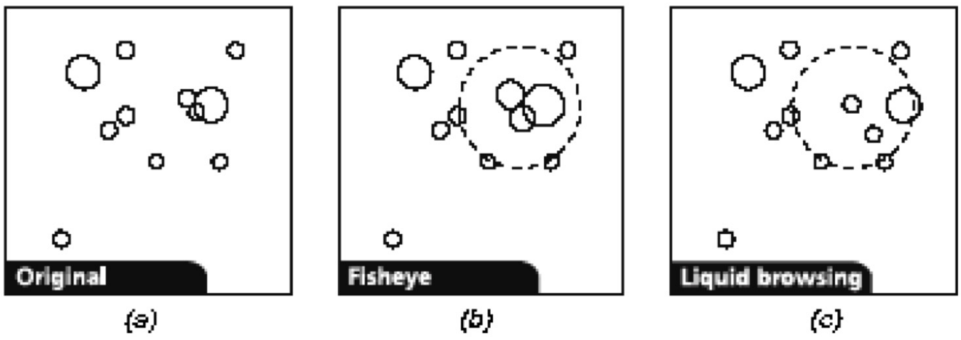

Abbildung 11: ML2DSS, Liquid Browsing

Während man bei einem Desktop PC als Eingabegerät für direkte Manipulation typischerweise eine Maus benutzt, wird bei einem Pocket PC ausschließlich der Stylus als direktes Eingabegerät verwendet. Diese Tatsache wird hier in zweierlei Hinsicht ausgenutzt. Wenn man mit dem Stylus auf einen Bereich, in dem sich viele

[18] Mit *Look and Feel* werden, meist durch Hersteller oder Konsortien standardisierte Design-Aspekte, wie zum Beispiel Farben, Layout, Schriftgröße usw., in Software bezeichnet.

Objekte befinden, drückt, wird je nach Druckstärke bzw. Verweildauer mit dem Stylus der Raum zwischen den Objekten bis zu einer definierten Obergrenze vergrößert. Eine Filtermöglichkeit wird beim ML2DSS durch einfaches Zeichnen eines Rechteckes innerhalb des Starfield Displays gegeben. Wenn das Rechteck gezeichnet wurde, erscheint ein Kontextmenü, aus dem man die Filterung auswählen kann. Die Achsen können in diesem 2D Scatter Space beliebig gewählt werden. Abbildung 12 zeigt einen Vergleich, in dem der Vorteil vom ML2DSS sichtbar wird. Während in einer Tabellenansicht nur ca. 15 Objekte dargestellt werden können, sind bei gleicher Displaygröße mit dem ML2DSS 250 Objekte möglich.

Abbildung 12: ML2DSS, Vergleich von Tabellenansicht und Starfield Display

Es entstehen auch Nachteile bei dieser Art der Visualisierung. Durch die Vergrößerung der Räume zwischen den Objekten stimmen die Werte der auf den Achsen angegebenen Attribute nicht mehr mit der Position des Objektes überein. Dadurch wird Inkonsistenz erreicht, die den Benutzer verwirren könnte. Weiterhin stellt sich die Frage, ob Filterung durch Selektion eines Bereiches ausreicht um befriedigende Anfragen erstellen zu können. Komplexe Anfragen sind, wenn nur diese Funktion verwendet wird, kaum möglich.

3.1.3 ZuiScat

Der ZuiScat [Büring2005] bedient sich ebenso wie der Mobile Liquid 2D Scatter Space der Idee eines Scatterplots. Er ist ebenso wie das Projekt MOBILE MEDIOVIS in der Arbeitsgruppe Mensch Computer Interaktion entstanden. Der ZuiScat wurde als Prototyp mit Macromedia Flash umgesetzt. Wie in Kapitel 2.3.2 bereits erwähnt, traten hierbei Performanzprobleme auf.

Beim ZuiScat wird semantischer Zoom verwendet um zu ermöglichen, dass sowohl Trefferübersicht als auch Detailinformationen zu einem Titel ohne Wechsel der Ansicht möglich ist. Der Benutzer gibt zu Beginn einen Suchbegriff ein und wird dann zur Trefferansicht weitergeleitet (Abbildung 13 links). In der Trefferansicht werden die Treffer anhand von zwei wählbaren Attributen in dem zweidimensionalen Scatterplot angeordnet. Anschließend besteht die Möglichkeit mit dem Stylus einen Navigationspunkt zu setzen und mit den Hardwareknöpfen (Hoch und Runter) des Pocket PCs in dem zoombaren Scatterplot einzutauchen (Abbildung 13 rechts).

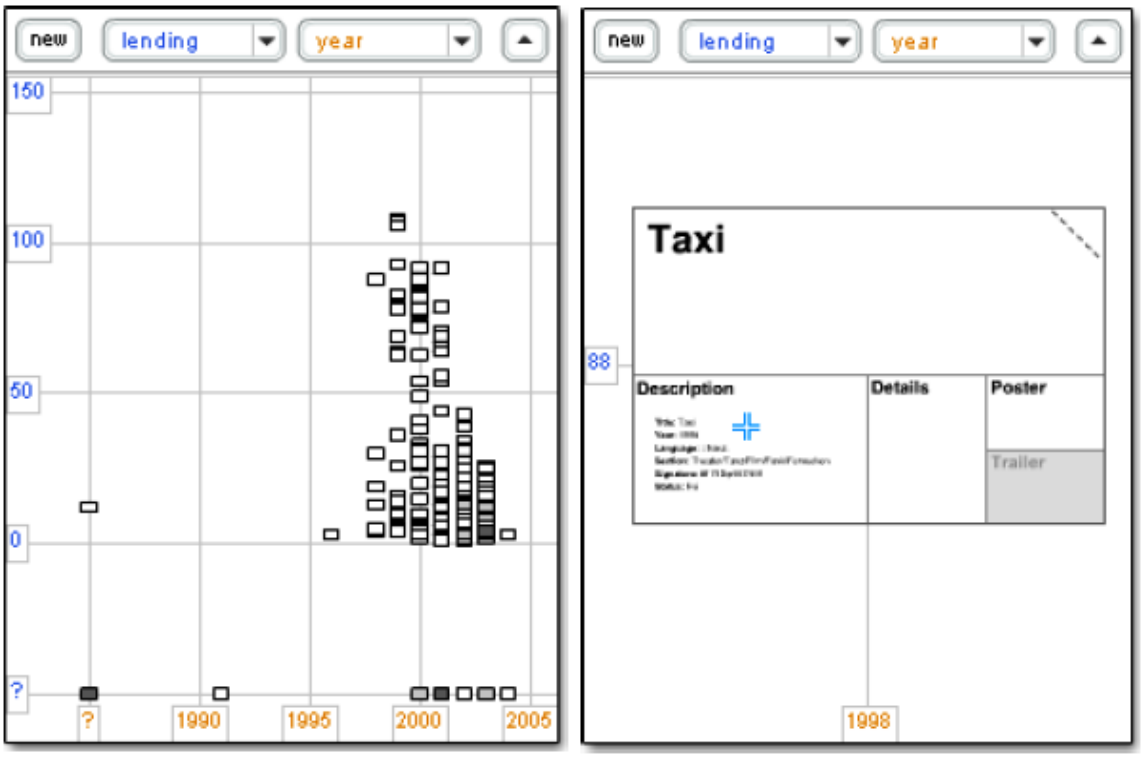

Abbildung 13: ZuiScat, Darstellung aller Ergebnisse und Einzeldarstellung

Durch den semantischen Zoom wird je nach Skalierung mehr oder weniger Information zu einem Titel angezeigt. Der Benutzer kann jederzeit wieder mit den

Hardwareknöpfen herauszoomen und durch das Setzen eines neuen Navigationspunktes einen anderen Titel des ZuiScat im Detail betrachten. Das Problem der Überlappung von mehreren Objekten wird hierbei durch den Zoom gelöst. Wenn mehrere Objekte an derselben Stelle auf dem Scatterplot liegen, wird beim Hineinzoomen sichtbar, dass mehrere Objekte vorhanden sind (Abbildung 14 links). Der Benutzer kann sich durch Setzen eines neuen Navigationspunktes und Zoomen mit den Hardwareknöpfen mehr Details von dem gewünschten Objekt anzeigen lassen. Die Objekte sind ähnlich wie eine Karteikarte aufgebaut. Wenn der Benutzer einen Eintrag markieren will, kann er dies durch Umknicken der rechten oberen Ecke der Karteikarte tun (Abbildung 14 rechts).

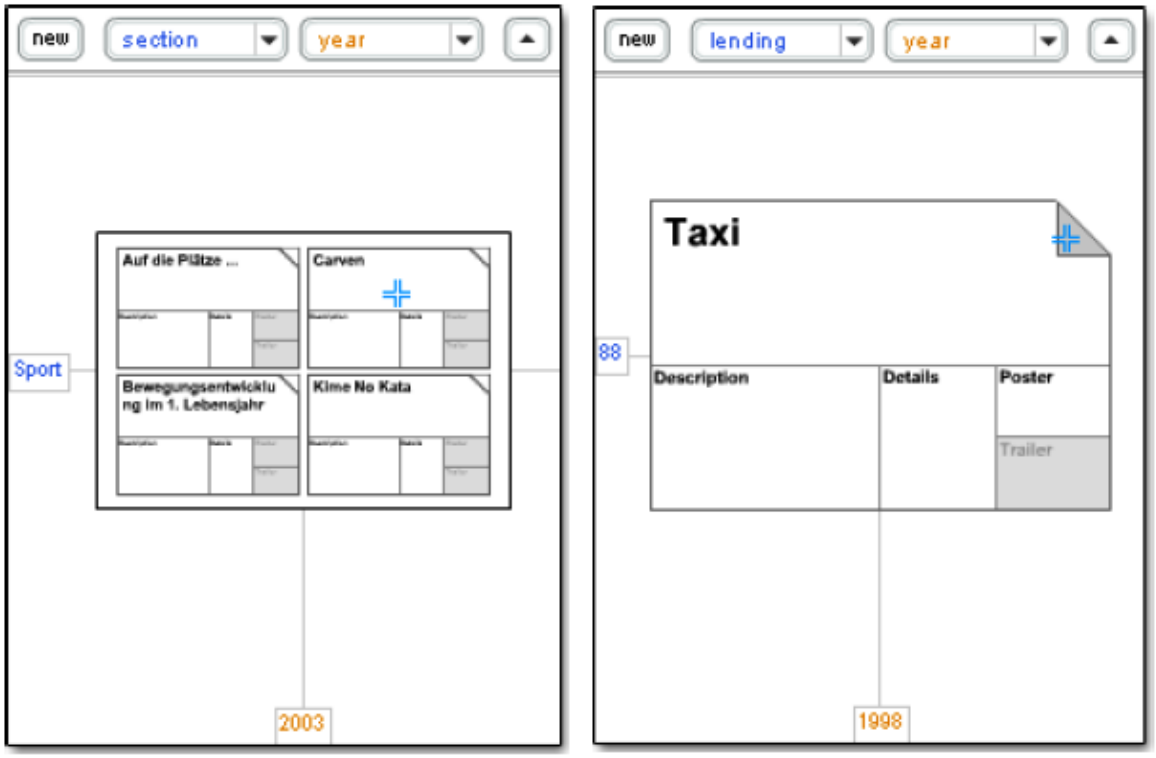

Abbildung 14: ZuiScat, Überlappung von Objekten und Selektion

Ebenso wie die Trefferanzeige können die ausgewählten Objekte oder die Suchhistorie auch im ZuiScat angezeigt werden. Dadurch wird nur eine Ansicht benötigt um alle Funktionen eines visuellen Suchsystems zu erfüllen.

Der ZuiScat kann effektiv viele Treffer auf kleinen Bildschirmen darstellen und durch den semantischen Zoom können Probleme wie zum Beispiel Überlappungen von Objekten vollkommen verhindert werden. Ein Benutzertest hat gezeigt, dass zum

Beispiel die Verwendung der Hardwareknöpfe für Verwirrung sorgte. Des Weiteren wurde die Metapher der Karteikarte bei der Selektion nicht verstanden.
Wegen der beschriebenen Performanzprobleme, die ebenfalls zu Verwirrung der Benutzer beitrug, wird der ZuiScat entweder in C# oder Embedded C++ neu implementiert werden.

3.1.4 DateLens

Die *Starfield* Visualisierung und der Mobile Liquid 2D Scatter Space bieten keine Möglichkeit, sich Detailinformationen von einem Titel anzeigen zu lassen und gleichzeitig im Kontext einer Titelübersicht zu bleiben.
Eine Visualisierung, welche aus einer anderen Anwendungsdomäne kommt, hat sich mit dieser Problematik beschäftigt. Die DateLens [Bederson2003] wurde konzipiert um die Verwaltung und Analyse von Terminen auf kleinen Bildschirmen zu unterstützen. Abbildung 15 veranschaulicht die Funktionsweise der DateLens. Auf der linken Seite wird die Übersicht über einen Zeitraum von mehreren Wochen gezeigt. Dabei werden die Termine in der Übersicht durch blaue Zeilen angedeutet. Rechts ist dargestellt, was sich dem Benutzer präsentiert, wenn er einem bestimmten Tag, durch Klicken mit dem Stylus, auswählt. Der ausgewählte Tag erhält genug Platz um die Termine des Tages darzustellen. So entsteht ein Hineinzoomen in den Tag, mittels semantischen Zoom. Semantischer Zoom bedeutet, dass, abhängig vom Skalierungsfaktor des Objektes, unterschiedliche Repräsentationen des Objektes erscheinen [Perlin1993]. Die Veränderung der Zellengrößen wird dabei jeweils animiert, so dass der Benutzer die Aktion nachvollziehen kann. Mittels dieser Visualisierung ist es möglich, sich einzelne Objekte detailliert zu betrachten ohne den Überblick zu verlieren. Über den Regler an der rechten Seite lässt sich die Größe der einzelnen Zellen und somit die Anzahl der Wochen bzw. Monate einstellen. Vorteile dieser Visualisierung sind sicherlich das schnelle Erkennen und Finden von Terminen sowie der effektive Umgang mit der Bildschirmgröße. Des Weiteren kann die DateLens sowohl auf großen wie auch auf kleinen Bildschirmen eingesetzt werden. Somit kann eine Visualisierung auf allen Geräten unabhängig von der Bildschirmgröße eingesetzt werden, was im Bereich der Terminplanung, aber auch in den meisten anderen Bereichen, ein großer Vorteil ist. Benutzertests

[Bederson 2004] haben allerdings gezeigt, dass bei einfachen Aufgabenstellungen die Standard-Kalenderanwendung bei Pocket PCs der DateLens bezüglich der durchschnittlichen Bearbeitungszeit der Aufgabe überlegen ist. Bei komplexeren Aufgaben schnitt die DateLens besser ab. Außerdem berichteten einige Benutzer, dass sie das Gefühl hatten die DateLens nicht im Griff zu haben.
Trotzdem ergeben sich für die Ergebnispräsentation von Objekten, wegen der Idee des semantischen Zooms, interessante Aspekte und Ideen für MOBILE MEDIOVIS.

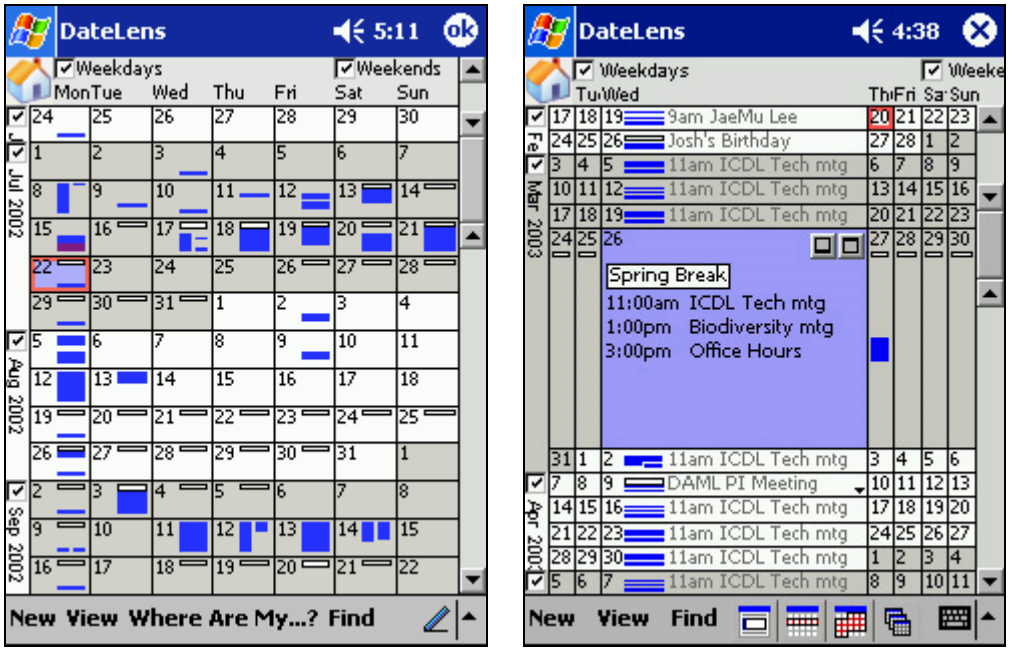

Abbildung 15: DateLens, Übersicht und Detailansicht

3.1.5 TreeMaps

Eine TreeMap ist eine visuelle Repräsentation, die für die Darstellung von komplexen Baumstrukturen entwickelt wurde [Shneiderman1991]. Typischerweise wird eine Baumstruktur durch einen mit einer Wurzel versehenen, gerichteten Graphen ausgedrückt. Dieser Graph wird grafisch so aufbereitet, dass der Wurzelknoten ganz oben angeordnet ist, und alle weiteren Kinderknoten sind unterhalb des Wurzelknoten dargestellt. Linien verbinden die Knoten um die Hierarchie auszudrücken. Diese Repräsentation eignet sich nur bedingt, um zum einen effizient

mit Bildschirmplatz umgehen zu können, und zum anderen effektiv große Baumstrukturen übersichtlich darstellen zu können.
In Abbildung 16 ist eine Baumstruktur auf die oben beschriebene Art dargestellt. Die Zahlen in den jeweiligen Blättern drücken dabei die Größe des jeweiligen Objektes aus.

