Adaption von Webseiten

Methoden und Ansätze

Ali Ikinci
ali@ikinci.de

November 2005

Mobile Business Seminar

am Lehrstuhl für Praktische Informatik IV
Prof. Dr. W. Effelsberg
Betreuer: Stephan Kopf
UNIVERSITÄT MANNHEIM

Inhaltsverzeichnis

1 Einführung

Mit zunehmender Verbreitung von Personal Digital Assistants (PDA), Smart Phones und anderen heterogenen mobilen Geräten die Zugriff auf das Internet haben, entstehen neue Probleme und Fragestellungen für die noch keine zufriedenstellende Lösung besteht. Verschiedene Systeme haben jeweils Teillösungen für Teilprobleme gestaltet, die jeweils mehr oder weniger zufriedenstellend arbeiten [1][2][3][4]. Zentrales Thema dieser Arbeit ist die Frage: „Wie kann man Webseiten die für einen Personal Computer (PC) entwickelt worden sind, auf mobilen heterogenen Geräten darstellen, die größere Beschränkungen im Hinblick auf Input/Output Kapazitäten haben?“ Dabei wird kein eigenes System entworfen , sondern vielmehr allgemein anerkannte und angewandte Methoden und Ansätze für die Lösung dieser Frage präsentiert und einen Überblick über diese gegeben. Für eine einfachere Darstellung wird im Folgenden der hypothetische Anpassungsalgorithmus als „Adaptor“ bezeichnet.

2 Grundlegende Theorie und Annahmen

Im Internet werden Inhalte hautsächlich für PCs mit einer Mindestauflösung von 640x480 Bildpunkten und oft sogar für eine Standardauflösung von 1024x768 Bildpunkten entwickelt. Eine einfache Skalierung des Inhalts würde zu einem Verhältnis von 4- bis 100- facher Verkleinerung des Inhalts führen und zur Folge haben, dass der Inhalt unnavigierbar, unästhetisch oder gar unlesbar werden würde.

Im Vergleich zu normalen PCs haben heterogene Geräte spezielle Eigenschaften in Bezug auf:

- Bildschirmgröße
  - von 1 Zoll bis zu 4 Zoll Bilddiagonale, weshalb eine Anpassung der Schriftgröße zur besseren Lesbarkeit benötigt wird.
- Bildschirmauflösung
  - von einer mehrzeiligen Textdarstellung bis zu einer Halb-VGA Auflösung von 640x240 Bildpunkten.
- Farbtiefe
  - von Monochrom bis zu 4094 Farben, welche ein sinnvolles Downscaling des Farbraumes benötigt.
- Rechenkapazitäten
  - sind stark begrenzt und bringen große Restriktionen für eine Anpassung auf dem Client mit sich, da eventuell keine Anpassung in Echtzeit erfolgen kann.

- Arbeitsspeicher und Festwertspeicher
  - sind stark begrenzt und können Probleme bei der Aufnahme der ursprünglichen Webseite bringen, da das Gerät diese weder verarbeiten, noch abspeichern kann.
- Software
  - eine Beschränkung auf das Wesentliche bringt nur eine Grundausstattung im Hinblick auf Software und vor allem auf nutzbare Bibliotheken auf den Clients mit sich. Bei der Verwendung von verschiedenen Bild-, Audio- und Videoformaten erfordert dies eine sinnvolle Anpassung oder Filterung der Ursprungsdaten auf die Clientbibliotheken.
- Bandbreite
  - in aktuellen GSM, UMTS und WLAN Netzen ist die Bandbreite oft kleiner als in herkömmlichen drahtgebunden Netzen wie ADSL oder LAN. Dieser Umstand kann sich negativ auf das Surfverhalten auswirken, falls eine Anpassung auf dem Client geschehen soll und die Seite somit komplett vom Internet geladen werden muss.
- Kosten für die Bandbreite
  - Funknetze werden oft mit Zeit- oder Volumentarifen abgerechnet, wobei eine Minimierung des Datenverkehrs wünschenswert wäre.
- Interaktionsmöglichkeiten
  - eine fehlende Maus und eine fehlende Tastatur welche Bestandteile jedes modernen PC's sind, benötigen eine sinnvolle Reduzierung der Benutzerinteraktion für die heterogenen Geräte.
- Nutzerpräferenzen
  - können eine optimale Anpassung auf den spezifischen Client und die Auswahl der Inhalte erleichtern.

Wenn in dieser Arbeit von Anpassung und Adaption die Rede ist, dann geht es hauptsächlich um die Anpassung von statischem (X)HTML-Code [5][6]. Dieser kann vom Webserver dynamisch erzeugt worden sein. Uns interessiert hier nur die statische Antwort des Webservers auf eine HTTP-Anfrage.

Dynamische Multimediainhalte wie Java-Applets, Flash-Animationen, Audio/Video-Daten oder Bilder sollen nicht Bestandteil dieser Arbeit sein, sie werden teilweise in anderen Arbeiten behandelt [7]. Ausserdem beinhalten die vorgestellen Methoden auch keine Lösung für DHTML, also, HTML mit umfangreicher Java-Script Anwendung für die Gestaltung, welches ignoriert wird.

