

Entwicklung einer Benutzeroberfläche für  
die Mehrfachauswahl von Werten im  
Kontext von Faceted Search

LISA POLLHAMMER

DIPLOMARBEIT

eingereicht am  
Fachhochschul-Masterstudiengang

INTERACTIVE MEDIA

in Hagenberg

im Juni 2011

© Copyright 2011 Lisa Pollhammer

Diese Arbeit wird unter den Bedingungen der *Creative Commons Lizenz Namensnennung–NichtKommerziell–KeineBearbeitung Österreich* (CC BY-NC-ND) veröffentlicht – siehe <http://creativecommons.org/licenses/by-nc-nd/3.0/at/>.

# Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die aus anderen Quellen entnommenen Stellen als solche gekennzeichnet habe.

Hagenberg, am 23. Juni 2011

Lisa Pollhammer

# Inhaltsverzeichnis

<b>Erklärung</b>	<b>iii</b>
<b>Vorwort</b>	<b>vi</b>
<b>Kurzfassung</b>	<b>vii</b>
<b>Abstract</b>	<b>viii</b>
<b>1 Einleitung</b>	<b>1</b>
<b>2 Stand der Technik</b>	<b>3</b>
2.1 Was ist Faceted Search? . . . . .	3
2.1.1 Arten von Werten und Facetten . . . . .	4
2.1.2 Auswahl von Werten . . . . .	5
2.1.3 Technische Aspekte . . . . .	5
2.2 Herausforderungen und Probleme von Faceted Search Interfaces	6
2.2.1 Probleme bei Mehrfachauswahl . . . . .	7
2.3 Beispiele für Faceted Search Interfaces . . . . .	8
2.3.1 Flamenco . . . . .	9
2.3.2 eBay . . . . .	10
2.3.3 Geizhals . . . . .	12
<b>3 Design</b>	<b>14</b>
3.1 Übergreifende Designentscheidungen . . . . .	14
3.1.1 Darstellung der Facetten . . . . .	16
3.1.2 Komponenten der Interfaces . . . . .	18
3.1.3 Suchergebnisse . . . . .	19
3.1.4 Konsistenz . . . . .	20
3.2 Blockinterface . . . . .	21
3.2.1 Anordnung der Elemente . . . . .	21
3.2.2 Darstellung der Werte . . . . .	24
3.3 Ellipseninterface . . . . .	24
3.3.1 Anordnung der Elemente . . . . .	24
3.3.2 Darstellung der Werte . . . . .	26

<b>4 Implementierung</b>	<b>28</b>
4.1 Designentscheidungen . . . . .	28
4.1.1 Drupal . . . . .	29
4.2 Faceted Search Modul . . . . .	29
4.2.1 Konfiguration und Testdaten . . . . .	30
4.2.2 Grundfunktionen des Moduls . . . . .	31
4.2.3 Erweiterungen im Modul . . . . .	31
4.3 Erstellung der Interfaces . . . . .	37
4.3.1 Blockinterface . . . . .	37
4.3.2 Ellipseninterface . . . . .	38
<b>5 Evaluierung</b>	<b>41</b>
5.1 Testdesign . . . . .	41
5.1.1 Szenarien . . . . .	42
5.2 Testablauf . . . . .	43
5.3 Testergebnisse . . . . .	43
5.3.1 Interface 1: Geizhals . . . . .	44
5.3.2 Interface 2: Drupal's Faceted Search Modul . . . . .	47
5.3.3 Interface 3: Blockinterface . . . . .	49
5.3.4 Interface 4: Ellipseninterface . . . . .	51
5.4 Diskussion . . . . .	53
5.4.1 Darstellung der Facetten und Werte . . . . .	54
5.4.2 Hervorhebung aktiver Werte . . . . .	55
5.4.3 „Current Search“ . . . . .	56
5.4.4 Suchergebnisse . . . . .	56
5.4.5 Position der Interfaces . . . . .	58
5.4.6 Dynamisches Nachladen von Interface-Elementen . . . . .	58
5.4.7 Zeit und Zufriedenheit . . . . .	59
<b>6 Schlussbemerkungen</b>	<b>61</b>
<b>7 Inhalt der CD-ROM/DVD</b>	<b>63</b>
7.1 PDF-Dateien . . . . .	63
7.2 Quellcode . . . . .	63
7.3 Sonstiges . . . . .	64
<b>Literaturverzeichnis</b>	<b>65</b>

# Vorwort

Jeder, der sich auch kurz im World Wide Web aufhält, hat früher oder später mit Faceted Search Interfaces Bekanntschaft gemacht. Meistens findet man diese auf E-Commerce Seiten, wo sie einem die Suche nach gewünschten Produkten erleichtern sollen. Besucht man nun mehrere dieser Seiten regelmäßig, fängt man unweigerlich an die verschiedenen Benutzeroberflächen zu vergleichen. Man realisiert, dass das Interface an sich einen großen Teil dazu beiträgt, ob man auf einer Seite gerne Zeit verbringt oder sich nur die nötigsten Informationen aneignet und die Seite so schnell wie möglich wieder verlässt. Einige dieser Interfaces irritieren beim ersten Besuch der Seite, wirken unüberlegt und unorganisiert. Gerade diese Interfaces motivierten mich, dieses Thema im Masterprojekt und dieser Arbeit aufzugreifen und zu versuchen, ein für mich und andere Benutzer ansprechendes Interface zu gestalten.