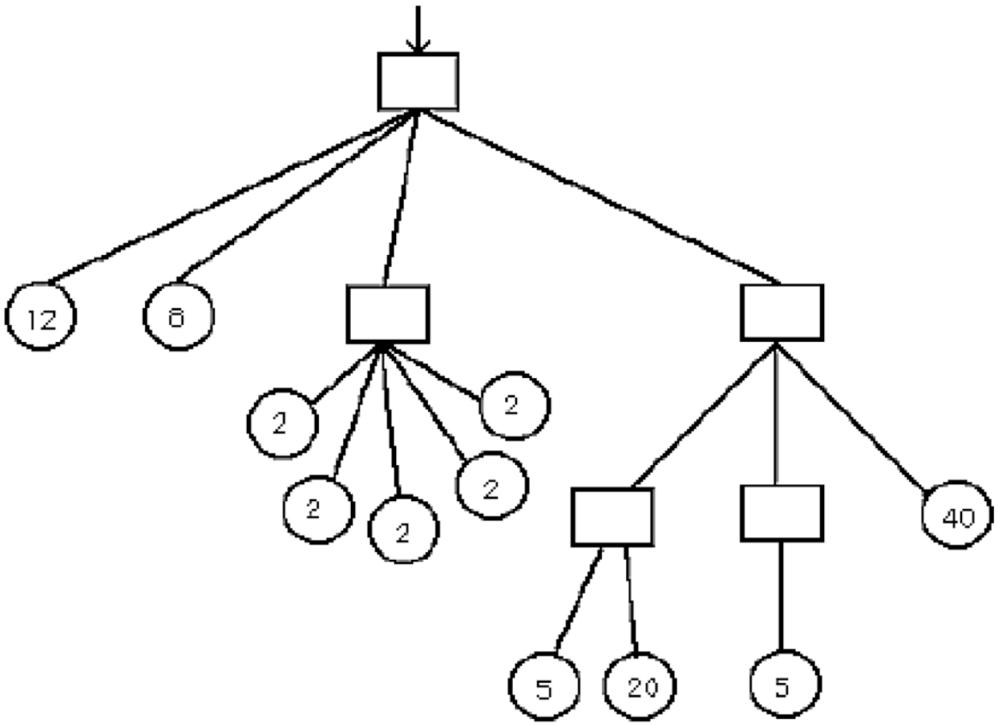

Abbildung 16: Baumstruktur der TreeMap

Der TreeMap-Algorithmus teilt den zur Verfügung stehenden Bildschirmplatz anhand der vorliegenden Baumstruktur auf. Dabei beginnt der Algorithmus mit dem Wurzelknoten aus Abbildung 16 und definiert zunächst ein Rechteck, dass durch die obere linke und untere rechte Ecke des zur Verfügung stehenden Bildschirmplatzes bestimmt ist (siehe Punkte P1(x1, y1) und Q1(x2, y2) in Abbildung 17). Die Anzahl der ausgehenden Kinderknoten vom Wurzelknoten bestimmt dabei die Anzahl der Unterteilungen auf der X-Achse. Wenn die Größe des am weitesten links liegenden Kinderknoten nicht leer ist, dann wird auf der horizontalen Achse die erste Trennlinie bei x3 = x1 + (Größe von Kinderknoten / Gesamtgröße) * (x2-x1) gezeichnet. Der Algorithmus geht dann rekursiv an diesem Kinderknoten weiter, und benutzt dabei das um 90 Grad rotierte Viereck, welches durch die Koordinaten P2(x3, y1) und Q2(x1, y2) definiert ist. Bei vorhandenen Knoten wird dann das Rechteck vertikal

geteilt. Gleichzeitig geht der Algorithmus weiter nach rechts im Baum und teilt das übrig gebliebene Rechteck P2(x3, y1), Q1(x2, y2) nach der gleichen Vorgehensweise ein (Abbildung 17). Somit teilen Knoten mit gerader Tiefe das Rechteck vertikal und Knoten mit ungerader Tiefe horizontal (Wurzelknoten hat Tiefe 0).

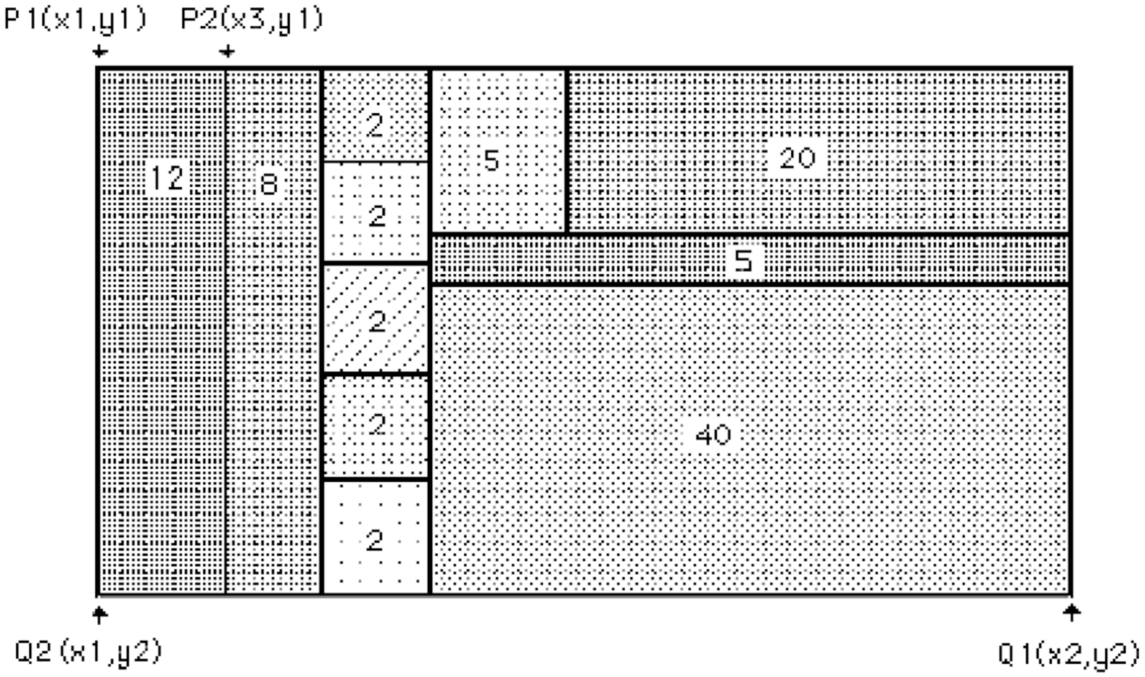

Abbildung 17: TreeMap

Um die entstandenen Rechtecke visuell voneinander abzugrenzen, müssen verschieden Farben oder Schattierungen verwendet werden. Ursprünglich wurde die TreeMap entwickelt, um eine bessere Repräsentation der Speicherplatzverwendung bei Betriebssystemen zu erhalten. Dabei werden die Dateien als Blätter, und Verzeichnisse, die in Unterverzeichnissen eingebettet sind, als Knoten des Baumes interpretiert. Die Farbkodierung kann hier über verschiedene Dateitypen, Alter der jeweiligen Datei oder die Häufigkeit der Benutzung erfolgen. Bei der Darstellung von 2000 Dateien oder anderen Objekten stehen bei dieser Visualisierung bei einer Auflösung von 640x480 Pixel pro Objekt über 150 Pixel für jede Datei bzw. jedes Objekt zur Verfügung. So ist die TreeMap eine Visualisierung, die auch auf kleinen Bildschirmen große komplexe Bäume darstellen kann.

3.2 Konzept des visuellen Suchsystems

Nachdem die Visualisierungen, die in das Konzept des visuellen Suchsystems mit eingeflossen sind, betrachtet und erklärt wurden, wird dieses im Detail hier vorgestellt. Zunächst wird das Anwendungsszenario, welches dem Konzept zugrunde liegt, dargestellt. Daraufhin werden Ergebnisse von Brainstorming-Sitzungen, die in internen Besprechungen der Arbeitsgruppe Mensch-Computer Interaktion stattfanden, präsentiert. Diese Ergebnisse bilden die Grundlage für die beiden Visualisierungskomponenten Pocket SieveMap und Mobile Media Grid, die das visuelle Suchsystem darstellen.

3.2.1 Anwendungsszenario

MOBILE MEDIOVIS ist die konsequente Weiterentwicklung von MEDIOVIS für mobile Geräte. Wie bereits in der Einleitung erwähnt, besteht das Ziel des Projektes MEDIOVIS in der Realisierung einer visuellen Benutzungsschnittstelle zur freien sowie zielgerichteten Exploration in der Mediothek. Dasselbe soll, mit kleinen Einschränkungen, auch für MOBILE MEDIOVIS gelten. Der Benutzer soll die Möglichkeit haben, sowohl gezielt nach Titeln in der Mediothek suchen, als auch sich, durch Exploration, einen Überblick über die Mediotheksdaten verschaffen zu können. Dabei spielt der Faktor Mobilität eine entscheidende Rolle. So soll die Suche unabhängig vom Standort des Benutzers erfolgen können. Der Benutzer kann auf einem Pocket PC mit begrenzter Bildschirmgröße die Mediothek durchsuchen, und bekommt gleichzeitig ein Gefühl für den durchsuchten Informationsraum.

3.2.2 Brainstorming

Mit diesem Anwendungsszenario als Vorgabe wurden in der Arbeitsgruppe mehrere Brainstorming-Sitzungen abgehalten. Dabei wurden Ideen entwickelt, Vor- und Nachteile diskutiert und über mögliche Umsetzungen nachgedacht. Ergebnisse dieser Sitzungen, die auf dem Weg zum Konzept entstanden sind, werden nun vorgestellt.

Aufgrund der Tatsache, dass dieses visuelle Suchsystem für einen in der Größe beschränkten Bildschirm entworfen werden soll, wurden folgende Anforderungen definiert:

- Übersichtliche Präsentation des gesamten Informationsraumes, um dem Benutzer ein Gefühl für die Größe zu geben.
- Effektive visuelle Einschränkungsmöglichkeit des Informationsraumes, um den Benutzer schnell zum Ziel kommen zu lassen.
- Effektive Darstellung der Suchergebnisse, die sowohl Übersicht bietet als auch auf Verlangen Detailinformationen zu den Titeln bereitstellt, um dem Benutzer eine geeignete Vergleichsmöglichkeit von mehreren Titeln zu bieten.

Um diese Anforderungen zu erfüllen, wurden erste Ideen entwickelt und als Mock-ups umgesetzt um sie zu diskutieren. Beim Diskutieren der Ideen wurde klar, dass eine geleitete Suche für ein solches Anwendungsszenario einige Vorteile mit sich bringt. Bei der geleiteten Suche wird der Benutzer in einen Suchprozess eingebunden, bei dem das System den Benutzer in geeigneter Form unterstützt um schneller ans Ziel zu kommen. Gleichzeitig soll der Benutzer die Möglichkeit haben schnell zur Trefferansicht zu gelangen, wenn er gezielt nach einem bestimmten Titel sucht. Dies hat zur Folge, dass das visuelle Suchsystem aus zwei Komponenten besteht. Die erste Komponente bietet dem Benutzer ein visuelles Filterwerkzeug, mit dem er in einer geleiteten Suche durch direkte Manipulation seine Suchanfrage dynamisch formulieren kann. Mit der zweiten Komponente kann sich der Benutzer Detailinformationen zu einem Titel anzeigen lassen ohne dabei den Kontext verlassen zu müssen. Die beiden entstandenen Visualisierungskomponenten werden nun im Detail vorgestellt.

3.2.3 Datenaufbereitung für die Pocket SieveMap

Der Name Pocket SieveMap, übersetzt in die deutsche Sprache *Taschen-Siebkarte*, hat den Ursprung in der Idee, dass durch die SieveMap der Informationsraum gesiebt werden kann. Durch diesen Siebvorgang können alle überflüssigen

Ergebnisse visuell herausgefiltert werden. Die Bezeichnung Pocket kennzeichnet dabei die Tatsache, dass die SieveMap für Pocket PCs und generell kleine Bildschirmgrößen entwickelt wurde.

Grundvoraussetzung für ein visuelles Suchsystem ist die visuelle Darstellung von Daten. Damit dies geschehen kann, müssen die Daten als erstes in brauchbare interne Strukturen überführt werden. Im zweiten Schritt können dann die Daten in visuelle Strukturen überführt werden, aus denen dann Ansichten generiert werden können. Ein oft zitiertes Modell, welches die Verarbeitung der Daten in interne Strukturen bis hin zur Visualisierung veranschaulicht, ist in Abbildung 18 dargestellt.

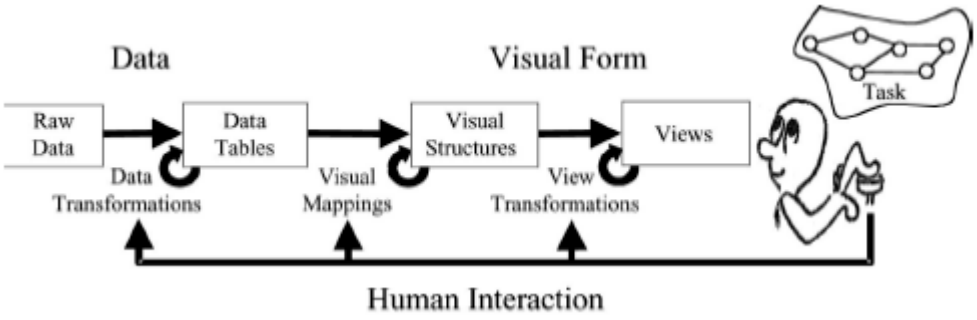

Abbildung 18: Referenzmodell für Visualisierungen (aus [Card1999])

Die Daten der Mediothek, die für die Pocket SieveMap zur Verfügung stehen, befinden sich bereits in einer internen Struktur. Sie sind innerhalb einer Tabelle in einer relationalen Datenbank abgelegt, wobei jede Zeile der Tabelle einen Mediothekstitel darstellt.

Hinter der Pocket SieveMap (PSM), welche hier in den folgenden Kapiteln als visuelles Filterwerkzeug vorgestellt wird, steht die Idee der TreeMaps. TreeMaps haben den Vorteil, dass komplexe Baumstrukturen visuell ansprechend präsentiert werden. Gleichzeitig kann der Benutzer sich einen Überblick verschaffen, welches der Objekte, die in der TreeMap dargestellt werden, besonders groß oder klein ist. Darüber hinaus können Zusammenhänge zwischen Objekten leicht erkannt werden.

Um eine geeignete Baumstruktur für die PSM zu bekommen, war es nötig die vorhandenen Mediotheksdaten zu analysieren um geeignete Kategorien und Cluster für eine geleitete Suche zu finden. Außerdem bot es sich an, das Suchverhalten der Benutzer zu analysieren.

Eine Mediothek beinhaltet typischerweise viele multimediale Titel wie DVDs, CDs, Video, usw. Des Weiteren gibt es viele Titel zu Fachgebieten wie *Fremdsprachen Lernen* oder *Theater, Tanz, Film, Funk und Fernsehen*. Die Mediothek wird typischerweise stark von Medienwissenschaftlern genutzt. Dadurch bekommt das Attribut Sprache ein größeres Gewicht, weil oftmals nach Titeln in einer bestimmten Sprache gesucht wird. Analysen der eingegebenen Suchbegriffe in MEDIOVIS durch DROID[19] haben gezeigt, das ein häufig eingegebener Suchbegriff zum Beispiel der Begriff *DVD* ist. Dies bestätigt die Notwendigkeit der Bildung von Clustern und Kategorien. Durch eine manuelle Datenanalyse wurden geeignete Kategorien und Cluster festgelegt. Bei dieser Analyse wurden die einzelnen Attribute und ihre Verteilung untersucht.

Durch das große Vorkommen von DVD als Suchbegriff und die Tatsache, dass der Hauptbestandteil einer Mediothek multimediale Titel sind, wurde Medientyp als eine Kategorie festgelegt.

Bei genauerer Analyse des Attributes Sektion wurde ersichtlich, dass sich viele Sektionen zu Übersektionen zusammenfassen lassen. Somit wurde Fachgebiet als weitere Kategorie bestimmt.

Da bei multimedialen Titeln die Sprache ein wichtiges Attribut ist, und sich darüber schnell Einschränkungen der Ergebnismenge durchführen lässen, wurde die Sprache als letzte Kategorie festgelegt.

Bei den Kategorien wurden die Cluster dadurch bestimmt, dass entweder die am meisten aufgetretenen Attributwerte ausgewählt wurden (Medientyp und Sprache), oder ähnliche Attributwerte unter einem Oberbegriff zusammengefasst wurden (Fachgebiet). Attributwerte, die selten auftreten oder nicht sinnvoll eingeordnet werden konnten, wurden in einem weiteren Cluster innerhalb der Kategorie zusammengefasst. Dies war insbesondere wegen der teilweise nicht vorhandenen Attributwerte von den Mediothekstiteln notwendig. Weitere Kategorien erschienen, sowohl aus Platz- als auch aus Datengründen, nicht sinnvoll. Die endgültige Einteilung der Kategorien und Cluster ist in Tabelle 2 zu sehen.

[19] *DROID*, Dynamic Remote Operation Incident Detection, Framework um Benutzereingaben und Aktionen zu erfassen und diese für die Weiterentwicklung der Software zu verwenden. Weitere Informationen unter: http://hci.uni-konstanz.de/research/projects/droid .

Medientypen	Fachgebiet	Sprache
DVD	Theater,Tanz,Film,Funk, Fernsehen	Deutsch
Video	Fremdspachen lernen	English
Tonträger	Mathematik- und Naturwissenschaften	Französisch
Mikrofiche	Geisteswissenschaften	Italienisch
Diskette	Rechts-,Wirtschafts- und Verwaltungswissenschaften	Russisch
Buch, Zeitschrift	Andere Fachgebiete	Spanisch
Andere Medientypen		Andere Sprachen

Tabelle 2: Kategorien und Cluster für Pocket SieveMap

Durch die Einteilung der Kategorien und zugehörigen Cluster konnte eine Baumstruktur für die SieveMap entwickelt werden, welche die Abhängigkeiten der einzelnen visuellen Komponenten veranschaulicht. Abbildung 19 zeigt die Kategorien und Cluster in einer geeigneten Baumstruktur.

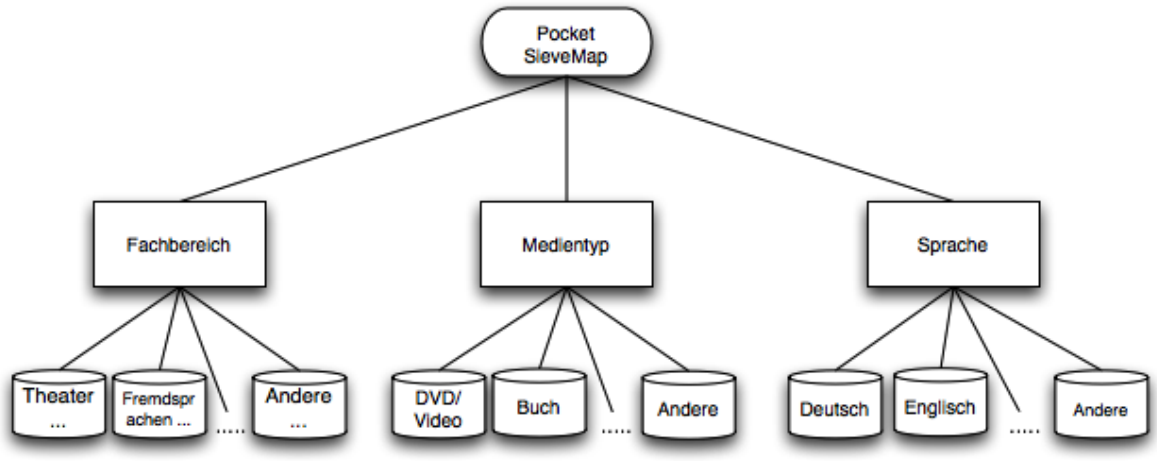

Abbildung 19: Baumstruktur der Pocket SieveMap am Beispiel der Mediothek

3.2.4 Pocket SieveMap 1

Das Konzept der Pocket SieveMap hat, wie bereits erwähnt, den Ursprung in einer TreeMap. So wird bei der Pocket SieveMap eine Baumstruktur (Abbildung 19) in eine übersichtliche visuelle Repräsentation überführt. Allerdings werden bei der Überführung in die visuelle Repräsentation gravierende Änderungen vorgenommen. Diese Änderungen sind notwendig um die Pocket SieveMap an das beschriebene Anwendungsszenario anzupassen, und dem Benutzer die Möglichkeit der direkten Manipulation zu geben.