Die Adaptionsschritte gestalten sich dabei wie folgt – wobei einige auch ausgelassen oder erweitert werden können. Sie werden in den nachfolgenden Kapiteln ausführlicher dargestellt:

- Analyse der Struktur
  - Bestimmung der Inhaltsblöcke und der Hierarchie
  - Tabellenanalyse
  - Imagemap Analyse
- Analyse des Inhalts
  - Textanalyse
- Anpassung von Webseiten
  - Textanpassung bzw. Textzusammenfassung
  - Medienanpassung (wird in dieser Arbeit nicht betrachtet)
  - Thumbnail-Generierung
  - Anpassung der Abfolge von zusammengesetzten Seiten (Page-Splitting)
- Zusammensetzung von Webseiten
  - Automatische Neupositionierung
  - Berücksichtigung des Benutzerverhaltens
  - Berücksichtigung von Benutzerpräferenzen (wird in dieser Arbeit nicht betrachtet)

Die meisten Adaptionsysteme sind HTTP-Proxy basiert. Die Anpassung kann bei dem Client selbst, bei einem dazwischengeschalteten Proxy oder durch Kombination von beiden geschehen.
Der Proxy (Adaptor) fängt die Anfrage des Clients nach Webseiten ab, holt sich die Anfrage, führt ihre Adaption durch und sendet diese dem Client. Diese Inhaltsadaption wird oft auch als *transcoding* bezeichnet. Dabei nennt man im Allgemeinen den Proxy, der als Eingang zum angepassten System dient, auch ein „Portal“.
Im Folgenden ist eine schematische Darstellung für eine Adaption beim Anbieter (Abb. 1) und eine Adaption beim Anwender (Abb. 2) angegeben. Hierbei ist zu beachten, dass der angedeutete Kreis ein Hinweis darauf ist, dass der Proxy Bestandteil des Webservers bzw. des Clients sein kann, aber nicht sein muss. Alle vier Kombinationen sind denkbar.

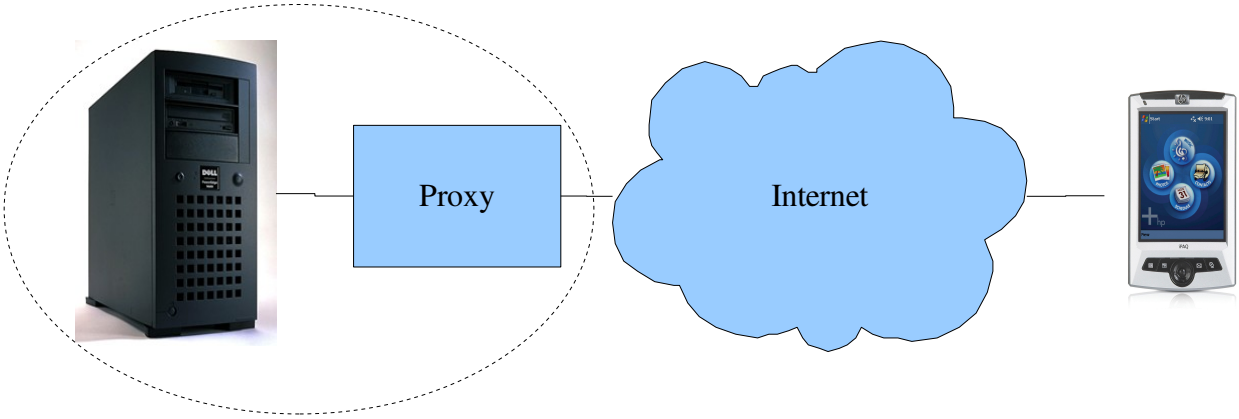

Abbildung 1: Adaption beim Anbieter

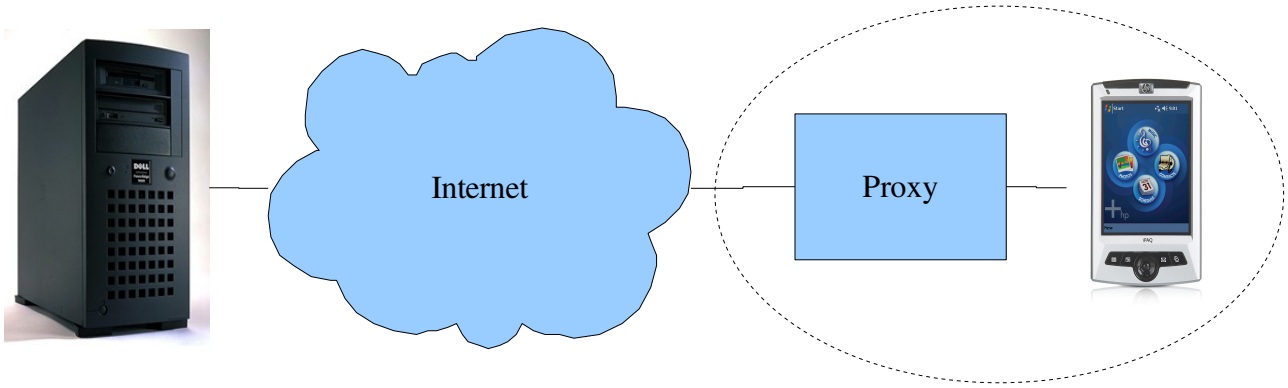

Abbildung 2: Adaption beim Anwender

Ein alternativer Ansatz wäre die automatische Adaption beim Server [8], der mit einem Framework oder Web Content Management System arbeitet und aus XML-Daten mit XSL-Templates eine automatische Anpassung für die verschiedenen Geräte sicherstellt. Auf dieses Thema gehen wir im Kapitel 5 noch ausführlicher ein.