In diesem Kapitel wird der ersten Entwurf eines Konzeptes, die Pocket SieveMap 1 (PSM1), beschrieben. Auf Details der Umsetzung wird in Kapitel 4.2.1 eingegangen. Abbildung 20 zeigt diesen ersten Entwurf. Bei der PSM1 wird die Baumstruktur folgendermaßen umgesetzt. Die Kategorien werden als einzelne Reiter dargestellt. In diesen Reitern sind die jeweiligen Siebkomponenten (Cluster) angeordnet. Die Größe der Cluster wird dabei bestimmt durch die Anzahl der jeweiligen Mediothekstitel in diesem Cluster proportional zur Gesamtanzahl. Alle Cluster einer Kategorie werden strukturiert nebeneinander angeordnet. Die Menge der enthaltenen Mediothekstitel in einem Cluster wird dem Benutzer neben dem Namen des Clusters angezeigt.

Der Benutzer kann durch direkte Manipulation mit der PSM1 einzelne Cluster zusammenschieben und erhält dadurch die Möglichkeit dynamisch seine Suchanfrage zu formulieren, ohne dabei Befehle oder Text eingeben zu müssen [Ahlberg1992]. Über die verschiedenen Reiter kann man die einzelnen Kategorien auswählen und sich die jeweiligen Cluster mit ihren Größen anzeigen lassen. Für den eigentlichen Siebvorgang benötigt man die Möglichkeit einen zusammengestellten Cluster auszuwählen und somit die Suchanfrage Stück für Stück zu formulieren. Bei der PSM1 kann der Benutzer einen Cluster mit dem Stylus auswählen und die Menge wird auf die Titel begrenzt, die in diesem Cluster enthalten sind. Gleichzeitig wird dem Benutzer angezeigt, in welcher Kategorie er welchen Cluster ausgewählt hat. Durch einen Knopf kann die Auswahl eines Clusters wieder rückgängig gemacht werden. Die Kategorie erscheint wieder mit den ursprünglichen Clustern und der Benutzer kann bei Bedarf ein neues Cluster auswählen. Abbildung 21 zeigt den Verlauf der dynamischen Erstellung der Suchanfrage mittels direkter Manipulation.

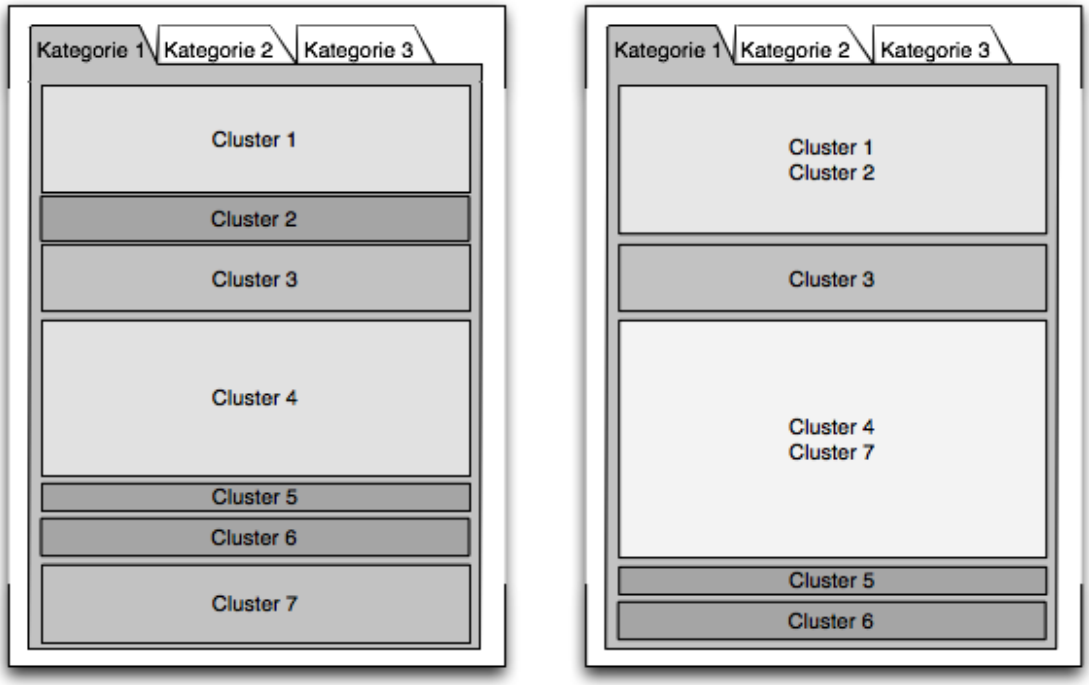

Abbildung 20: Schematischer Aufbau der Pocket SieveMap 1

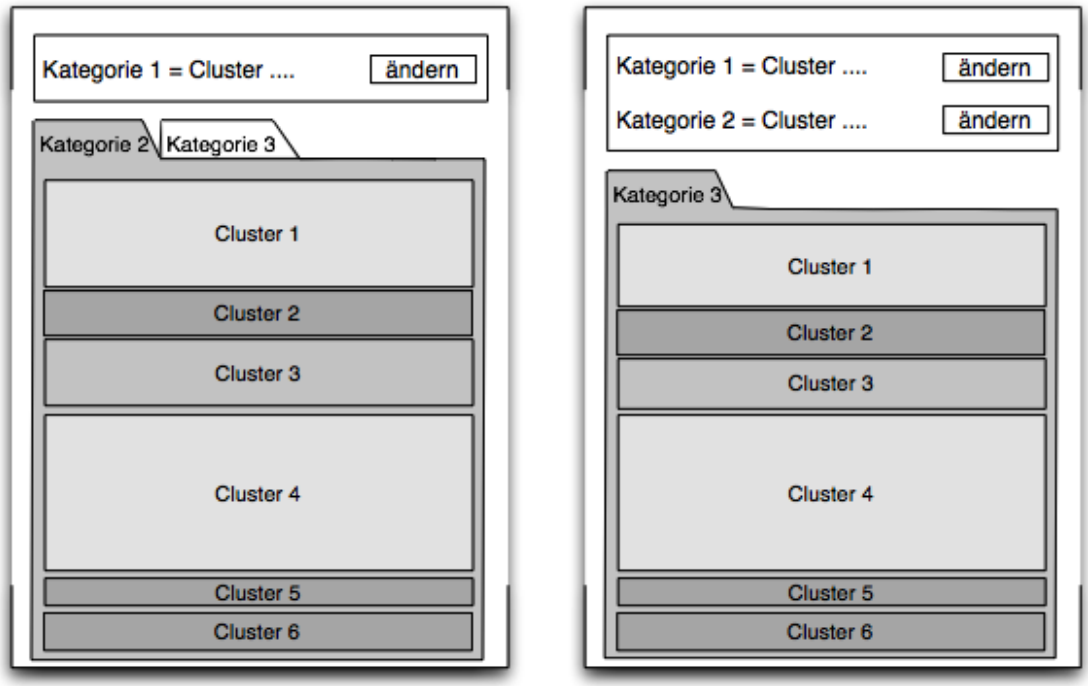

Abbildung 21: Verlauf der dynamischen Suche bei Pocket SieveMap 1

Bei Diskussionen über die PSM1 wurden einige Nachteile dieser Repräsentation sichtbar. Der Benutzer hat die Möglichkeit visuell seine Suchanfrage zu formulieren. Allerdings ist hierfür ein mehrstufiger Prozess notwendig, weil jeweils nur eine Kategorie im Blickfeld des Benutzers ist. Es kann die Kategorie durch Auswählen eines Reiters vom Benutzer bestimmt werden, allerdings wird vom System eine Reihenfolge der Reiter festgelegt, die der Benutzer unter Umständen anders gewählt hätte. Kleine informelle Benutzertests haben auch gezeigt, dass nicht klar ist, dass man die Kategorien über die Reiter wechseln kann. Das Zusammenfügen der Cluster über Drag & Drop[20] wurde allerdings schnell verstanden. Einen weiteren Nachteil hat die PSM1 darin, dass durch das Anzeigen der bereits gesiebten Kategorien der Platz für die eigentliche SieveMap immer kleiner wird. Bedingt durch die Tatsache, dass die proportionale Größe der Cluster über die Höhe ausgedrückt wird, werden die visuellen Unterschiede zwischen den einzelnen Clustern immer geringer.
Um die Vorteile der Pocket SieveMap als visuelles Filterinstrument besser auszunutzen, wurde die PSM1 genauer analysiert, mit folgenden Ergebnissen:

- Informationen (Name und Größe), die im Cluster dargestellt werden, benötigen nicht viel Bildschirmplatz. Somit können die Cluster kleiner werden.
- Alle Kategorien sollten parallel sichtbar sein, damit die Möglichkeit der dynamischen Suchanfrageformulierung visuell besser klar wird und einfacher durchgeführt werden kann.
- Die Anzahl der Titel in einem Cluster sollte über die Breite ausgedrückt werden, damit die proportionalen Unterschiede auch bei Änderungen der Ansicht gleich bleiben.

Mit Hilfe dieser Ergebnisse entstand eine weiterentwickelte PSM1, die Pocket SieveMap 2. Diese wird als endgültige Version der Pocket SieveMap im Folgenden beschrieben.

[20] *Drag & Drop* (englisch *drag & drop* = Ziehen und Fallenlassen) ist eine Methode zum Bewegen von Daten in einem Computerprogramm mittels eines Eingabegerätes.

3.2.5 Pocket SieveMap 2

Der schematische Aufbau der Pocket SieveMap 2 (PSM2) ist in Abbildung 22 zu sehen. Bei der PSM2 sind die Kategorien und Cluster ebenfalls auf der Baumstruktur aus Abbildung 19 aufgebaut.

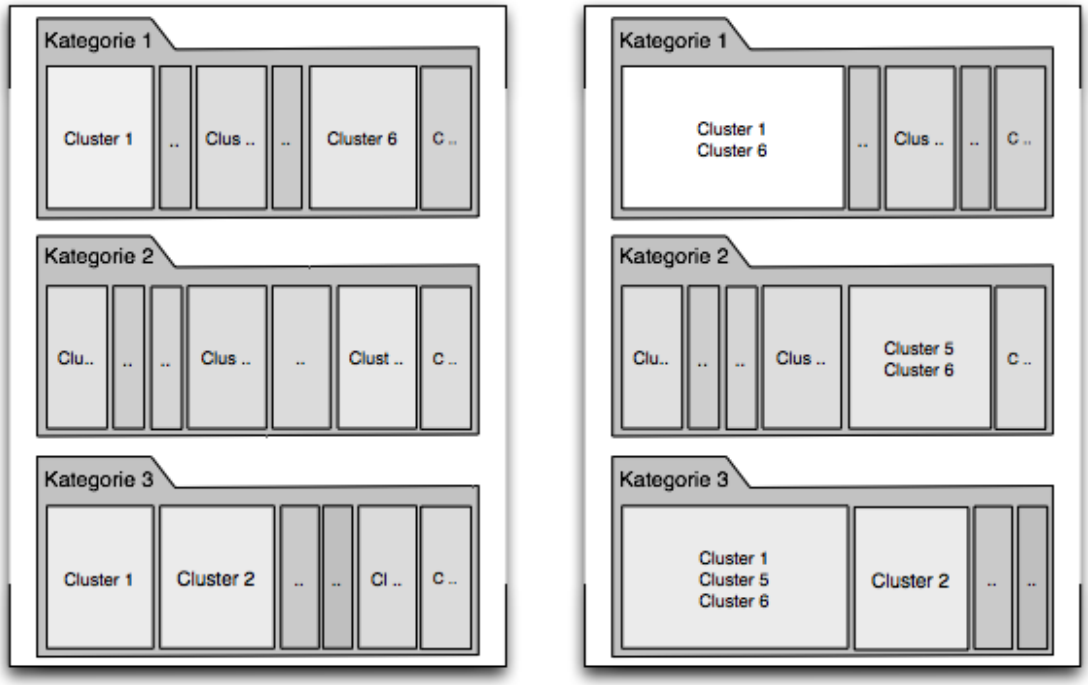

Abbildung 22: Schematischer Aufbau der Pocket SieveMap 2

Hier sind alle Kategorien gleichzeitig sichtbar und dem Benutzer wird dadurch nicht von dem System eine bestimmte Kategorie zu Beginn vorgesetzt. Dies bedeutet, der Benutzer hat einen Überblick über den durchsuchten Informationsraum, und durch das Zusammenführen und Auswählen einzelner Cluster ist eine schnelle Einschränkung der Titelmenge möglich. Das Zusammenführen von Clustern mittels Drag & Drop ist ebenso wie bei der PSM1 möglich. Die Auswahl eines Clusters innerhalb einer Kategorie erfolgt durch Klicken auf den gewünschten Cluster. Abbildung 23 zeigt den Verlauf einer visuellen Filterung durch Auswählen von Clustern. Die gesiebte Kategorie wird dabei soweit verkleinert, dass die Beschreibung der ausgewählten Cluster und ein Knopf, um die Auswahl rückgängig zu machen, dargestellt werden kann. Der nun frei gewordene Bildschirmplatz wird gleichmäßig auf die noch nicht gesiebten Kategorien verteilt. Die Höhe der Cluster

wird dadurch größer. Die proportionale Größe der Cluster zur Gesamtgröße der Kategorie wird über die Breite ausgedrückt. Dadurch bleiben die Unterschiede der einzelnen Clustergrößen erhalten und sind weiterhin visuell unterscheidbar. Durch die Darstellung von über dreimal so vielen visuellen Elementen wie bei der PSM1 sind bei der PSM2 einige Clusterbeschriftungen nicht darstellbar. Um sicherzustellen, dass die Größe des Clusters angezeigt werden kann und dass Cluster mit sehr wenigen Titeln nicht komplett verschwinden, muss eine Mindestbreite für ein Cluster festgelegt werden. Auf graphische Probleme bei der Darstellung der PSM2 und Lösungen dafür wird in Kapitel 4.2 näher eingegangen.

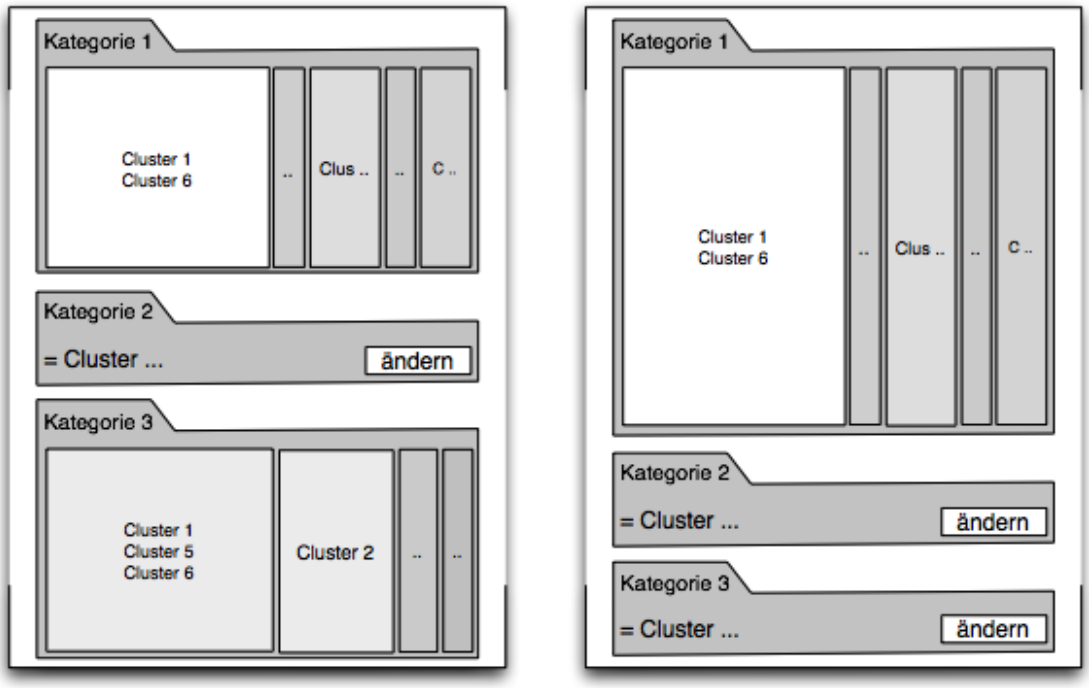

Abbildung 23: Verlauf der dynamischen Suche bei der Pocket SieveMap 2

Die PSM 2 ermöglicht eine übersichtliche visuelle Präsentation des Informationsraumes und der Benutzer hat die Möglichkeit der direkten Manipulation. Dadurch kann der Benutzer dynamisch eine Suchanfrage formulieren und die PSM2 wird zu einem effektiven visuellen Filterwerkzeug um schnell und einfach die Titelmenge einschränken zu können. Um die gefilterten Titel anzuzeigen, wird eine zweite Komponente für ein vollständiges visuelles Suchsystem benötigt. Diese

zweite Komponente, bezeichnet als Mobile Media Grid, wird in Kapitel 3.2.6 beschrieben.

Des Weiteren ist es für ein vollständiges visuelles Suchsystem notwendig eine Textsuche zur Verfügung zu stellen, um schnelle direkte Suchanfragen zur weiteren Einschränkung der Titelmenge zu ermöglichen. Abbildung 24 zeigt auf der linken Seite die integrierte Textsuche in die PSM2. Die Textsuche wird visuell gleich wie eine Kategorie dargestellt, weil auch durch die Textsuche die Titelmenge eingeschränkt wird und die anderen Kategorien direkt dadurch beeinflusst werden. Wenn ein Suchbegriff eingegeben und bestätigt wird, werden die Größen aller Cluster in den noch nicht gesiebten Kategorien neu berechnet und aktualisiert. Dadurch bekommt der Benutzer direktes Feedback über seine Textsuche. Die Textsuche kann ebenso wie die Auswahl eines Clusters in einer Kategorie wieder rückgängig gemacht werden.

Wenn alle Kategorien gesiebt wurden, entsteht, unabhängig davon, ob Suchbegriffe eingegeben wurden, freier Bildschirmplatz, der für eine natürlichsprachliche Beschreibung der Suchanfrage genutzt werden kann. Der Benutzer erhält dadurch eine zusammengefasste Beschreibung seiner Suchanfrage in natürlicher Sprache und kann leicht überprüfen, ob seine Suchanfrage korrekt ist. Dies ist rechts in Abbildung 24 zu sehen.

Die Idee der natürlichsprachlichen Zustandsanzeige kommt aus dem WOB-Modell (auf der Werkzeugmetapher basierende strikt objektorientierte graphisch-direktmanipulative Benutzungsoberfläche) [Krause1995]. In diesem Modell ist die Forderung nach einer natürlichsprachlichen veränderbaren Zustandsanzeige enthalten. Veränderbar ist hier die natürlichsprachliche Zustandsanzeige nicht. Allerdings sind die Vorteile, die im WOB-Modell beschrieben werden, unter anderen die Überprüfbarkeit der Suchanfrage durch den Benutzer, nicht von der Hand zu weisen.

Die PSM2 ist mit den oben beschriebenen Veränderungen ein visuelles Filterwerkzeug, welches die in Kapitel 3.2.2 beschriebenen Anforderungen erfüllt. Der gesamte Informationsraum wird übersichtlich dargestellt. Somit wird dem Benutzer ein Gefühl für den durchsuchten Informationsraum vermittelt. Durch die Möglichkeit, die Cluster zu kombinieren und diese dann auszuwählen, wird dem Benutzer visuell die Möglichkeit gegeben schnell eine Suchanfrage zu formulieren.

Auf weitere Details, vor allem bezüglich der Interaktion mit der PSM2, wird in Kapitel 4.2.1 und 5.2 eingegangen.

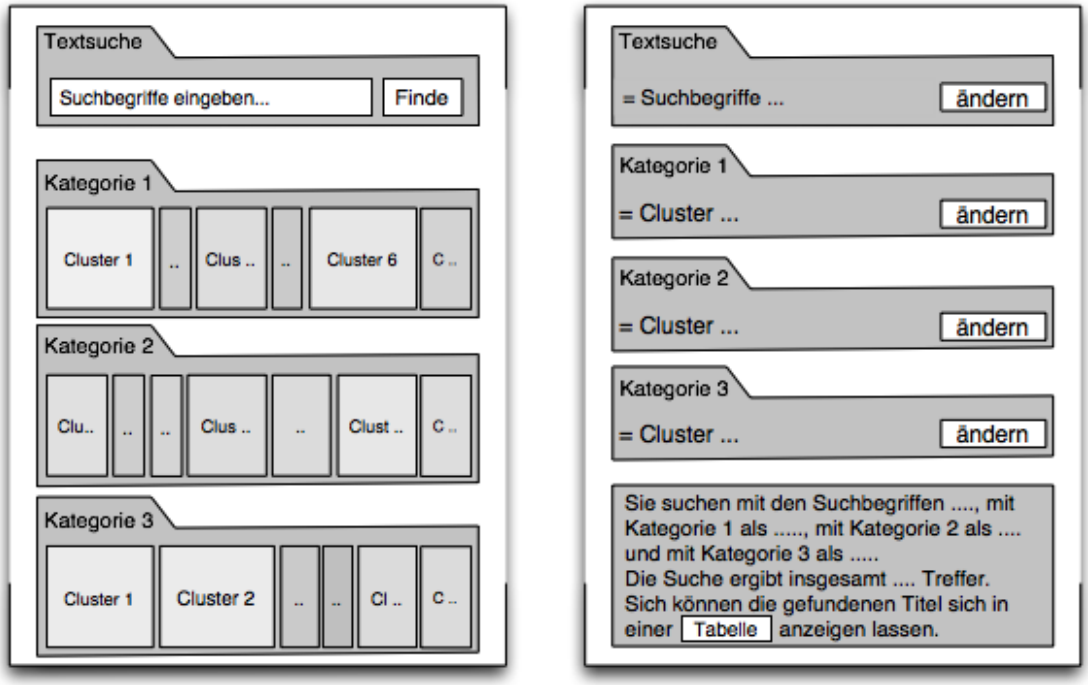

Abbildung 24: Textsuche und textuelle Darstellung der Suchanfrage in der Pocket SieveMap

3.2.6 Mobile Media Grid

Mit der Pocket SieveMap (PSM) ist ein effektives visuelles Instrument zur Filterung der Mediothekstitel entstanden. Ein visuelles Suchsystem sollte auch über eine komfortable Möglichkeit der Titelanzeige verfügen. Mit der Media Grid [Grün2004] und der Hypergrid [Jetter2005] existieren in MEDIOVIS bereits zwei Visualisierungen von tabellarischen Titelansichten, die eine Detailansicht eines Titels ermöglichen, ohne dabei die Tabellenansicht verlassen zu müssen. Dasselbe sollte auch für die Titelansicht in MOBILE MEDIOVIS gelten. Eine tabellarische Ansicht der Titel hat auch auf kleinen Bildschirmen Vorteile. Benutzer sind im Umgang mit Tabellen vertraut und durch die listenartige Anordnung können Titel besser verglichen werden, ohne dabei den Kontext zu verlieren. Die Media Grid, die im Rahmen des Projektes MEDIOVIS entstanden ist, zeichnet sich dadurch aus, dass verschiedene Granularitätsstufen definiert sind, die je nach Detailgrad des Titels in der Ansicht

angezeigt werden [Grün2004]. Der Detailgrad wird in der Media Grid in MEDIOVIS durch den Benutzer bestimmt. Ein Klick mit der linken Maustaste erhöht und ein Klick mit der rechten erniedrigt den Detailgrad um eine Stufe. Die einzelnen Granularitätsstufen sind in Tabelle 3 dargestellt.

	Kat. 1	Kat. 2	Kat. 3	Kat. 4	Kat. 5	Kat. 6	Kat. 7
I	Jahr	Sprache	Status	Titel	Fachgebiet	Medientyp	Ausleihen
II		Stadt Verlag	Leihfrist	Untertitel Originaltitel	Signatur	Format ISBN / ISSN	
III				Plakat Beschreibung Details			
IV				Trailer			

Tabelle 3: Darstellung der vier Granularitätsstufen der Media Grid

Durch die Einteilung der einzelnen Attribute in die Kategorien und die Zuordnung der Attribute zu einem Detailgrad wird definiert, wann welches Attribut erscheint. Durch diese Anordnung der Attribute ist es möglich, in der Media Grid viele Treffer auf einem Bildschirm als Übersicht anzubieten und sich gleichzeitig einzelne Treffer detailliert anzeigen zu lassen ohne dabei die Tabelle zu verlassen. Dadurch können Titel besser verglichen werden. Abbildung 25 zeigt die vier Granularitätsstufen einer einzelnen Zeile in der Media Grid. Der Benutzer kann mittels Interaktion selbst bestimmen, wie viele Details er von einem Titel sehen will.

Abbildung 25 zeigt auch, dass die Media Grid in dieser Form nicht auf einem Pocket PC umzusetzen ist. Während bei MEDIOVIS, bedingt durch große Bildschirme, für die Anzeige der sieben Spalten bei den meisten Desktop PCs mindestens 1024 Pixel in der Breite zur Verfügung stehen, sind es bei einem Pocket PC höchstens 480 Pixel. Selbstverständlich könnte man den Pocket PC auch im Querformat benutzen. Dadurch hätte man dann 640 Pixel in der Breite zur Verfügung. Allerdings müsste dann beim Übergang von der PSM zur Ergebnisansicht das Gerät gedreht werden und dies würde den flüssigen Übergang der zwei Komponenten brechen. Deswegen wurde das Konzept der Media Grid auf kleine Bildschirme angepasst.

Daraus entstand die Mobile Media Grid (MMG). Mobile soll hier andeuten, dass die Darstellung für mobile Geräte konzipiert ist. In der MMG sollten alle in Tabelle 3 aufgeführten Attribute eines Titels ebenfalls vorhanden sein.

Abbildung 25: Einzelne Zeile der Media Grid, visualisiert in den vier Granularitätsstufen

Es war nötig die Attribute neu anzuordnen, um der begrenzten Breite des Bildschirmes gerecht zu werden. Tabelle 4 zeigt die Aufteilung der Attribute auf die einzelnen Granularitätsstufen.

Stufe vier wurde bei der MMG weggelassen, weil die Einbindung der Trailer im Rahmen dieses Projektes nicht möglich war. Prinzipiell wäre jedoch eine Einbindung von Trailern und weiteren zusätzlichen Informationen in der Zukunft möglich.

Granularitätsstufe	Dargestellte Attribute
Stufe 1	Ausleihstatus, Titel, Jahr
Stufe 2	Medientyp, Sprache, Fachgebiet, Signatur
Stufe 3	Untertitel, Poster, ISBN/ISSN, Beschreibung, Personen, Details, Original, Herausgeber, Stadt, Institutionen, Leihfrequenz, Format

Tabelle 4: Darstellung der 3 Granularitätsstufen der Mobile Media Grid in MOBILE MEDIOVIS

Um dem Benutzer in der Startansicht der MMG eine Übersicht der Titel zu geben, wurden für Stufe 1 die Attribute Titel und Jahr ausgewählt. Diese lassen sich gut in einer Zeile darstellen und die Mediothekstitel lassen sich meistens durch den Titel und das Jahr eindeutig bestimmen. Des Weiteren wird direkt in der Startansicht der Ausleihstatus angezeigt. Dabei wird die Kontrollbox, die auch zum Auswählen einer Zeile verwendet wird, entweder grün, wenn der Titel verfügbar ist, oder rot, wenn er nicht verfügbar ist, umrahmt. Wenn der Benutzer mehr Detailinformationen zu einer bestimmten Zeile will, kann er durch Klicken mit dem Stylus auf die Zeile in Stufe 2 gelangen. Dort werden zusätzlich zu den in Stufe 1 schon vorhandenen Attributen die Attribute Medientyp, Sprache, Fachgebiet und Signatur angezeigt. Der Benutzer hat bei der PSM die Möglichkeit nach den Kategorien Fachgebiet, Medientyp und Sprache zu sieben. Um zu bestätigen, dass es auch Titel aus den jeweils gesiebten Kategorien sind, ist es notwendig dem Benutzer die Möglichkeit zu geben, dies zu überprüfen. Die Signatur wurde ebenfalls in Stufe 2 genommen, weil über dieses Attribut der Standort des Titels in der Mediothek gefunden werden kann. Um dies schnell zu ermöglichen, ist es sinnvoll dies frühzeitig anzuzeigen. Stufe 3 stellt dann alle Attribute, die zu dem jeweiligen Titel in der Mediothek vorhanden sind, dar. Der Benutzer kann dies erreichen, indem er ein zweites Mal auf den bereits fokussierten Titel klickt. Durch einen weiteren Klick wird Stufe 1 erreicht und es werden wieder Titel und Jahr des jeweiligen Mediothekstitel dargestellt.
In Abbildung 26 ist der Aufbau der MMG dargestellt. Es ist jeweils ein Beispiel für Stufe 2 und Stufe 3 angedeutet. Selbstverständlich können sich mehrere Titel innerhalb der Tabelle in verschiedenen Levels befinden und der Benutzer kann trotzdem durch die Tabelle navigieren. Sollte der Bildschirmplatz für die angezeigten Titel nicht ausreichen, wird rechts eine Bildlaufleiste eingeblendet, mit der durch die Tabelle navigiert werden kann. Dies wird dann notwendig, wenn mehrere Titel in Stufe 3 dargestellt oder sehr viele Treffer angezeigt werden. Über die farblich markierten Kontrollboxen ist zum einen der Ausleihstatus sichtbar und zum anderen jederzeit eine Selektion eines Titels möglich. Außerdem hat der Benutzer die Möglichkeit die MMG nach den Attributen der ersten Stufe, Titel und Jahr, auf- oder absteigend zu sortieren. Die Sortierung kann durch Klicken auf die jeweilige Spaltenüberschrift geändert werden. Bei jedem Klick wird dabei die Sortierung

umgekehrt. Die Sortierreihenfolge wird durch eine Pfeilspitze neben dem zuletzt sortierten Attribut angezeigt.

Das Konzept der Mobile Media Grid bietet auf der einen Seite eine übersichtliche Darstellung der Titel in einer tabellenbasierten Ansicht, und auf der anderen Seite zu jedem Zeitpunkt die Möglichkeit, sich einfach Detailinformationen zu einem gewünschten Titel anzeigen zu lassen. Auf weitere Umsetzungsdetails wird in Kapitel 4.2.2 eingegangen. Die Interaktionsmöglichkeiten der Mobile Media Grid werden in Kapitel 5.2 detailliert beschrieben.

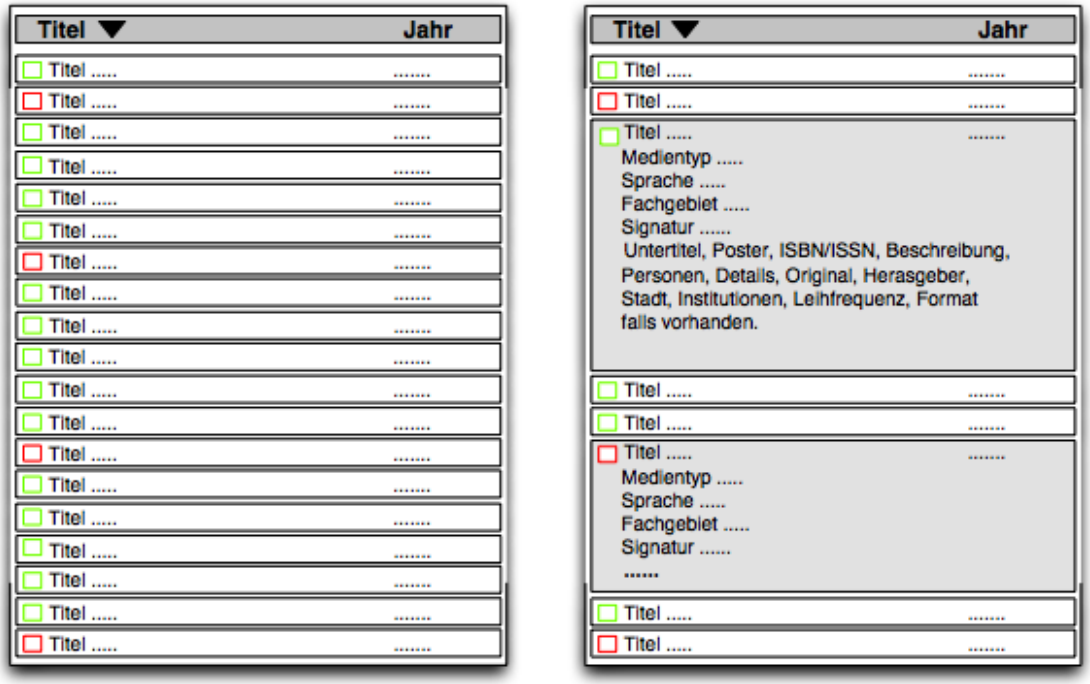

Abbildung 26: Visueller Aufbau mit den drei Granularitätsstufen der Mobile Media Grid

4 Implementierung

Nachdem die technischen Rahmenbedingungen geklärt waren und die Auswahl der Implementierungsplattform getroffen war, wurde parallel zu den ersten Ideen und Konzepten mit dem ersten Teil der Implementierung, der Portierung des Datenmodells, von MOBILE MEDIOVIS begonnen. Die Implementierung teilte sich dabei in drei Schritte. Zunächst wurde eine Analyse über die Machbarkeit der Portierung von der Java Anwendung MEDIOVIS in das .NET Compact Framework durchgeführt. Anschließend wurden die möglichen Teile mittels eines teilweise automatisierten Prozesses von Java in die Programmiersprache C# portiert. Nach diesem Schritt folgte die eigentliche Implementierung der Pocket SieveMap und der Mobile Media Grid. Dieses Kapitel wird diese drei Schritte beschreiben und dabei auf wichtige Details und Probleme, die bei der Implementierung aufgetaucht sind, eingehen.

4.1 Portierung des Datenmodells

Die bereits erwähnte Ähnlichkeit der beiden objektorientierten Programmiersprachen Java und C# ließ den Schluss zu, dass eine relativ rasche Portierung der Java Anwendung MEDIOVIS möglich sein musste. Zunächst wurde überprüft wie, und vor allem welche Teile von MEDIOVIS in das .NET Compact Framework (CF) portiert werden können, um eine vollständige neue Implementierung zu verhindern. Bei dieser Analyse kamen die folgenden Erkenntnisse zum Vorschein. Zwar konnten die vollständigen Visualisierungen, die bereits für MEDIOVIS implementiert waren, durch die unterschiedlichen Herangehensweisen der Oberflächenprogrammierung bei Java und C# nicht übernommen werden. Dennoch konnten durch das Benutzen des Java Language Conversion Assistant 3.0 beta [CF-4], der kostenlos von Microsoft zur Verfügung gestellt wird, große Teile des Datenmodells, die in MEDIOVIS implementiert sind, übernommen werden. Manche Klassen bereiteten Probleme und mussten manuell angepasst und verändert werden. Einige Klassen und Pakete mussten ganz ausgelassen werden.

Abbildung 27 zeigt zur Verdeutlichung das Java Datenmodell, welches im Projekt MEDIOVIS entstanden ist [Grün2004]. Dieses Datenmodell ist in verschiedene Pakete und Klassen aufgeteilt, welche die unterschiedlichen Aufgaben, die zur

Datenverarbeitung und zur Suche in den Mediotheksdaten notwendig sind, ausführen.