3 Inhaltsanalyse

Bei der Inhaltsanalyse werden Informationen aus der Webseite extrahiert, die für die Adaption benötigt werden. Eine Webseite kann man im Allgemeinen auch als eine Darstellung von Inhalten auffassen. Im Kapitel „Alternative Ansätze“ wird gezeigt, dass man Inhalte von der Darstellung trennen und sich so eine Inhaltsanalyse ersparen kann. Über die Jahre wurden viele Heuristiken entwickelt, mit denen man im Internet Data Mining [9] betreiben kann. Die folgenden Methoden haben von diesen Bemühungen profitiert.

3.1 Bestimmung der Struktur einer Webseite

Bei der Analyse wird die semantische Struktur einer vorhandenen Seite analysiert. Diese Struktur ist eine hierarchische Repräsentation der Webseite, in der jeder Knoten eine Gruppe von Objekten darstellt. Knoten an höheren Stellen in der Hierarchie beinhalten oft mehrere Objekte, während

Knoten an unteren Stellen eher eine oder wenige Objekte beinhalten, die Informationseinheiten darstellen. Diese Informationseinheiten können verwaltet und individuell angezeigt werden. Sie werden im Folgenden Inhaltsblöcke genannt.

3.1.1 Bestimmung der Inhaltsblöcke

Die Identifizierung von Inhaltsblöcken geschieht iterativ (Abb. 3). Am Anfang wird die gesamte Seite als ein Block aufgefasst. Bei jeder Iteration wird der Block in kleinere aufgeteilt. Am Ende bleibt eine Anzahl von atomaren Inhaltsblöcken (die Blattknoten) erhalten, die nicht mehr aufgeteilt werden können und die für die anschließende Anpassung benötigt werden [10].

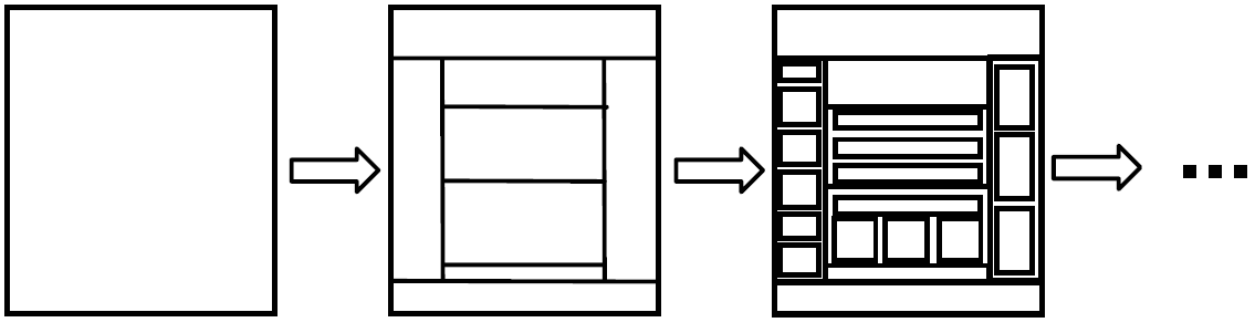

Abbildung 3: Identifizierung von Inhaltsblöcken als iterativer Algorithmus [10]

Der DOM-Baum [11] wird vom `<BODY>` bis zu seinen Blättern durchquert um angemessene Knoten zu finden und sie in einem Inhaltsblock zusammenzufassen. Dabei wird jeder Block nach Kopfzeile, Fußzeile, linker Rand, rechter Rand und atomarem Inhalt durchsucht und aufgeteilt.

Die Detektion der Kopfzeile erfolgt mit einem Schwellwert (z.B. Zehn Prozent der Höhe) für die obere Hälfte des Blocks. Passt das Objekt in den oberen Bereich, dann wird es als eine Kopfzeile erkannt. Analog werden auch Fußzeile, linker Rand und rechter Rand erfaßt. Falls der verbleibende Rest klein genug ist, was wiederum mit einem Schwellwert getestet werden kann, wird er nicht mehr weiter aufgeteilt und stellt einen atomaren Inhaltsblock dar. Falls der Rest zu groß ist, wird er als weiterer Inhaltsblock gewertet und weiter aufgesplittet. Die Schwellwerte für die Detektion werden in langwierigen Versuchen als Erfahrungswerte festgelegt und können je nach Webseite mehr oder weniger gut funktionieren. In einem weiteren Schritt werden die gefunden Blöcke dann nach Begrenzungen abgesucht und weiter aufgesplittet.

3.1.2 Tabellenanalyse

Hier werden zusätzliche Begrenzungen identifiziert, die für die weitere Aufteilung verwendet

werden [12]. Sie können aus drei Arten bestehen:

- <HR> wird oft als ein horizontaler Begrenzer verwendet, welcher eine horizontale Linie darstellt.
- Jeder <TR>- und <TD>-Block im <TABLE>-*Container* kann als eine Begrenzung angesehen werden.
- Bilder können auch als Begrenzung dienen wenn sie charakteristische Eigenschaften haben die einen besonders großen Unterschied im Verhältnis von Breite und Höhe eines Bildes aufweisen (z.B. 50:1).

3.1.3 Imagemap-Analyse

Eine Imagemap ist ein Konstrukt in HTML bei dem in einem normalen Bild Bereiche definiert werden können, die eine besondere Aktion oder einen Link beinhalten können. Imagemaps sind im Allgemeinen recht große Bilder und sind deshalb für die direkte Anzeige bei den Clients nicht geeignet. Bei der Imagemap-Analyse werden die Links in einer Imagemap als eine Indexseite für einen weiteren Inhaltsblock aufgefasst und eine Indexseite für diese Links generiert. Die Benennung der Links wird aus dem <ALT>-*tag* der Imagemap, oder falls existent, vom Link selbst extrahiert [13]. Die Indexseite wird in der Gesamthierarchie als Blatt mit aufgenommen.

3.2 Textanalyse

Texte können unter Umständen sehr lang sein und vertikales Scrollen auf dem Bildschirm erfordern. Um eine bessere Übersicht und eine leichtere Navigation zu erreichen, kann man sie analysieren und kompakte Inhaltsangaben über sie machen. Bei der Textanalyse [14] werden statistische Informationen aus den atomaren Inhaltsblöcken gesammelt, die für die spätere Textzusammenfassung benötigt werden. Hierfür spielen zwei Objekte eine Bedeutung.

- *Keywords* sind wichtige Wörter, die im Text vorkommen und stichwortartig den Inhalt des Textes wiedergeben. Sie werden erfasst durch ihre *importance*, welche durch die Anzahl des Auftretens eines Wortes im Text und im Gesamtdokument bestimmt wird.
  Intuitiv lässt sich festhalten, dass ein Wort in einem Text wichtig ist, wenn es in diesem oft aber relativ selten im Gesamtkontext vorkommt.
- *Summary sentences* sind wichtige Sätze als Teile des Textes, die dessen Inhalt bestmöglich wiedergeben. Jeder Satz im Text erhält einen *significance factor.* Der Satz mit dem höchsten *significance factor* wird der s*ummary sentence* des Textes. Die Prozedur zur Bestimmung des *significance factor* ist eine abgewandelte Form des Vorschlages von Luhn [15] und ist in [14] näher beschrieben. Sie erhält als Eingabe einen Satz und das Dokument in dem dieser

vorkommt. Die Ausgabe ist der *significance factor*.

Die Prozedur angewandt auf den Satz *S* arbeitet folgendermaßen:

1. Zuerst werden alle signifikanten Wörter markiert welche eine größere *importance* als ein Schwellwert *W* haben. *W* muss experimentell bestimmt werden.
2. Als zweiten Schritt finden wir alle *cluster* in *S* (Abb. 4).

Ein *cluster* ist eine Sequenz von aufeinander folgenden Worten für die folgendes gilt:

(i) *S* beginnt und endet mit einem signifikaten Wort.

(ii) weniger als *D* insignifikante Worte müssen beliebige signifikante Worte in *S* trennen. *D* wird auch der *distance cutoff* genannt, welcher wiederum experimentell bestimmt werden muss.

3. Die *importance* von allen *clustern* wird berechnet.
4. Das Maximum dieser *importance* Werte eines Satzes ist der *significance factor* des Satzes.

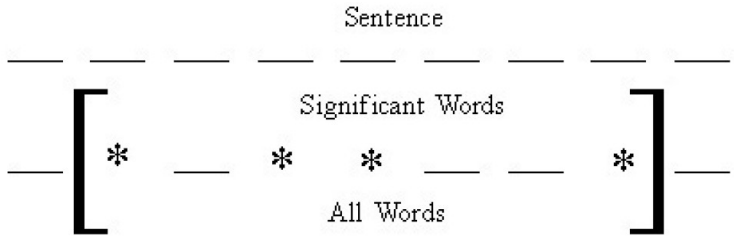

Abbildung 4: Identifikikation der Cluster in einem Satz [14]

4 Anpassung von Webseiten

Bei der Anpassung werden die analysierten Inhalte unter Berücksichtigung der gegebenen Eigenschaften des Clients, bestmöglich angepasst. Dabei geht es vor allem um die Minimierung von Benutzerinteraktion und eine bestmögliche Übersicht des Inhalts. Textzusammenfassungen können z.B. bei News-Seiten eine gute Übersicht bieten.

4.1 Textzusammenfassung

Texte sind die kleinste Einheit auf die Webseiten „runtergebrochen“ werden können. Neben der Option den Text auf verschiedene Seiten zu verteilen, kann er ihn auch so zusammengefasst werden, dass der Inhalt intuitiv ersichtlich ist und man bei Bedarf in den ganzen Text Schritt für Schritt „zoomen“ kann. Die verschiedenen Zustände können durch ein Klick auf den Text angewählt werden. Vier verschiedene Anzeigemethoden in denen ein Text dargestellt werden kann werden im Folgenden aufgezeigt [14]:

- *Incremental*: Hier wird der Text in drei verschiedenen Zuständen gezeigt; die erste Zeile; die ersten drei Zeilen; der gesamte Text (Abb. 5, links)
- *Keywords*: Der erste Zustand zeigt unsere wichtigen *keywords* aus der Textanalyse; der zweite Zustand zeigt die ersten drei Zeilen des Textes; der dritte Zustand zeigt den gesamten Text (Abb. 5, rechts)
- *Summary*: Besteht aus zwei Zuständen; der erste zeigt den wichtigsten Satz; der zweite den gesamten Text (Abb. 6, links)
- *Keyword/Summary*: Hier werden die zwei Methoden kombiniert; der erste Zustand zeigt die *keywords*; der zweite den wichtigsten Satz; der dritte den gesamten Text (Abb. 6, rechts)