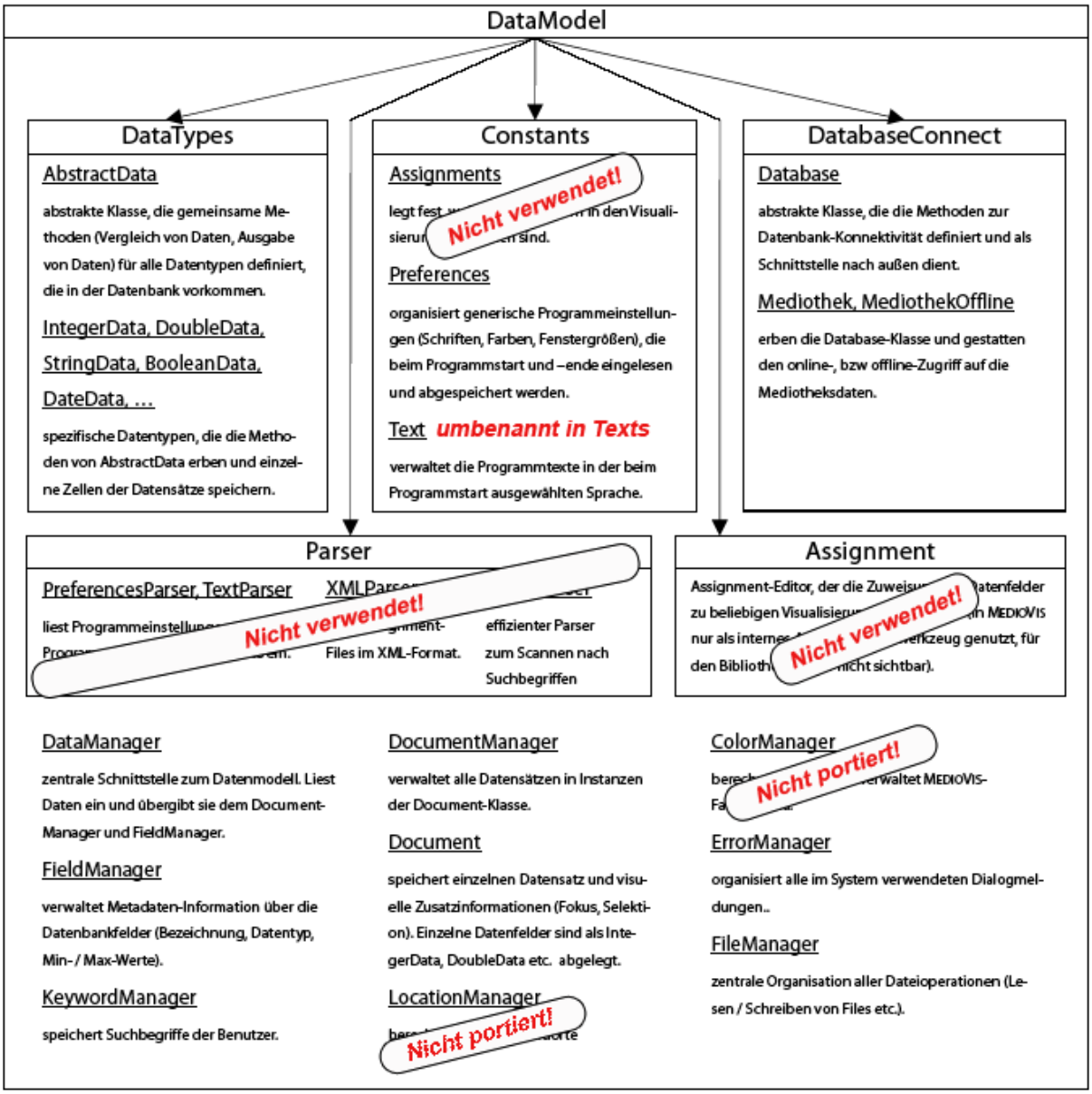

Abbildung 27: Datenmodell bei MedioVis

Im Paket DataTypes werden für alle Datentypen, die in einem Dokument der Mediothek vorkommen, Klassen mit Basisfunktionalitäten bereitgestellt. Alle diese Klassen leiten sich von der abstrakten Klasse AbstractData ab. Dieses Paket wurde vollständig in das CF portiert.

Das Paket Constants enthält alle Klassen, in denen Einstellungen und Konstanten für die Anwendung definiert sind. In der Klasse Preferences sind zum Beispiel

Schriftgrößen, Farben und weitere Einstellungen definiert. Die Klasse Texts, welche aus Kompatibilitätsgründen von Text in Texts umbenannt werden musste, enthält alle Beschriftungen, die in der Anwendung verwendet werden.
Die Klasse Assigments wurde nicht portiert, da sie für das visuelle Suchsystem nicht benötigt wird. Ebenso wurde das gesamte Paket Assignment nicht portiert. Hierbei handelt es sich um einen Editor, mit dem in MEDIOVIS die verschiedenen Ansichten kombiniert und die Belegung der einzelnen Attribute innerhalb der Ansichten verändert werden können. Da es in MOBILE MEDIOVIS bisher nur eine Kombination von Ansichten gibt, wurde auf dieses Paket verzichtet. Ohnehin ist eine Portierung dieses Paketes, wegen der benutzten visuellen Komponenten von Java, schwierig.
Im Paket DatabaseConnect können verschiedene Arten, wie auf die Daten der Mediothek zugegriffen werden kann, definiert werden. Alle Klassen werden dabei von der abstrakten Klasse Database abgeleitet. Dieses Paket wurde komplett portiert und an die verfügbaren Datenzugriffsmöglichkeiten angepasst. So ist es in MOBILE MEDIOVIS bei aktiver Internetverbindung möglich auf die auch bei MEDIOVIS verwendete PostgreSQL-Datenbank und bei nicht aktiver Internetverbindung auf eine lokal auf dem Gerät installierte SQL-Datenbank zuzugreifen. Die lokal installierte SQL-Datenbank wurde eingerichtet, um die Anwendung überall präsentieren zu können. Die Geschwindigkeit der Datenaufbereitung ist bei dieser Art der Verbindung allerdings drastisch langsamer.
Das Paket Parser wurde komplett ausgelassen bei der Portierung, weil die darin enthaltenen Klassen und Funktionen für die Visualisierung nicht relevant sind und viele Probleme bei der Portierung auftraten.
Die weiteren Klassen DataManager, FieldManager, KeywordManager, DocumentManager, Document, ErrorManager und FileManager, welche direkt im Paket DataModel liegen, wurden mit kleineren Änderungen komplett portiert. Alle diese Klassen werden benötigt, um die gewünschte Anfrage an die aktive Datenbank zu senden und die zurückgelieferten Ergebnisse in eine interne Struktur (Dokument) zu bringen um die Daten anzeigen zu können. Nicht portiert wurden hier die Klassen ColorManager und LocationManager. Die Klasse ColorManager wurde nicht benötigt, weil die Farbwahl in MOBILE MEDIOVIS bewusst einfach gehalten wurde und die Farbeinstellungen direkt in der Klasse Preferences möglich sind. Die Klasse

LocationManager beinhaltet bei MEDIOVIS die Funktion sich den Standort eines Titels in der Mediothek anzeigen zu lassen. Diese Funktion ist bisher noch nicht umgesetzt.

Durch die vollständige Portierung aller benötigten Komponenten konnte somit für das Projekt MOBILE MEDIOVIS auf eine ausgereifte Schnittstelle zur Datenaufbereitung der über 26.000 Titel in der Mediothek zugegriffen werden. An dieser Stelle möchte ich mich bei Christian Grün bedanken. Ohne seine fundierten Kenntnisse des Datenmodells der Java Anwendung MEDIOVIS [Grün2004] und seinen persönlichen Einsatz wäre eine so schnelle Portierung nach C# nicht möglich gewesen.

4.2 Umsetzung der Visualisierung

Durch die relativ rasche Portierung des Datenmodells konnte früh mit dem Beginn der Umsetzung der Visualisierung im .NET Compact Framework (CF) mit der Programmiersprache C# begonnen werden. Das Umsetzen der Visualisierung war geprägt von der Suche nach gleichwertigen Komponenten und Funktionen, die in Java vorhanden sind, und von der Tatsache, dass das CF eine Teilmenge vom .NET Framework ist. Einige Klassen und Methoden sind nicht vorhanden und müssen über Umwege nachgebildet werden. In Kapitel 6.1 werden Erfahrungen und Probleme mit dem CF berichtet.
Im Folgenden wird die Umsetzung der Pocket SieveMap und der Mobile Media Grid auf einem Pocket PC beschrieben. Da bei der Umsetzung keine neuen Programmierparadigmen entstanden sind, wird auf Quellcodeauflistungen verzichtet. An relevanten Stellen wird Pseudo-Code benutzt um die Umsetzung zu verdeutlichen.

4.2.1 Pocket SieveMap

Das Konzept der Pocket SieveMap (PSM) beruht auf der Idee einer TreeMap. Die darzustellenden Komponenten sind ebenfalls durch eine Baumstruktur ausgedrückt, wie in Abbildung 19 dargestellt ist. Bei der PSM ist allerdings kein rekursiver Algorithmus notwendig um die Blätter und Knoten visuell aufzubereiten. Um die

Baumstruktur in eine visuelle Repräsentation umzuwandeln, wurde nach in Pseudo-Code 1 aufgelistetem Schema vorgegangen.

```
Für jede Kategorie
        Erstelle Kategorie
        Für jedes Cluster in der Kategorie
                Bestimme Anzahl der enthaltenen Objekte im Cluster.
                Wenn Anzahl der Objekte im Cluster > 0
                        Berechne visuelle Größe des Clusters
                        Füge Cluster zu der Kategorie hinzu.
        Ende Für.
Ende Für.
```

Pseudo-Code 1: Erzeugung der Kategorien und Cluster der Pocket SieveMap

Bei der ersten Version der Pocket SieveMap, die parallel zu den ersten Ideen entstanden ist, wurde jeweils nur die aktive Kategorie nach oben angegebenen Pseudo-Code erzeugt. Die beiden inaktiven Kategorien in der Startansicht wurden erst erzeugt, wenn sie vom Benutzer ausgewählt wurden. Dabei wurde die aktuelle Kategorie direkt in dem Hauptfenster der Kategorie dargestellt und auch die Cluster wurden direkt in das Hauptfenster platziert. Die inaktiven Kategorien können über Knöpfe vom Benutzer ausgewählt werden und die ausgewählte Kategorie ersetzt dann die aktuelle Kategorie. Die Umsetzung und der Siebvorgang sind in den Abbildungen 28 bis 31 zu sehen. Dabei ist in Abbildung 28 die Startansicht dargestellt, die sich dem Benutzer präsentiert, wenn die PSM1 gewählt wird. Als Startkategorie wurde hier Fachgebiet gewählt. Die Kategorien wurden jeweils mit einer anderen Farbe versehen um sie besser visuell unterscheiden zu können. In der oberen linken Ecke wird zu jeder Zeit die derzeitige Größe der Treffermenge angezeigt. Abbildung 29 zeigt die PSM1, wenn ein Benutzer einen Cluster mittels Drag & Drop in einen anderen Cluster zieht. Dabei wird dem Benutzer über einen dem Stylus folgenden Tooltip angezeigt, welchen Cluster er gerade bewegt. Der bewegte Cluster wird außerdem durch einen grauen Rand gekennzeichnet. Nach Loslassen des Stylus über einem anderen Cluster wird der bewegte Cluster mit dem darunter liegenden Cluster verbunden.

Abbildung 28: Startansicht PSM1

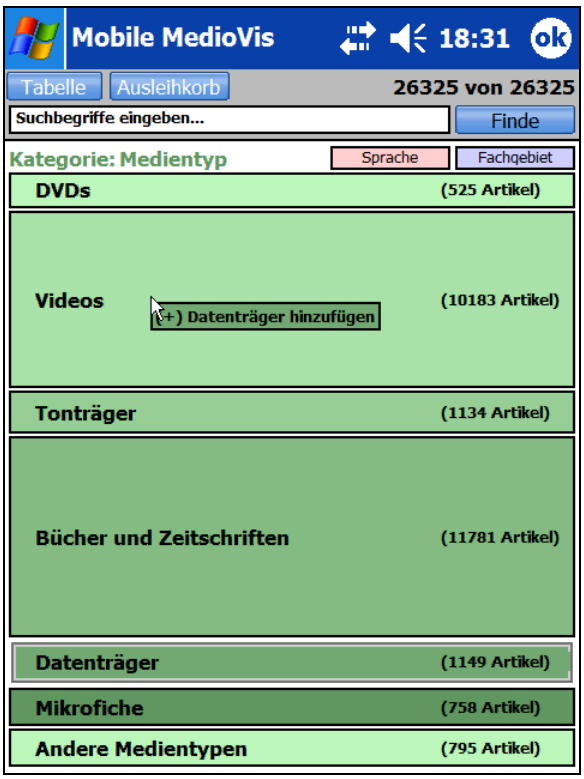

Abbildung 29: Cluster vereinen bei PSM1

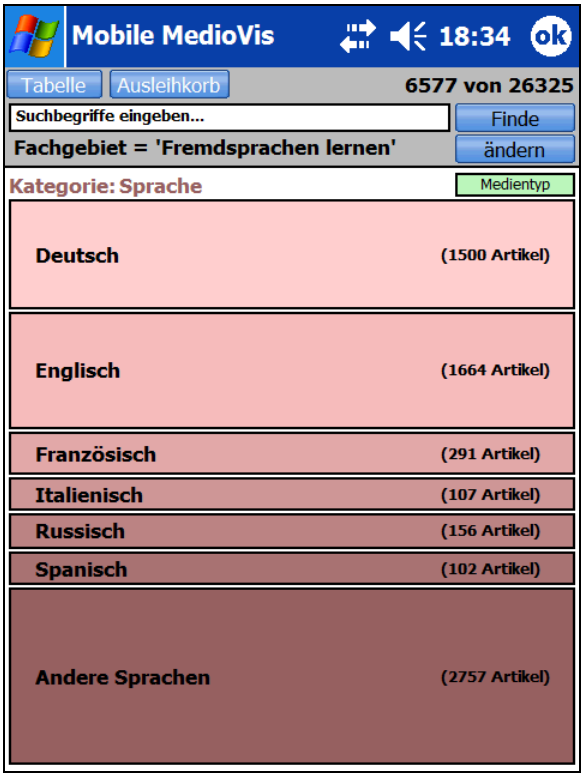

Abbildung 30: Eine Kategorie gesiebt bei PSM1

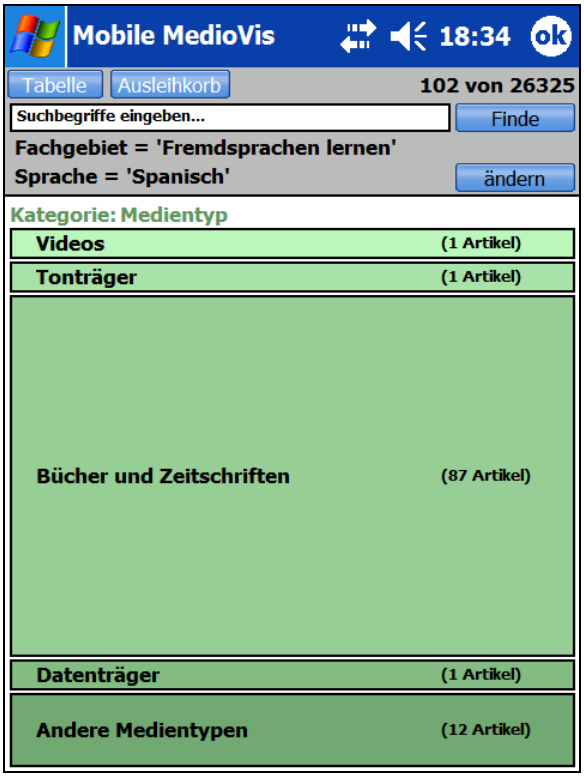

Abbildung 31: Zwei Kategorien gesiebt bei PSM1

Die Ansicht, die dem Benutzer erscheint, wenn er einen Cluster in einer Kategorie, in diesem Fall der Cluster *Fremdsprachen lernen* in der Kategorie *Fachgebiet*, gesiebt hat, wird in Abbildung 30 gezeigt. Wenn der Benutzer in der Kategorie Sprache Spanisch auswählt, erscheint nach einer Aktualisierung der Daten die letzte Kategorie (Abbildung 31). Die komplette Interaktion mit dem visuellen Suchsystem wird detailliert in Kapitel 5.2 beschrieben.
Wegen der Probleme der Bildschirmaufteilung und wegen der Tatsache, dass immer nur eine Kategorie für den Benutzer zu sehen ist, wurde diese erste Version nicht weiterverfolgt. Sie ist aber vollständig implementiert und verfügt über alle in Kapitel 3.2.4 beschriebenen Funktionalitäten.

Im Folgenden wird detailliert die Umsetzung der zweiten Version der PSM beschrieben, da ihre Vorteile überwiegen. Die Umsetzung der zweiten Version der PSM wurde dynamischer und effektiver vorgenommen. Bei der PSM wird für jede Kategorie ein Panel[21] erzeugt, welches auf das Hauptfenster der Anwendung gesetzt wird. Die Höhe einer Kategorie wird festgelegt durch die Höhe des verfügbaren Bildschirmplatzes, geteilt durch die Anzahl der Kategorien minus der Höhe, die für die Textsuche und weitere Steuerungskomponenten benötigt wird. Anschließend werden in die Kategorien die festgelegten Cluster gesetzt. Zuvor wird für jedes Cluster die Größe anhand einer SQL-Anfrage über die Datenbank ermittelt.
Die Clustergröße ergibt sich aus der Anzahl der darin enthaltenen Mediothekstitel proportional zur Gesamtkategoriegröße. Die Gesamtkategoriegröße ist durch die Größen aller Cluster in der Kategorie bestimmt. Um die SQL-Anfragen zu generieren und einen möglichst generischen PSM Aufbau zu haben, wurde für jede Kategorie ein Array bestehend aus Strings erstellt, in dem die Clusterinformationen und die notwendigen SQL-Anfragebegriffe definiert sind. XML[22] wurde nicht benutzt, weil der XML-Parser nicht ohne Probleme von Java nach C# portiert werden konnte. Das Array einer jeden Kategorie wird dabei wie folgt aufgebaut. Es wird ein

[21] Bei einem *Panel* handelt es sich im .NET Compact Framework um einen Container für weitere visuelle Komponenten.

[22] Die Extensible Markup Language, abgekürzt *XML*, ist ein Standard zur Erstellung maschinen- und menschenlesbarer Dokumente in Form einer Baumstruktur.

zweidimensionales String-Array verwendet. Das äußere Array dient dabei als Container für die Cluster. Die inneren Arrays beschreiben die Cluster der jeweiligen Kategorie in der Form, dass der erste String (Indexnummer 0) die Beschreibung des Clusters ausdrückt und alle danach folgenden Strings die Schlüsselwörter definieren, anhand derer die SQL-Abfrage erstellt wird. Ein Beispielaufbau der Clusterdefinition für die Kategorie Medientyp wird in Pseudo-Code 2 gezeigt. Bei der Zusammenstellung der SQL-Anfrage wird das Array durchlaufen und es wird in der jeweiligen Spalte der Datenbank ein Zählen der Titel nach den definierten Suchbegriffen durchgeführt.
Anhand der Anzahl der gefundenen Titel in der jeweiligen Spalte wird dann die Breite des Clusters bestimmt, so dass der Benutzer sofort visuell erkennen kann, welches Cluster besonders viele oder wenige Titel enthält. Die Höhe des Clusters ist durch die Höhe der Kategorie minus der Höhe, die zur Anzeige des Kategorienamens benötigt wird, definiert.

```
string[][] TYP_CLUSTERS =
{
        new string[]{"DVDs","DVD"},
        new string[]{"Videos","Video"},
        new string[]{"Tonträger","Tonträger"},
        new string[]{"Bücher und Zeitschriften","Buch","Zeitschrift","Fachzeitschrift"},
        new string[]{"Datenträger","CD-Rom","Diskette"},
        new string[]{"Mikrofiche","Mikrofiche"},
        new string[]{"Andere Medientypen","Kartenmaterial","Medienkombination",""}
};
```

Pseudo-Code 2: Aufbau der Clusterdefinitionen einer Kategorie bei der Pocket SieveMap

Nachdem alle Arrays durchlaufen sind, sind alle benötigen Größen berechnet und die Pocket SieveMap erscheint wie rechts in Abbildung 32 dargestellt. Der Benutzer hat die Möglichkeit die Cluster innerhalb der Kategorien per Drag & Drop zusammenzuführen (Abbildung 32 links) und kann durch Klicken auf ein gewünschtes Cluster dieses auswählen und die PSM wird neu aufgebaut.