Versuche mit verschiedenen Benutzern und Webseiten haben ergeben, dass die besten Ergebnisse

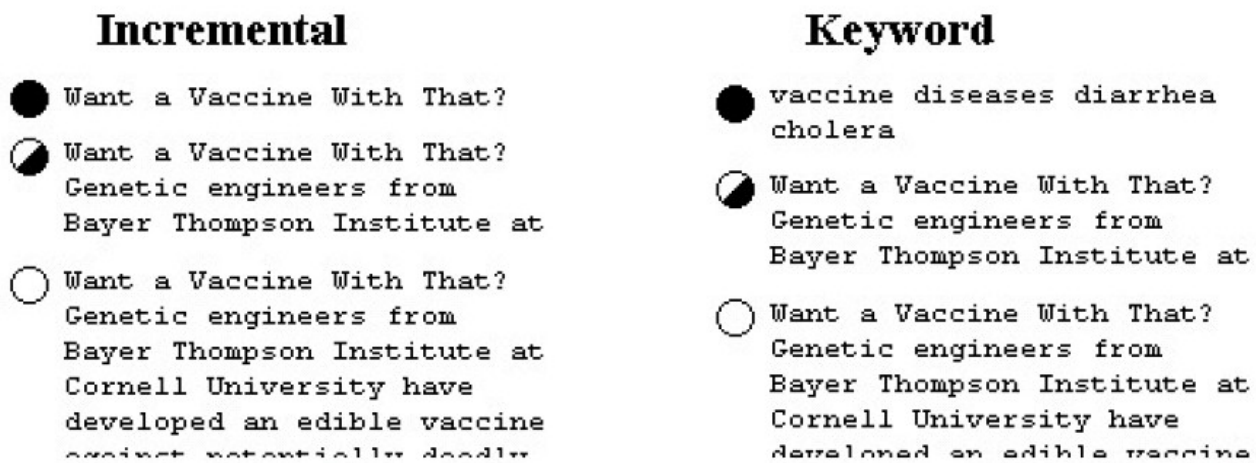

Abbildung 5: Die Inkrementelle- und die Keyword-Darstellung [14]

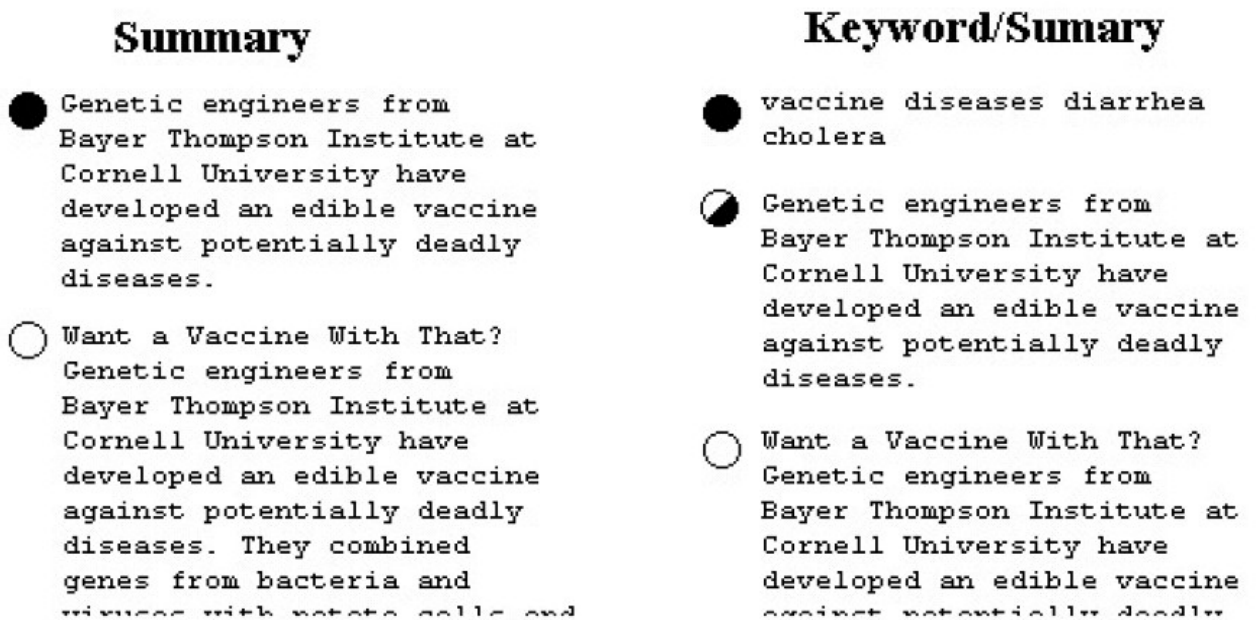

Abbildung 6: Die Summary und die kombinierte Keyword/Summary-Darstellung [14]

jeweils zu 50% die *Summary* und die *Keyword/Summary* Darstellung erzielt haben [14].