Abbildung 32: Startansicht der Pocket SieveMap von Mobile MedioVis

Um dies zu realisieren, ist es nötig die jeweiligen Informationen über ein Cluster in einem Objekt Cluster zu speichern. Im Objekt Cluster sind die Bezeichnung, die Schlüsselwörter, die Größe, die Farbe und weitere benötigte Attribute gespeichert. Dieses Objekt Cluster wird zur visuellen Repräsentation in ein Objekt ShowCluster überführt, welches von der Basisklasse Panel abgeleitet wird. Das Objekt ShowCluster kann mehrere Clusterbeschreibungen und Schlüsselwörter aufnehmen und dient somit als Behälter für mehrere zusammengeführte Cluster. Abbildung 33 zeigt den Klassenaufbau der Pocket SieveMap.

Wenn in einer Kategorie ein Cluster gesiebt wird, werden anhand der zugeordneten Cluster in dem Objekt ShowCluster die Größen der anderen noch nicht gesiebten Kategorien berechnet und wieder dargestellt. Dadurch kann der Benutzer die Cluster beliebig innerhalb einer Kategorie zusammenführen und auswählen. Die Farbe des jeweiligen Cluster wird dabei über die darin enthalten Titel bestimmt. Je mehr Titel enthalten sind desto heller wird das ShowCluster. Sollte ein ShowCluster die ganze Breite einer Kategorie einnehmen, so wäre es weiß. Die Farbanpassung scheint deswegen, zusätzlich zur Breite der ShowCluster, sinnvoll, weil der Benutzer dadurch die Cluster besser unterscheiden kann und die PSM dynamischer wirkt.

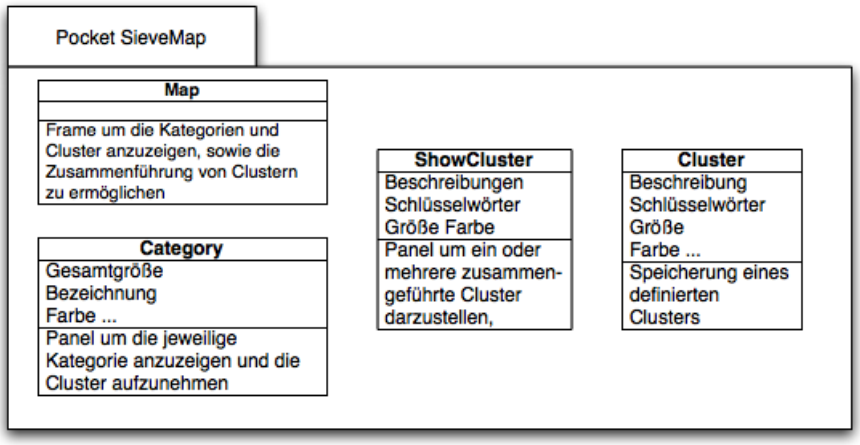

Abbildung 33: Klassenaufbau der Pocket SieveMap

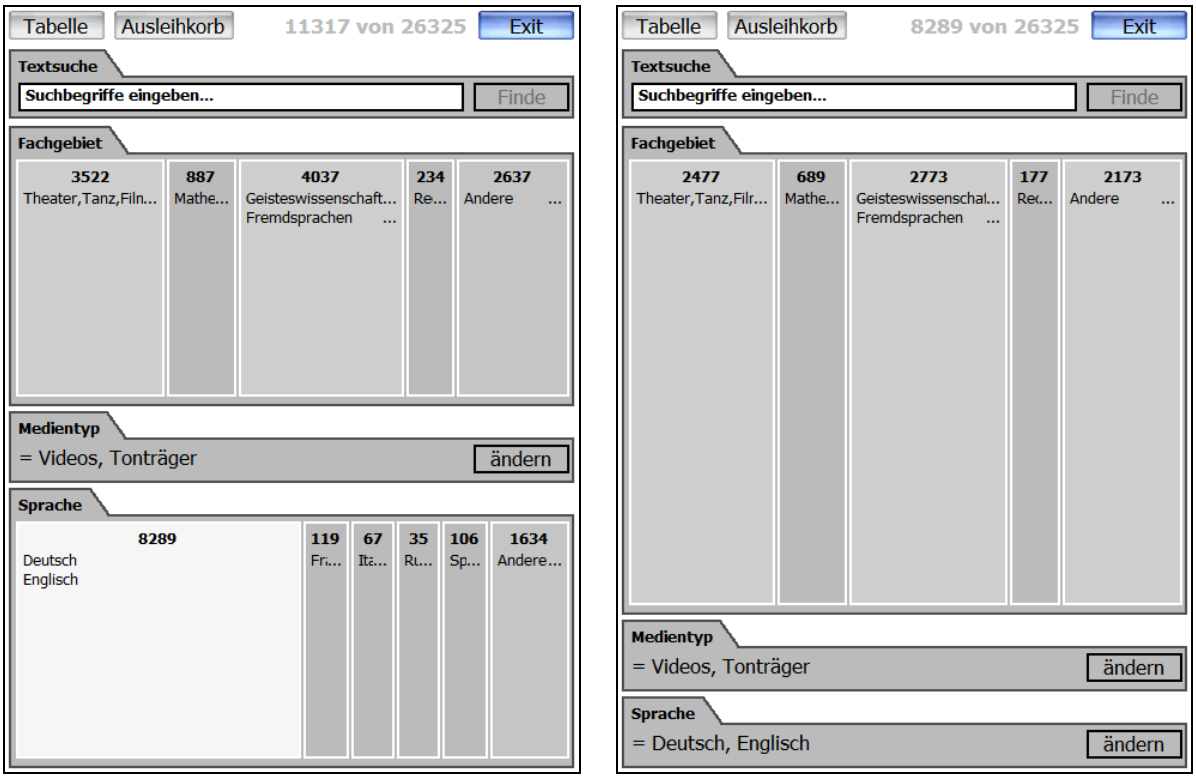

Abbildung 34: Zusammengeführte Cluster und gesiebte Kategorien bei der Pocket SieveMap

Abbildung 34 zeigt, wie zusammengeführte Cluster erscheinen und wie die gesiebten Kategorien in der PSM dargestellt werden.

Wie bereits in Kapitel 3.2.4 beschrieben, wird, wenn alle drei Kategorien gesiebt wurden, die Suchanfrage in einem weiteren Panel unterhalb der Kategorien in einer natürlichsprachlichen Zustandsanzeige dargestellt (Abbildung 24 rechts). Die

Umsetzung dieser natürlichsprachlichen Zustandsanzeige erfolgt durch die Kombination der definierten Clusterbeschreibungen und der Suchbegriffe, verknüpft mit natürlichsprachlichen Verbindungen. Pseudo-Code 3 zeigt den schematischen Aufbau.

```
Ausgabe: „Sie suchen“
Wenn SUCHBEGRIFFE vorhanden
        Ausgabe: „mit den Suchbegriffen SUCHBEGRIFFE“
Für Kategorie Medientypen
        Ausgabe: „nach den Medientypen“ +GESIEBTE CLUSTER
Für Kategorie Fachgebiete
        Ausgabe: „ in den Fachgebieten „ +GESIEBTE CLUSTER
Für Kategorie Sprache
        Ausgabe: „mit den Sprachen“ + GESIEBTE CLUSTER +“.“
Ausgabe: „Ihre Suche ergibt insgesamt ANZAHL TITEL Treffer“.
Ausgabe: „Sie können Sich die gefundenen Titel in einer TABELLE anzeigen lassen“
```

Pseudo-Code 3: Schematischer Aufbau der natürlichsprachlichen Formulierung der Suchanfrage

Alle hier beschriebenen Funktionalitäten wurden komplett implementiert und sind im .NET Compact Framework in der Programmiersprache C# umgesetzt. Das Paket Pocket SieveMap umfasst ca. 1600 Zeilen Code.

4.2.2 Mobile Media Grid

Die Umsetzung der Mobile Media Grid (MMG) ist geprägt von der Umsetzung der Media Grid in MEDIOVIS, mit den in Kapitel 3.2.6 beschriebenen Änderungen. Für die Realisierung dieser tabellenbasierten Ansicht wird eine Klasse, welche von der Basisklasse Panel abgeleitet ist, verwendet. Eine Aufspaltung in mehrere Klassen ist aufgrund des vorhandenen Datenmodells nicht nötig. Die Klasse MMGrid lädt bei Aufruf über den DataManager aus dem Paket DataModel die benötigen Daten zur Darstellung der einzelnen Dokumente. Der Begriff Dokumente wird hier verwendet, weil ein Datensatz im Datenmodell intern so bezeichnet wird und um keine

Verwirrungen zwischen einem Titel der Mediothek und dem Attribut Titel entstehen zu lassen. Die benötigen Daten bestehen aus der durch die Pocket SieveMap (PSM) in Verbindung mit der Textsuche eingeschränkten Ergebnismenge. Dazu wird der aus den Clustern und Suchbegriffen zusammengesetzte SQL-Ausdruck verwendet. Dieser Vorgang kann bei mehreren Hunderten Dokumenten mehrere Sekunden dauern.

Nachdem die Daten geladen sind, wird die tabellenbasierte Darstellung der Dokumente aufgebaut. Alle Zeilen werden dabei direkt auf dem Panel gezeichnet. Im .NET Compact Framework (CF) sind Komponenten zur Darstellung einer Tabelle vorhanden. Diese weisen allerdings den Nachteil auf, dass sie bei großen Datenmengen sehr langsam werden. Des Weiteren wären sehr viele Anpassungen nötig gewesen und manche Anforderungen hätten gar nicht erfüllt werden können.

Beim Aufruf der MMG werden alle Daten aller eingeschränkten Dokumente geladen. Direkt gezeichnet werden immer nur die Dokumente, die auch für den Benutzer auf dem Bildschirm sichtbar sind. Dadurch wird der Bildaufbau recht flüssig und gerade bei Pocket PCs, die über begrenzte CPU-Leistung verfügen, ist dies notwendig um die Anwendung bedienbar zu halten. Nachteile ergeben sich daraus für die Verwendung der Bildlaufleiste. Die vorhandene Komponente im CF wird normalerweise in der Art benutzt, dass sie einem Panel zugewiesen wird und dann automatisch die Funktionalität liefert, das gesamte Panel durchlaufen zu können. Diese Funktionsweise ist mit der direkten Zeichnung der nur sichtbaren Dokumente nicht möglich. Um einen Bildlauf zu realisieren, wurde dafür ein Array aus Integer-Werten angelegt, in dem für jedes Dokument die aktuelle Höhe gespeichert wird. Die aktuelle Höhe eines Dokumentes wird errechnet, indem der benötigte Platz abhängig von den darzustellenden Attributen berechnet wird. Wenn die Aufsummierung aller Höhen geladener Dokumente die Höhe der MMG nicht übersteigt, wird kein Bildlauf benötigt, und die Bildlaufleiste wird auch nicht angezeigt. Wenn mehr Höhe für alle Dokumente benötigt wird, kann die im CF vorhandene Bildlaufleiste für den Bildlauf benutzt werden. Bei einer Bildlaufleiste müssen Minimum- und Maximumwert definiert werden. Diese werden mit 0 und der Aufsummierung aller Dokumenthöhen initialisiert. Dann wird die Methode *ValueChanged()* überschrieben. Diese Methode wird immer dann ausgeführt, wenn die Bildlaufleiste durch den Benutzer verändert wird und gibt einen Wert zurück, der zwischen Minimum und Maximum liegt und

aufgrund der grafischen Position des Bildlaufreglers in der Bildlaufleiste errechnet wird. Anhand dieses Wertes kann man die zu zeichnenden Dokumente berechnen und das Panel wird mit den darzustellenden Dokumenten neu gezeichnet.

Jedes Dokument besitzt die Felder *Focused*, *FullScreen* und *Selected*. Alle drei Felder werden durch den Datentyp bool[23] ausgedrückt. Das Feld *Focused* ist auf wahr gesetzt, wenn das Dokument in Granularitätsstufe 2 durch Klicken mit dem Stylus auf das Dokument gebracht wurde. Bei allen sichtbaren Dokumenten, bei denen das Feld *Focused* wahr ist, werden dann alle in Tabelle 4 definierten Attribute der Stufe 2 dargestellt. Wenn der Benutzer auf ein Dokument, welches sich in Stufe 2 befindet, nochmals klickt, wird das Feld *FullScreen* auf wahr gesetzt und alle Attribute, die für dieses Dokument in der Datenbank vorhanden sind werden dargestellt. Das Feld *Selected* wird auf wahr gesetzt, wenn der Benutzer ein Dokument mittels der Kontrollbox auswählt. Die Kontrollbox wird zu Visualisierung der Auswahl schwarz eingefärbt und das Dokument wird auch in der Normalansicht (Stufe 1) grau hinterlegt.

Abbildung 35 zeigt links die Ansicht der Mobile Media Grid mit mehreren selektierten Titeln und rechts je ein Beispiel für Stufe 2 und Stufe 3 eines Dokuments. Die ausgewählten Dokumente können in einer separaten Ansicht, dem Ausleihkorb, angezeigt werden. Diese Ansicht ist ebenfalls eine Mobile Media Grid. Es werden aber nur die durch den Benutzer ausgewählten Dokumente dargestellt. Dies ist deswegen sinnvoll, weil sich der Benutzer dadurch eine Liste zusammenstellen kann, die er anschließend in der Mediothek ausleihen kann. Die Interaktionsmöglichkeiten von MOBILE MEDIOVIS werden in Kapitel 5 detailliert beschrieben.

Die Mobile Media Grid wurde mit ca. 700 Zeilen Quellcode im .NET Compact Framework in der der Programmiersprache C# umgesetzt und verfügt über alle in Kapitel 3.2.6 beschriebenen Funktionalitäten.

[23] Ein *bool* ist ein Datentyp in der Programmiersprache C#, der die Werte wahr oder falsch annehmen kann.

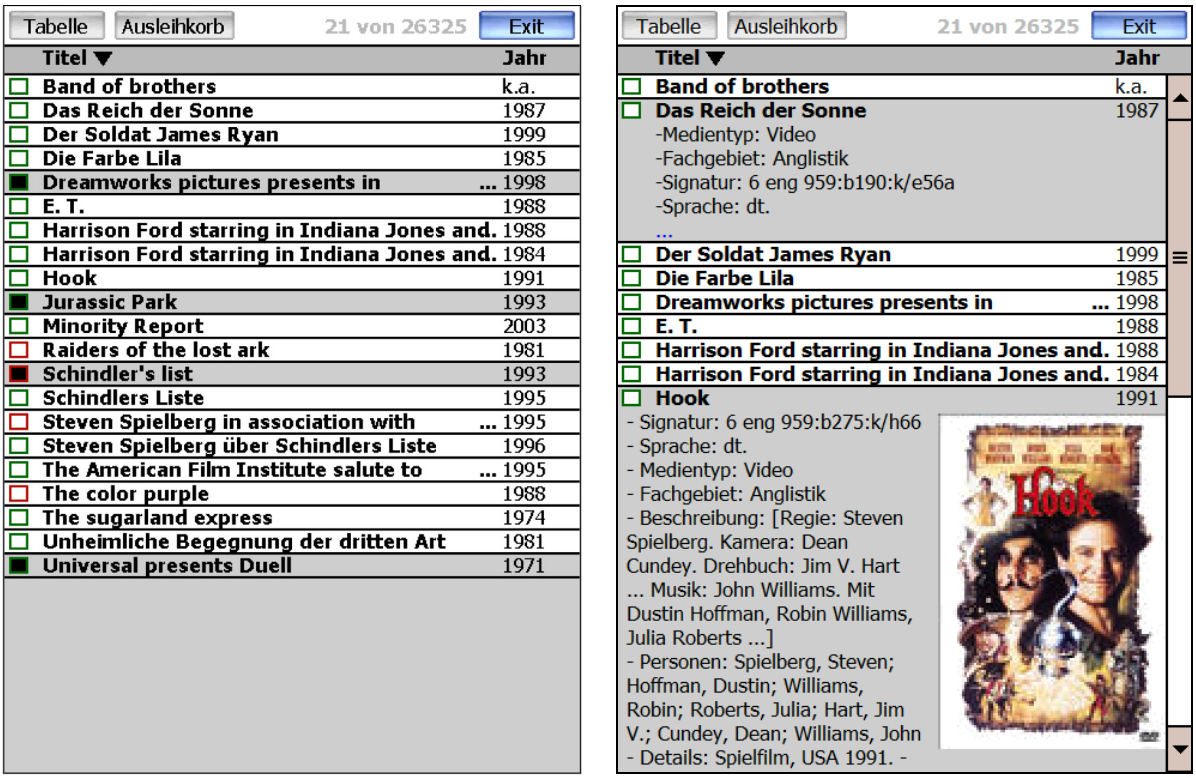

Abbildung 35: Ansicht der Mobile Mediagrid in MOBILE MEDIOVIS

5 Mobile MedioVis

In dem Projekt MOBILE MEDIOVIS ist ein Konzept für ein visuelles Suchsystem entstanden. Gleichzeitig ist eine Anwendung für Pocket PCs entstanden, in der das visuelle Suchsystem voll funktionsfähig umgesetzt ist. Die Anwendung wird ebenfalls mit MOBILE MEDIOVIS bezeichnet. Die Anwendung MOBILE MEDIOVIS befindet sich im beta Stadium. Dies bedeutet, alle Funktionalitäten sind umgesetzt. Allerdings beinhaltet die Software noch kleinere Fehler und die Performanz der Anwendung kann und muss noch verbessert werden. Das hier abgebildete Logo ist aus dem Logo von MEDIOVIS entstanden. In diesem Kapitel wird die Anwendung dokumentiert. Des Weiteren werden die möglichen Interaktionen und die Funktionen von MOBILE MEDIOVIS beschrieben.