4.2 Selektion von Inhalten

4.2.1 Page-Splitting

Single-subject splitting ist ein Verfahren um einen längeren Inhalt auf mehrere Seiten zu verteilen. Dabei wird der Originaltext optimal auf mehrere Seiten verteilt und mit Links für *vorher* und *nacher* verbunden. Somit wird sichergestellt, dass der Benutzer den Inhalt in der richtigen Reihenfolge durchgehen kann (Abb. 7).

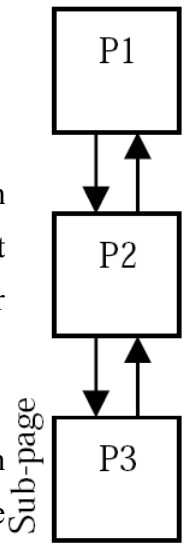

Abbildung 7: Single-subject splitting [10]

Multi-subject splitting generiert zusätzlich zu den Unterseiten eine neue Indexseite um diese zu gruppieren. Aus jeder Seite kann man wieder zum Index zurückkehren. Diese Aufteilung erzeugt eine zweistufige Hierarchie, die an die Gesamthierarchie angehängt wird (Abb. 8).

Section-outlining transformation erzeugt mehrere Unterseiten aus mehreren Kapiteln. Ein Index wird erzeugt, über den die Kapitel zugänglich sind (Abb. 9).

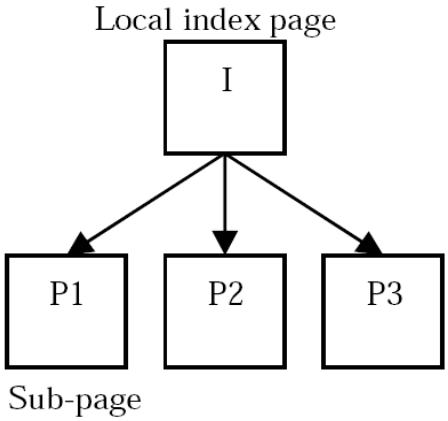

Abbildung 8: Multi-subject splitting [10]

4.2.2 Berücksichtigung des Benutzerverhaltens

Es ist mittlerweile eine weit verbreitete Strategie Statistiken über das Benutzerverhalten zu führen. Große Webseiten wie Amazon.com verwenden diese Statistiken um ihr Webangebot an den Benutzer anzupassen und so ein besseres Geschäftsergebnis

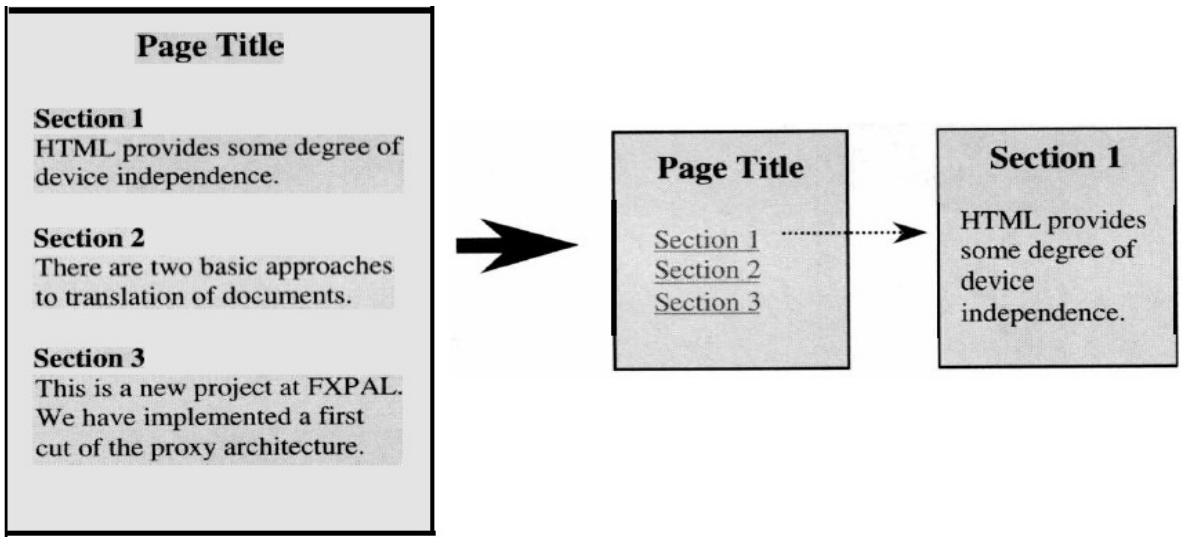

Abbildung 9: Section-outlining transformation [13]

zu erzielen.

Der Adaptor kann die gleiche Technik anwenden um weniger genutzte von öfter genutzen Inhalten auf einer bestimmten Webseite zu unterscheiden und weniger genutze Inhalte sukzessiv auszublenden. Sie können aber bei Bedarf über einen Hinweis immer noch erreichbar bleiben [16]. Somit kann eine bessere Ausnutzung der Bildschirmfläche und eine schnellere Navigation erreicht werden.

4.3 Automatisiertes Layout bzw. Navigation und Browsing

Die Neugestaltung der Webseite für den kleineren Bildschirm ist entscheidend für das subjektive Empfinden und ausschlaggebend für eine breite Akzeptanz bei den Benutzern. Je besser das Redesign ist, um so größer ist die Chance auf dem Markt mit dem Produkt erfolgreich zu sein. Mit dem Thumbnail-Browsing und der automatischen Positionierung wird ein ähnliches Surfverhalten wie auf einem PC forciert.

4.3.1 Thumbnail-Browsing

Beim Thumbnail-Browsing wird die ursprüngliche Webseite als Thumbnail dargestellt und dient als visueller Index. Alle Inhaltsblöcke sind auf dem Thumbnail verlinkt, farblich voneinander abgesetzt und können einzeln angewählt werden. Diese Verfahrensweise gibt den Komfort der Übersicht und des Browsing eines PCs sehr gut wieder, da man die komplette Seite sehen und anvisieren kann.

Abbildung 10: Thumbnail-Browsing [10]

Abbildung 10 verdeutlicht diese Verfahrensweise sehr gut. Es werden somit weniger Benutzerinteraktionen mit beispielsweise einem Stift benötigt, um an den gewünschten Inhalt auf der Seite zu gelangen.

4.3.2 Auto-positioning vs. Page-Splitting

Beim Page-Splitting kann unter Umständen folgendes Problem auftreten. Wenn wir auf der Subpage einen Inhalt haben, der nicht den gesamten Bildschirm in Anspruch nimmt, dann verschwenden wir Bildfläche. Außerdem kann eine Verbesserung in der Navigation erfolgen, wenn wir das Thumbnail-Browsing verwenden. Wir könnten sozusagen auf die Stelle hinzoomen und wären in der Lage das Ergebnis zu scrollen, statt mit einem Link hinein- und wieder herausspringen zu müssen, was in den Abbildungen 11 und 12 verdeutlicht wird.

Abbildung 11: Page Splitting [10]

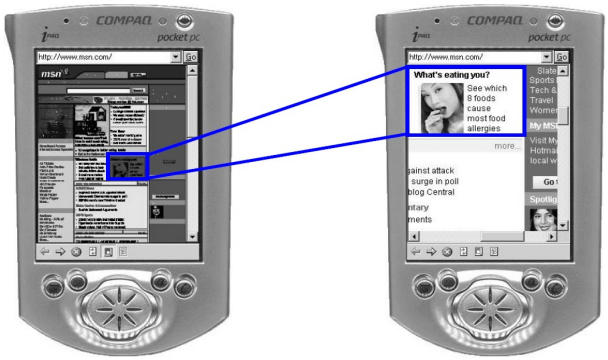

Abbildung 12: Auto-positioning [10]

5 Alternative Ansätze

Bei unseren Ausführungen wurde die ganze Zeit davon ausgegangen statische (X)HTML Seiten nachträglich anzupassen. Diese Vorgehensweise bringt für viele Seiten eine akzeptable Lösung, ist aber nicht das, was informationstechnisch möglich wäre. Heuristiken können nie eine Lösung für alle Probleme sein, da sie viele Voraussetzungen benötigen, um überhaupt halbwegs gut zu funktionieren.

Ein alternativer Ansatz wäre die Trennung von Daten, Layout und Logik bereits auf dem Webserver und jeweils eine automatisch angepasste Darstellung für jeden Clienttyp. Als Beispiel für eine solche Lösung soll hier beispielhaft Cocoon [17] betrachtet werden.

Cocoon ist ein Framework das sich auf das XML-Publishing spezialisiert hat. Es wird oft als *web development framework* eingesetzt. Cocoon hält die drei Informationstypen in XML Dokumenten

[18] und ist in der Lage, mit XSLT [19] eine für einen spezifischen Zweck angepasste Ausgabe unter anderem auch in HTML zu erzeugen.

XSLT ist eine Programmiersprache zur Transformation von XML-Dokumenten. Es baut auf die logische Baumstruktur eines XML-Dokumentes auf und erlaubt die Definition von Umwandlungsregeln. Die Umwandlung geschieht dann in einem XSLT-Prozessor (Abb. 13).

Die einzige noch zu erledigende Aufgabe ist es, geeignete Transformationsregeln in XSLT für die verschiedenen Clienttypen, wie PC, PDA, SmartPhone u.ä. zu erstellen, die unseren Bedingungen aus dieser Arbeit genügen, und den Client anzuweisen, bei jeder Anfrage seinen Clienttyp anzugeben (Abb. 14).