5.1 Dokumentation

Dieses Kapitel wird benutzt um MOBILE MEDIOVIS in seiner jetzigen Form zu dokumentieren. Da bereits zahlreiche Abbildungen von MOBILE MEDIOVIS und dem visuellen Suchsystem in dieser Arbeit vorhanden sind, werden im Folgenden die Abbildungen, die für die Erklärung der Interaktion benötigt werden, hier dargestellt. Auf weitere, schon vorhandene Abbildungen wird, wenn nötig, referenziert. In Abbildung 36 ist die Startansicht von MOBILE MEDIOVIS dargestellt. Hier ist es möglich zwischen der Pocket SieveMap 1 (PSM1) und der Pocket SieveMap 2 (PSM2) zu wählen. Aktuell sind beide Visualisierungen umgesetzt und können benutzt werden. Des Weiteren wird beim Aufruf dieses Startbildschirmes überprüft, ob eine Verbindung zur PostgreSQL Datenbank möglich ist. Ist dies aufgrund fehlender aktiver Verbindung zum Internet nicht möglich, wird überprüft, ob eine lokale Datenbank auf dem Pocket PC installiert ist, um auf die Daten zuzugreifen. Sollten beide Datenquellen nicht erreichbar sein, bekommt der Benutzer den Hinweis, dass die Benutzung von MOBILE MEDIOVIS derzeit nicht möglich ist. Dieser Startbildschirm sollte, bei einer realen Benutzung, durch eine kleine Einführung in MOBILE MEDIOVIS ersetzt werden und der Benutzer wird dann direkt zur PSM2 weitergeleitet.

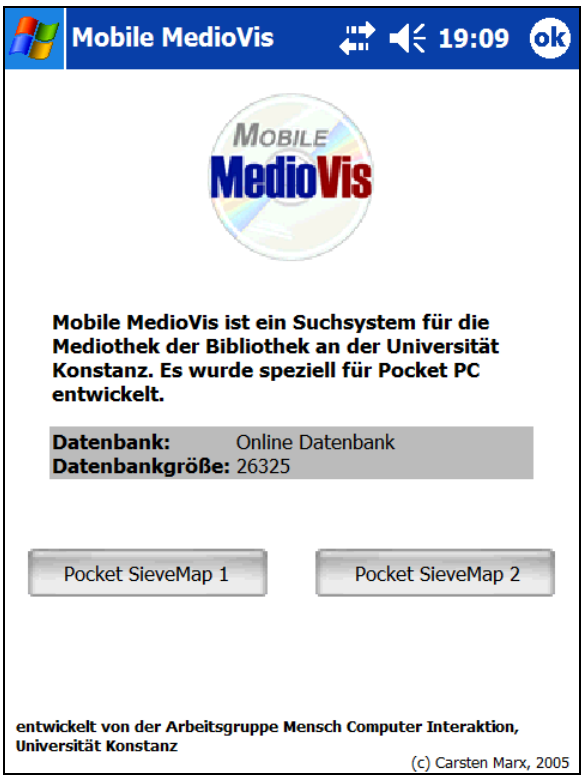

Abbildung 36: Startansicht MOBILE MEDIOVIS

Abbildung 37: Pocket SieveMap 1

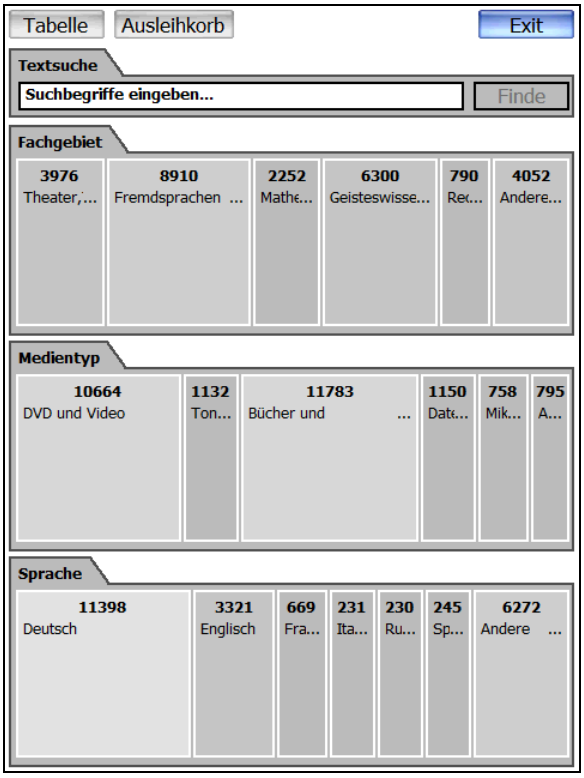

Abbildung 38: Pocket SieveMap 2

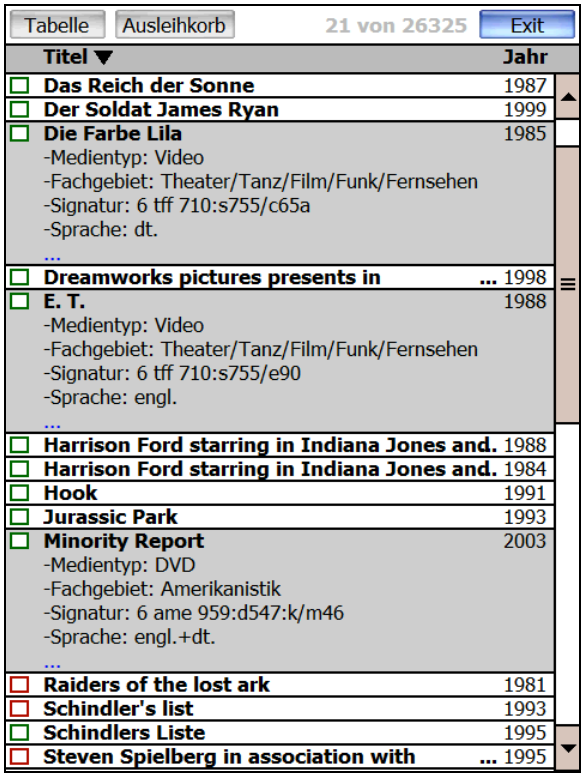

Abbildung 39: Mobile Media Grid mit fokussierten Dokumenten

Abbildung 37 zeigt die PSM1, wie sie dem Benutzer anfangs erscheint. Die Tastatur (Abbildung 5 c) zur Texteingabe erscheint, wenn der Benutzer in das Textfeld zur Eingabe der Suchbegriffe klickt. Beim Absenden der Textsuche wird die Tastatur automatisch wieder ausgeblendet. In Abbildung 38 ist die PSM2 in ihrer Ausgangsposition dargestellt. Die Mobile Media Grid mit einer textuellen Beispielsuche nach *Spielberg* und drei fokussierten Titeln ist in Abbildung 39 gezeigt.

5.2 Interaktion mit dem visuellen Suchsystem

Die Interaktion mit der Pocket SieveMap (PSM) und der Mobile Media Grid wurde so einfach wie möglich gehalten. Alle Eingaben und Aktionen sind mittels des Stylus möglich. Der Benutzer wählt zuerst in der Startansicht (Abbildung 36) die gewünschte PSM aus. Anschließend kann er mit der eigentlichen Suche beginnen. Unabhängig davon, welche Version der PSM der Benutzer auswählt, bleibt die Interaktion dieselbe. Der einzige Unterschied besteht darin, dass bei der ersten Version die Kategorien über die Reiter gewechselt werden müssen. Die Mobile Media Grid ist ebenso bei beiden Versionen gleich. Da die zweite Version die endgültige der PSM ist, wird die Interaktion im Folgenden am Beispiel der zweiten Version der PSM beschrieben.

Abbildung 38 zeigt die PSM, wie sie sich zu Beginn präsentiert. Der Benutzer kann mit dem Stylus die einzelnen Cluster per Drag & Drop zusammenfügen. Dabei tippt der Benutzer auf den gewünschten Cluster und bleibt mit dem Stylus auf dem Bildschirm, während er den Stylus über einen anderen Cluster innerhalb der Kategorie zieht. Der Cluster wird während dieses Vorganges minimal verkleinert, um dem Benutzer zu signalisieren, dass Interaktion stattfindet. Gleichzeitig folgt ein Tooltip mit der Beschriftung *(+)* dem Stylus (Abbildung 40 links). Beim Entfernen des Stylus vom Bildschirm wird der ausgewählte Cluster mit dem unter dem gezogenen Stylus liegenden Cluster verbunden, und die Kategorie wird aktualisiert. Sollte der Benutzer den Stylus nicht über einem Cluster innerhalb der Kategorie loslassen, wird keine Aktion ausgeführt.

Wenn der Benutzer einen Cluster in einer Kategorie auswählen bzw. sieben will, dann klickt er mit dem Stylus auf den gewünschten Cluster und bewegt den Stylus um mindestens vier Pixel in eine Richtung und entfernt den Stylus vom Bildschirm.

Dabei muss er auf dem Cluster bleiben. Dieser Kompromiss war nötig um eine Anzeige der Beschriftungen bei Clustern zu ermöglichen, bei denen die Beschriftung nicht vollständig dargestellt werden kann. Solange der Stylus nicht um vier Pixel in eine Richtung bewegt wird, wird die Beschriftung des Cluster angezeigt (Abbildung 40 rechts). Dabei überlappt die Beschriftung die anderen Cluster.

Abbildung 40: Drag & Drop und Clusterbeschriftungen bei der Pocket SieveMap

Nach dem Sieben eines Clusters werden alle Kategorien mit der entsprechenden SQL-Anfrage aktualisiert und die Ansicht wird komplett neu aufgebaut. Während der Aktualisierung wird am oberen Rand eine Fortschrittsanzeige über die gesamte Breite angezeigt. Durch diese werden die Knöpfe *Tabelle*, *Ausleihstatus* und *Exit* überdeckt, um dem Benutzer zu signalisieren, dass keine Interaktion mit der Anwendung möglich ist. Während der Aktualisierung wird keine Eingabe durch den Benutzer durch die Anwendung akzeptiert. Dies dient der Fehlervorbeugung, da sonst Inkonsistenzen entstehen können. Wenn der Benutzer eine Textsuche starten will, klickt er mit dem Stylus in das Textfeld. Daraufhin erscheint am unteren Rand die Bildschirmtastatur (Abbildung 5 c). Der Benutzer kann seine Suchbegriffe über die Tastatur eingeben und mittels der Enter-Taste auf der Tastatur oder dem *Finde*-

Knopf hinter dem Textfeld die Suche starten. Die neuen Clustergrößen werden berechnet und die gesamte Ansicht wird aktualisiert. Wiederum wird während dieses Prozesses eine Fortschrittsanzeige am oberen Rand eingeblendet und es ist keine Interaktion mit der Anwendung möglich (Abbildung 41 links).

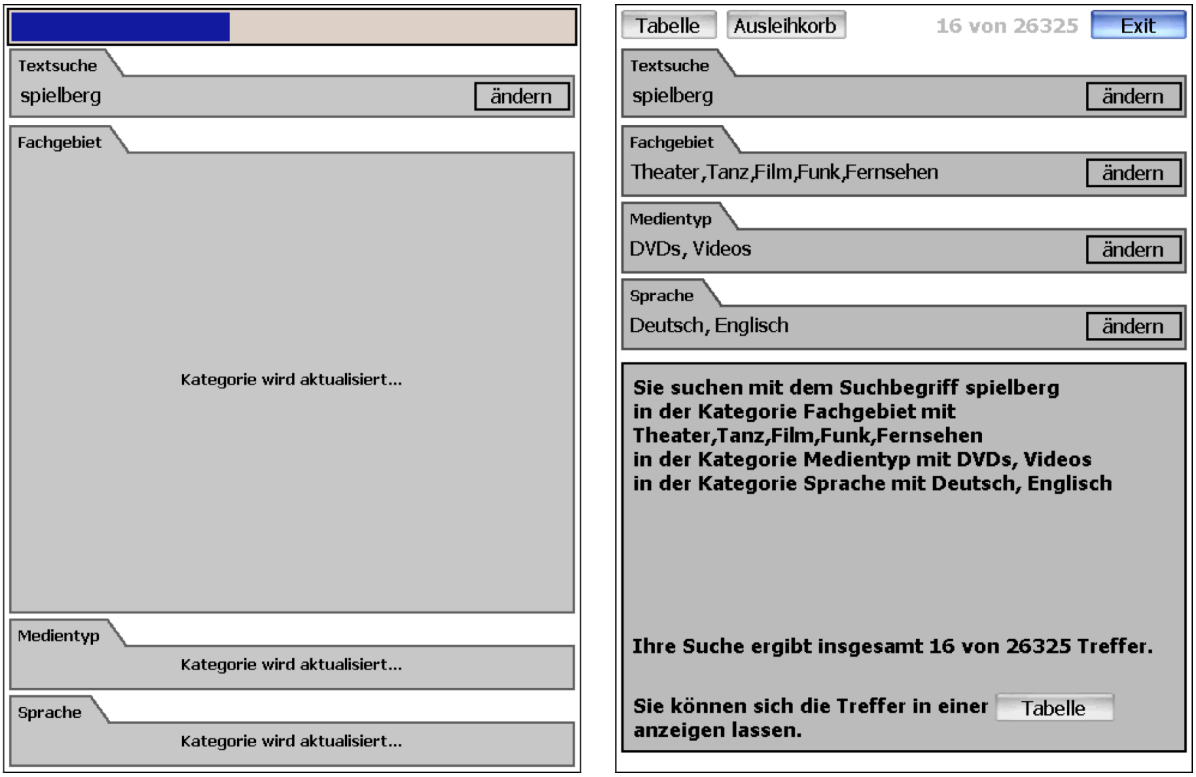

Abbildung 41: Fortschrittsanzeige und natürlichsprachliche Formulierung bei der Pocket SieveMap

Die bisher beschriebenen Interaktionen können in beliebiger Reihenfolge ausgeführt werden. So kann der Benutzer zuerst einige Cluster innerhalb einer Kategorie zusammenfügen und dann diesen Cluster sieben. Danach kann beispielsweise eine Textsuche gestartet werden. Daraufhin kann wieder ein Cluster in einer anderen Kategorie gesiebt werden. Sollte das erste gesiebte Cluster nicht mehr gewünscht sein, kann der Benutzer die Kategorie durch den *ändern*-Knopf zurücksetzen und die ursprünglichen Cluster erscheinen wieder (Abbildung 23 links). Dabei werden die anderen Kategorien mit der aktuellen Treffermenge aktualisiert. Wenn der Benutzer in allen drei Kategorien ein Cluster gesiebt hat, erscheint, unabhängig von der Eingabe einer Textsuche, eine natürlichsprachliche Formulierung der Suchanfrage unterhalb der Kategorien (Abbildung 41 rechts).

In dieser natürlichsprachlichen Formulierung ist ein Knopf eingebettet, mit dem der Benutzer zur Tabellenansicht, der Mobile Media Grid, gelangt. Der Benutzer hat auch die Möglichkeit direkt aus der PSM in die Tabellenansicht zu gelangen. Über den Knopf *Tabelle* in der oberen linken Ecke der Anwendung kann jederzeit direkt die Tabelle aufgerufen werden. Allerdings wird, wenn die Titelmenge mehr als 500 Dokumente enthält, eine Nachricht eingeblendet, die dem Benutzer mitteilt, dass zu viele Treffer angezeigt werden müssten. Gleichzeitig wird der Benutzer freundlich aufgefordert die Menge weiter einzuschränken. Diese Meldung hat einen eher technischen Aspekt. So ist es mit dem derzeitigen Datenmodell nicht möglich mehr als 9000 Dokumente in den Hauptspeicher eines handelsüblichen Pocket PC zu laden. Zudem macht es aber auch keinen Sinn so viele Titel in der Mobile Media Grid anzuzeigen, wenngleich es technisch möglich ist. Außerdem wird die Ladezeit der Daten ab ca. 500 Titeln sehr lang.
Mit dem Knopf *Ausleihkorb* kann der Benutzer sich die bisher ausgewählten Titel anzeigen lassen. Solange er keine Titel ausgewählt hat, ist dieser Knopf inaktiv und dem Benutzer wird beim Klicken mitgeteilt, dass zuerst Titel ausgewählt werden müssen um den Ausleihkorb anzeigen zu können.
In der Mobile Media Grid (MMG) hat der Benutzer seine ausgewählte Titelmenge vor sich (Abbildung 39). Alle dargestellten Dokumente befinden sich in Stufe 1 der Granularitätsstufen. Wenn der Benutzer zu einer Zeile in der MMG mehr Details haben will, kann er mit dem Stylus auf die gewünschte Zeile klicken. Die Granularitätsstufe des entsprechenden Dokumentes wird auf Stufe 2 gesetzt und die Ansicht wird aktualisiert. Wenn noch mehr Details zu einer sich in Stufe 2 befindenden Zeile gewünscht sind, kann dies durch nochmaliges Klicken mit dem Stylus auf die gewünschte Zeile geschehen. Daraufhin wird das Dokument in Stufe 3 gebracht und es werden alle vorhandenen Informationen zu dem Dokument dargestellt (Abbildung 39). Sollte der Benutzer zu einem Dokument, welches sich in Stufe 3 befindet, die Informationen nicht mehr benötigen, wird durch Klicken in die Zeile das Dokument in Stufe 1 gebracht. Diese beschriebenen Interaktionen können in der Mobile Media Grid mit beliebig vielen Zeilen gemacht werden. Sollten so viele Dokumente bzw. Informationen dargestellt sein, dass eine Bildlaufleiste benötigt wird (Abbildung 39), kann der Benutzer mit dieser rauf- und runterscrollen, bis er bei der gewünschten Zeile bzw. Information ist. Die verschiedenen Granularitätsstufen der

jeweiligen Dokumente werden dabei nicht verworfen, sondern bleiben erhalten. Die Mobile Media Grid ist anfangs absteigend nach dem Attribut Titel sortiert. Durch einen Klick auf die Beschriftung Titel der Tabelle kann die Sortierung umgekehrt werden. Durch einen Klick auf die Beschriftung Jahr der Tabelle kann die MMG nach dem Attribut Jahr sortiert werden, dabei wird beim ersten Klick absteigend und bei einem weiteren Klick aufsteigend sortiert. Auch hier bleiben die einzelnen Granularitätsstufen der Dokumente erhalten.

Die Auswahl einer Zeile ist jederzeit, egal in welcher Granularitätsstufe sich das Dokument befindet, möglich. Der Benutzer muss dazu auf die Kontrollbox am Beginn der Zeile klicken. Daraufhin wird das Dokument als selektiert markiert und die Kontrollbox wird schwarz ausgefüllt. Wenn der Benutzer mindestens ein Dokument ausgewählt hat, wird der Knopf *Ausleihkorb* aktiv. Die Dokumente können jederzeit, sowohl in der MMG als auch im Ausleihkorb, wieder abgewählt werden. Dies geschieht durch nochmaliges Klicken auf die Kontrollbox. Dabei wird die Kontrollbox wieder mit der Hintergrundfarbe der MMG gefüllt und das Dokument wird als nicht selektiert markiert (Abbildung 42 links).