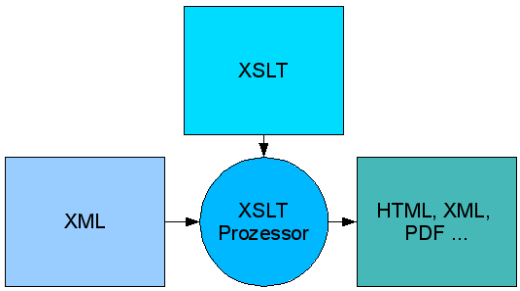

Abbildung 13: XSLT-Transformation

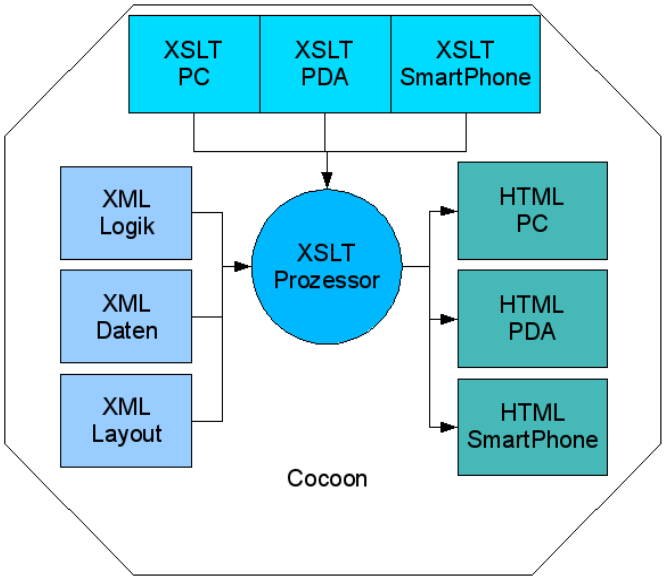

Abbildung 14: Arbeitsweise von Cocoon als Adaptor

6 Zusammenfassung und Ausblick

In dieser Arbeit wurden verschiedene Verfahren zur automatischen Anpassung von Webseiten an heterogene Clients aufgezeigt. Die präsentierten Lösungen werden z.T. mit geringen Abwandlungen in verschiedenen Systemen bereits erfolgreich eingesetzt [20][13][10].

Eine Anpassung mit einem XML-Framework ist zwar wünschenswert und sehr sinnvoll, erfordert aber Anpassungen beim Server. Eine Anpassung der Inhalte vom Anbieter aus für jeden individuellen Gerätetyp wäre sehr schwierig und zu aufwändig. Auch aus diesem Grund werden Adaptionstechnologien in der Zukunft des Pervasive Computing [21] an immer größerer Bedeutung

gewinnen.

Wie sich auch schon in der Vergangenheit gezeigt hat gab und gibt es im Internet für eine Problemstellung immer mehrere freie und kommerzielle Lösungen. Uns bleibt zu hoffen dass auch in Zukunft Vielfalt in jeder Hinsicht erhalten bleibt.

Literaturverzeichnis

[1] T. W. Bickmore, B.N. Schilit, "Digestor: Device-independent access to the World Wide Web" Proc. of the 6th International WWW Conference, Santa Clara, CA, April 1997

[2] Avant Go, http://www.avantgo.com

[3] Opera Mini, http://www.opera.com/products/mobile/operamini/

[4] Jinlin Chen, Baoyao Zhou, Jin Shi, Hongjiang Zhang, Qiu Fengwu, "Function-Based Object Model Towards Webseite Adaption" Proceedings of the 10th international conference on World Wide Web, S. 587 – 596, 2001

[5] W3C HTML Working Group XHTML™ 1.0, „The Extensible HyperText Markup Language (Second Edition)“, http://www.w3.org/TR/xhtml1

[6] Dave Raggett, Arnaud Le Hors, Ian Jacobs, „HTML 4.01 Specification“, http://www.w3.org/TR/html4

[7] Rakesh Mohan, John R. Smith, Chung-Sheng Li, „Adapting Multimedia Internet Content for Universal Acess“ 1998

[8] Kwang Bok Lee, Roger A. Grice, "An Adaptive Viewing Application for the Web on Personal Digital Assistants" Proceedings of the 21st annual international conference on Documentation, S.125 – 132, 2003

[9] Jiawei Han, Micheline Kamber, „Data Mining: Concepts and Techniques“ Morgan Kaufmann Publishers Inc. 2001

[10] Yu Chen, Wei-Ying Ma, Hong-Jiang Zhang, „Detecting Web Page Structure for Adaptive Viewing on Small Form Factor Devices“, Proceedings of the 12th international conference on World Wide Web, May 2003

[11] Gerald Penn, Jianying Hu, Hengbin Luo, Ryan McDonald, „Flexible Web Document Analysis for Delivery to Narrow-Bandwith Devices“, Proceedings of the Sixth International Conference on Document Analysis and Recognition, Seattle, Washington, S. 1074-1078, 2001

[12] Document Object Model (DOM), http://www.w3.org/DOM/

[13] Timothy Bickmore, Andreas Girgensohn, Joseph W. Sullivan: „Web Page Filtering and Re-Authoring for Mobile Users“, MIT Media Laboratory: The Computer Journal, Vol. 42, No. 6, 1999

[14] Orkut Buyukkokten, Hector Carcia-Molina, Andreas Paepcke, „Seeing the Whole in Parts: Text Summarization for Web Browsing on Handheld Devices“, Proceedings of the 10th international conference on World Wide Web, April 2001

[15] H.P. Luhn, „The Automatic Creation of Literature Abstracts“, IBM Journal of Research and Development, S. 155-164, April 1958

[16] Mike Perkowitz, Oren Etzioni, „Towards Adaptive Web Sites: Conceptual Framework and Case Study“, Proceeding of the eighth international conference on World Wide Web, S.1245-1258, May 1999, Toronto, Canada

[17] The Apache Cocoon Project, http://cocoon.apache.org/

[18] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, François Yergeau, „Extensible Markup Language (XML) 1.0 (Third Edition)“: http://www.w3.org/TR/REC-xml

[19] James Clark, „XSL Transformations (XSLT)“, http://www.w3.org/TR/xslt

[20] Orkut Buyukkokten, „Power Browsing“, http://dbpubs.stanford.edu:8091/~testbed/doc2/PowerBrowsing/

[21] M. Satyanarayanan: „Pervasive Computing: Vision and Challenges“, IEEE Personal Communications, 2001