Sofern mindestens ein Dokument als selektiert markiert ist, kann der Benutzer sich über den Knopf *Ausleihkorb* alle ausgewählten Dokumente anzeigen lassen (Abbildung 42 rechts). Die ausgewählten Dokumente werden dabei ebenfalls in der MMG dargestellt. Der Benutzer kann nach Belieben zwischen dem Ausleihkorb und der Tabelle hin- und herschalten.

Es besteht für den Benutzer immer die Möglichkeit von der Mobile Media Grid wieder in die Pocket SieveMap zurück zu gelangen. Dazu muss lediglich der Knopf *Tabelle* eine weiteres Mal geklickt werden. Die bisherige Auswahl der Cluster oder eine getätigte Textsuche bleiben dabei erhalten. Dies bedeutet, die PSM ist in derselben Anordnung wie vor dem Anzeigen der MMG. Dadurch kann der Benutzer, wenn das Ergebnis nicht das Gewünschte war, sofort direkt die Suchanfrage umformulieren und sich erneut die Treffer anzeigen lassen. Diese Funktionalität ersetzt praktisch eine Historiefunktion. Der Benutzer kann seine Änderungen direkt und einfach wieder rückgängig machen.

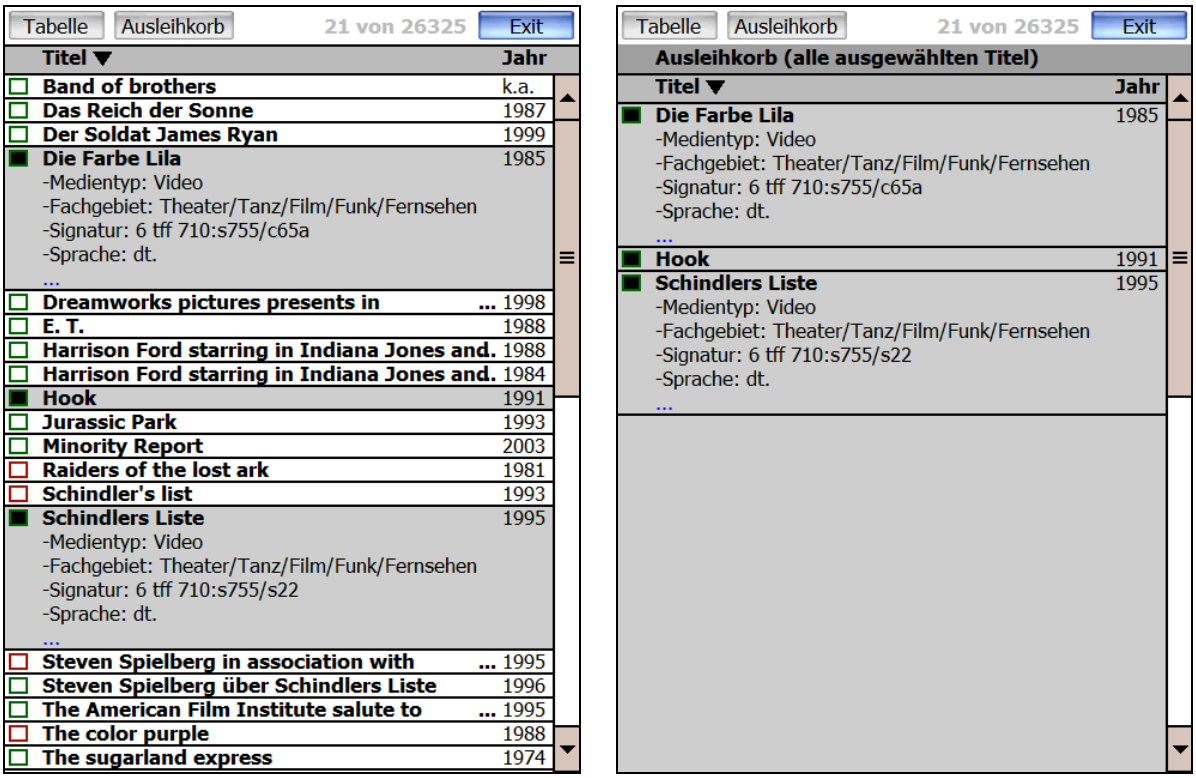

Abbildung 42: Selektion in der Mobile Media Grid und der Ausleihkorb

Die Anwendung MOBILE MEDIOVIS kann jederzeit über den *Exit*-Knopf in der rechten oberen Ecke durch den Benutzer verlassen werden. Dadurch werden alle gesiebten Cluster und ausgewählten Dokumente verworfen und die Anwendung wird beendet.

6 Schlussbetrachtung

Im Rahmen des Projekts MOBILE MEDIOVIS, welches in dieser Arbeit beschrieben wurde, ist ein Konzept für ein effektives Filterwerkzeug und eine übersichtliche Titeldarstellung entstanden. Diese beiden Komponenten wurden zu einem visuellen Suchsystem vereint und als reale Anwendung auf einem Pocket PC umgesetzt. Die Umsetzung brachte einige Erkenntnisse und Erfahrungen in .Net Compact Framework mit sich, die in Kapitel 6.1 berichtt werden. Ein Ausblick auf weitere Funktionen und Möglichkeiten der Pocket SieveMap und der Mobile Media Grid und ein Fazit werden am Ende dieses Kapitels die Arbeit abschließen.

6.1 Erfahrungen mit dem .NET Compact Framework

Neben der Entwicklung und Konzeption eines visuellen Suchsystems für die Mediothek war ein Großteil der Projektarbeit damit verbunden die entstandenen Konzepte der Pocket SieveMap und der Mobile Media Grid im .NET Compact Framework (CF) umzusetzen. Hierbei sind einige Erkenntnisse entstanden, die es Wert sind, kurz beschrieben zu werden. Wie bereits in Kapitel 2.3.4 beschrieben, handelt es sich beim CF um eine Teilmenge des kompletten .NET Framework von Microsoft. Ein großer Vorteil des .NET Framework ist die gute Dokumentation durch Microsoft [CF-1] und die aktiven Newsgroups [CF-2], die zur Recherche und Problemlösung sehr hilfreich sind. Des Weiteren wirkt die von Microsoft vertriebene Entwicklungsumgebung, Visual Studio .NET 2003 [CF-3], sehr ausgereift und verfügt über viele nützliche Tools zur Quellcodevervollständigung und zur Fehlersuche. Es kann auch ohne einen realen Pocket PC mit einem Emulator[24] entwickelt werden. Es empfiehlt sich allerdings einen Pocket PC zur Verfügung zu haben, da der Emulator sehr langsam wirkt und das Entwickeln mit einem realen Gerät sehr viel komfortabler ist. Dies begründet sich durch die Tatsache, dass der Emulator mit der Maus bedient wird und direkt auf dem Rechner läuft. Dies bedeutet, das Gerät kann nicht, wie für einen Pocket PC üblich, direkt in der Hand gehalten werden und die Interaktion wird nicht direkt auf dem Bildschirm mittels des Stylus durchgeführt.

[24] Ein *Emulator* bildet ein anderes System nach. Das nachgebildete System erhält die gleichen Daten, führt die gleichen Programme aus und erzielt die gleichen Ergebnisse wie das originale System.

Durch die Verwendung des Java Language Conversion Assistant [CF-4] konnten große Teile des Datenmodells von MEDIOVIS übernommen werden. Ohne diese Automatisierung wäre es nicht möglich gewesen, die Anwendung MOBILE MEDIOVIS so weit voranzutreiben. Dieser Assistent, welcher sich nach Aussage von Microsoft im beta Stadium befindet, wirkt sehr ausgereift. Die Teile, die nicht komplett portiert werden konnten, waren meist übersichtlich dokumentiert.

Das CF weist allerdings einige Schwächen auf. So musste für die benötigte Verbindung zu der PostgreSQL Datenbank das Produkt PostgreSQLDirect .NET Data Provider [CF-5] beschafft und eingebunden werden, weil Microsoft im CF keine eigene Möglichkeit dafür bereitstellt. Auch für die Offline Version musste ein externes Produkt benutzt werden. eSQL [CF-6] ermöglich es eine SQL-Datenbank auf einem Pocket PC zu installieren und über Standard SQL darauf zuzugreifen.

Um Bildschirmfotos von Anwendungen zu machen, eignet sich die Anwendung dot Pocket [CF-7] hervorragend. Mit ihr ist es auch möglich ganze Interaktionssequenzen aufzuzeichnen. Allerdings wird dadurch die Ausführungsgeschwindigkeit des Pocket PCs merklich langsamer. Ein leider vorhandenes Manko der Entwicklungsumgebung Visual Studio .NET ist es, dass eine Anwendung nicht in einen Emulator gepackt und auf einem anderen Rechner vorgeführt werden kann. Um den Emulator zu nutzen muss die komplette Entwicklungsumgebung installiert werden.

Abschließend betrachtet ist es mit dem CF und der Entwicklungsumgebung Visual Studio .NET relativ komfortabel möglich für einen Pocket PC zu entwickeln und in Zukunft werden wohl viele Funktionalitäten, die bisher noch nicht möglich sind, umgesetzt werden.

6.2 Ausblick auf weitere Funktionen und Möglichkeiten

Visuelle Suchsysteme für kleine Bildschirme, insbesondere auch für mobile Geräte, stehen, was die Forschung betrifft, noch am Anfang ihrer Reise. Visualisierungen wie die DateLens oder der Mobile Liquid 2D Scatter Space aus jüngster Vergangenheit zeigen, dass es möglich ist, effektive Visualisierungen, die effizientes Arbeiten bzw. Suchen ermöglichen, auf kleinen Bildschirmen umzusetzen. Im Projekt MOBILE MEDIOVIS ist ein visuelles Suchsystem entstanden, mit dem eine effektive Suche und Exploration in der Mediothek der Bibliothek möglich ist. Alle Basisfunktionalitäten von

MEDIOVIS sind auch in MOBILE MEDIOVIS enthalten. Des Weiteren steht durch die Portierung des Datenmodells in das .NET Compact Framework nun eine Basisanwendung zur Verfügung, mit der weitere Visualisierungen und Anwendungen im Mediothek-Kontext entstehen können. Die Anwendung MOBILE MEDIOVIS kann natürlich um Funktionalitäten erweitert werden. So sind die Anzeige des Standortes und ein Navigationssystem, um die ausgewählten Titel schneller in der Mediothek zu finden, denkbar. An einem entsprechenden Projekt wird bereits in der Arbeitsgruppe Mensch Computer Interaktion gearbeitet. Auch die Einbindung von weiteren externen Daten, zum Beispiel Trailer bei DVDs oder Musikhörproben bei CDs, könnte vorgenommen werden um dem Benutzer mehr Informationen über die Titel anbieten zu können. Des Weiteren könnten während der Datenaktualisierung die Übergänge beim Sieben durch Animation visuell unterstützt werden. Zuvor sollte jedoch ein vollständiger Usability Test durchgeführt werden, um weitere Verbesserungen an MOBILE MEDIOVIS vorzunehmen.

6.3 Fazit des Projektes

In dieser Arbeit wurden alle relevanten Schritte, die zur Entwicklung eines visuellen Suchsystems für die Mediothek auf einem Pocket PC nötig waren, beschrieben. Die technischen Rahmenbedingungen zu definieren erwies sich im Nachhinein als sehr sinnvoll, da einige Unterschiede bezüglich der Hard- und Software gravierende Auswirkungen auf die Konzeption der Pocket SieveMap und der Mobile Media Grid hatten. Die richtige Auswahl der Implementierungsplattform, die sich teilweise durch widersprüchliche Informationen schwierig gestaltete, war entscheidend dafür, dass MOBILE MEDIOVIS als reale Anwendung entstanden ist. Durch die Analyse bereits bestehender Visualisierungen für Bildschirme mit begrenzter Größe bzw. Auflösung entstanden Ideen für ein Konzept, welches in Brainstorming-Sitzungen diskutiert und verfeinert wurde. Daraus entstand durch die Kombination der Pocket SieveMap und der Mobile Media Grid eine geeignete Herangehensweise um komfortabel auf einem Pocket PC die Mediothek zu durchsuchen und die Suchanfrage über direkte Manipulation dynamisch zu erstellen. Parallel zu der Entwicklung des Konzeptes wurde mit der Portierung des Datenmodells begonnen. Dies gestaltete sich teilweise, auch mit Hilfe von automatischen Tools, schwierig, weil einige Konstrukte aus Java

in C# angepasst werden mussten, und keine Erfahrung mit der Implementierungsplattform .NET Compact Framework in der Arbeitsgruppe Mensch-Computer Interaktion vorhanden war. All diese Schwierigkeiten wurden dennoch überwunden und durch die Entscheidung, ein visuelles Suchsystem für die Mediothek auf einem Pocket PC zu entwickeln, wurden neue und wertvolle Erkenntnisse gewonnen, die für zukünftige Projekte im Bereich mobile Geräte sicherlich von Nutzen sind. Da die Pocket SieveMap nicht nur auf Pocket PCs seinen Reiz hat, wurde das Konzept der SieveMap in die Java Anwendung MEDIOVIS experimentell übernommen und steht somit auch auf dem Desktop als effektives Filterwerkzeug den Benutzern der Mediothek bald testweise zur Verfügung. Somit schließt sich der Kreis der Visualisierungen für eine visuelle Benutzerschnittstelle für die Mediothek und die SieveMap wird durch Kombinationen mit anderen visuellen Filterwerkzeugen sinnvoll den Suchprozess verbessern.

Aktuelle Informationen zu dem Projekt können der Projektseite entnommen werden:
http://hci.uni-konstanz.de/research/projects/mobilemediovis

Abbildungs- und Tabellenverzeichnis

Bibliographie

[Furnas1986] Furnas, G. W.; **Generalized Fisheye Views.** , CHI'86 ACM Press, pp. 16-23

[Shneiderman1991] Ben Shneiderman, **Tree Visualization with Tree-maps: A 2-d space filling approach**, *ACM Transactions on Graphics,* vol. 11, 1 (Jan. 1992) 92-99. HCIL-91-03 , CS-TR-2645 , CAR-TR-548

[Ahlberg1992] Ahlberg, C., Williamson, C., Shneiderman, B. (Sept. 1991), **Dynamic queries for information exploration: An implementation and evaluation,** *ACM CHI `92 Conference Proc.* (Monterey, CA, May 3-7, 1992) 619-626. Also Sparks of Innovation in Human-Computer Interaction, Shneiderman, B., Ed., Ablex (June 1993) 281-294. HCIL-91-11 , CS-TR-2763 , CAR-TR-584

[Ahlberg1993] Ahlberg, C. and Shneiderman, B., **Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays,** *ACM CHI '94 Conference Proc.* (Boston, MA, April 24-28, 1994) 313-317

[Frohlich1993] Frohlich, David M.; **The history and future of direct manipulation**, Behaviour & Information Technoloy 12, 6 (1993), 315-329

[Perlin1993] Ken Perlin, David Fox, **Pad: an alternative approach to the computer interface,** International Conference on Computer Graphics and Interactive Techniques, Proceedings of the 20th annual conference on Computer graphics and interactive techniques, Pages: 57 – 64, 1993

[Ahlberg1994] Ahlberg, C.; Shneiderman, Ben: **Visual Information Seeking Using the FilmFinder.** *In: ACM CHI'94 Conference Companion, 433-434, 1994.*

[Krause1995] Jürgen Krause, **Das WOB-Modell,** IZ-Arbeitsbericht Nr. 1, Dezember 1995, 77 Seiten

[Shneiderman1997] Shneiderman, Ben: **Direct Manipulation for Comprehensible, Predictable and Cotrollable User Interfaces**, *Proceedings of IUI97, 1997 International Conference on Intelligent User Interfaces*, Orlando, FL, January 6-9, 1997, 33-39. HCIL-97-01

[Card99] Card, Stuart / Mackinlay, Jock / Shneiderman, Ben, **Readings in Information Visualization**. Morgan Kaufmann Publishers, San Fracisco

[Bederson2003] Benjamin B. Bederson, Aaron Clamage; **A Fisheye Calendar Interface for PDAs: Providing Overviews for Small Displays**, CHI 2003 ACM 1-58113-637-4/4/03/0004

[Bederson2004] Benjamin B. Bederson, Aaron Clamage, Mary P. Czerwinski. George G. Robertson; **DateLens: A Fisheye Calendar Interface for PDAs**, ACM Transactions on Computer-Human Interaction, Vol 11, No. 1, März 2004, p. 90-119

[Waldeck2003] Carsten Waldeck, Daniel Hess, Dirk Balfanz, **Mobile Liquid Information Spaces – Maximierung der Informationsdichte für visuelle Echtzeitsuche auf kleinen Screenflächen mit Hilfe von Transparenz und Wühlfunktionalität**, G. Szwillus, J. Ziegler (Hrsg.): Mensch & Computer 2003: Interaktion in Bewegung. Stuttgart: B. G. Teubner, 2003, S. 435-438

[Grün2004] Christian Grün, **Entwicklung eines visuellen Metadaten-Browsers für die Mediothek Konstanz,** Masterarbeit im Fachbereich Informatik und Informationswissenschaften Universität Konstanz, Dezember 2004

[Grün2005] Christian Grün, Jens Gerken, Hans-Christian Jetter, Werner König, Harald Reiterer; **MedioVis - a User-Centred Library Metadata Browser.** Eingereicht zur ECDL 2005

[Büring2005] Thorsten Büring, Harald Reiterer; **ZuiScat – Querying and Visualizing Information Spaces on Personal Digital Assistans**, Eingereicht zur MobileHCI 2005

Internet Quellen

(alle URLs wurden 19.04.2005 überprüft und waren erreichbar)

[Wikipedia2005] **Microsoft Pocket PC**,
http://de.wikipedia.org/wiki/Microsoft_Pocket_PC

[JavaFAQ] **Java on PocketPC,** http://www.vikdavid.com/mobile/

[J2ME] **Java 2 Platform, Micro Edition**,
URL: http://java.sun.com/j2me/docs/j2me-ds.pdf

[DevTools] **Introduction to Development Tools for Windows Mobile-based Devices,**
http://msdn.microsoft.com/mobility/windowsmobile/default.aspx?pull=/library/en-us/dnppcgen/html/devtoolsmobileapps.asp

[CF-1] **.NET Framework Dokumentation**,
http://msdn.microsoft.com/smartclient/understanding/netcf/

[CF-2] **Microsoft Newsgroups**,
http://msdn.microsoft.com/newsgroups/default.aspx

[CF-3] **Microsoft Visual Studio**, http://msdn.microsoft.com/vstudio/

[CF-4] **Java Language Conversion Assistant 3.0 beta**,
http://msdn.microsoft.com/vstudio/downloads/tools/jlca/30Beta/

[CF-5] **PostgreSQLDirect .NET Data Provider**,
http://crlab.com/pgsqlnet/

[CF-6] **Vieka eSQL**, http://www.vieka.com/esql.htm

[CF-7] **dot Pocket**, http://www.dotpocket.com/