Dass der Entwurf eines guten und funktionsfähigen Interfaces doch nicht so einfach ist wie gedacht, durfte ich während der Entwicklungszeit selbst erfahren. Viele Kompromisse müssen getroffen werden, viele Ideen scheinen zuerst sinnvoll, innovativ und selbsterklärend und im Usability-Test stellt sich heraus, dass niemand ausser man selbst den Sinn der Komponente verstanden hat.

Ich denke, dass ich in dieser Arbeit trotz allem gute Ergebnisse liefern konnte und möchte mich dafür bei meinem Diplomarbeits-Betreuer DI Rimbart Rudisch-Sommer bedanken. Er hat mich während der gesamten Zeit tatkräftig unterstützt, mir geholfen nicht immer zu kompliziert zu denken aber auch ermutigt, ungewöhnliche Ansätze trotzdem zu verfolgen.

Großer Dank gilt meinem Partner Rainer, der immer ein offenes Ohr für meine Ideen hatte und mir oft eine andere Sichtweise auf Dinge lieferte. Wenn ich wieder einmal ratlos vor meinem Computer saß, weil das Drupal Modul machte was es wollte, hat er mich immer wieder ermutigt einen anderen Weg zu finden und plötzlich funktionierten Dinge die vorher unmöglich schienen.

Zu guter Letzt möchte ich mich bei meinen Eltern Rudolf und Hermine Pollhammer bedanken, die immer an mich geglaubt haben und mir die Möglichkeit gegeben haben, dieses Studium sorgenlos absolvieren zu dürfen. Beide verfügen über keine Fachkenntnis und haben trotzdem tapfer Korrektur gelesen und damit maßgeblich zur Korrektheit dieser Arbeit beigetragen.

# Kurzfassung

Faceted Search Interfaces sind eine weiterentwickelte Form der Kategorie-Suche, wobei hier das gewünschte Suchergebnis durch die Anwendung von Werten, gruppiert in Facetten, als Filter auf den gesamten Datenbestand erreicht wird. Diese Interfaces sind weit verbreitet und werden vielseitig eingesetzt. Gerade bei der Mehrfachauswahl von Werten kann es jedoch zu gewissen Problemen und Herausforderungen kommen, die bei der Erstellung eines solchen Interfaces beachtet werden müssen.

Aufbauend auf nachweislich erfolgreichen Funktionen und Komponenten von Faceted Search Interfaces werden neue Ansätze in diesem Bereich vorgestellt und diese in zwei verschiedenen Prototypen umgesetzt: Der erste Prototyp, das Blockinterface, vereint konventionelle Techniken und Funktionen aus Faceted Search Interfaces mit neuen Technologien und überarbeiteten Komponenten. Mit dem zweiten Prototypen, dem Ellipseninterface, wird ein gänzlich neuer Ansatz zur Organisierung und Strukturierung der Elemente in Faceted Search Interfaces präsentiert.

Beide Prototypen wurden in einer Usability-Studie miteinander verglichen, wobei zum zusätzlichen Vergleich zwei fremde Faceted Search Interfaces in die Studie mit aufgenommen wurden. Die Ergebnisse der Studie zeigen, dass Struktur und Übersichtlichkeit in einem Faceted Search Interface mitunter den größten Teil zur Zufriedenheit der Benutzer beitragen und ein gut organisiertes Interface bevorzugt wird, auch wenn mehr Zeit benötigt wird bis das gewünschte Suchergebnis erreicht ist. Während die gänzlich neuartige Organisation der Interface-Elemente im Ellipseninterface von den Teilnehmern der Studie nicht gut angenommen wurde, konnte mit den überarbeiteten Komponenten und Funktionen des Blockinterfaces die größte Zufriedenheit unter allen Teilnehmern erreicht werden.

# Abstract

Faceted search interfaces are a very popular tool for searching and browsing websites. Those with the option of selecting multiple values from facets are powerful and versatile but also require certain considerations and decisions during the design process.

This document presents new approaches for the development of faceted search interfaces with the option of selecting multiple values based on known and successful components and functions. It follows the design and development process of two different interfaces which feature these new functions and behavior. The first design, the block interface, combines well-established interface components and presentation methods with new functionality and behavior whereas the second prototype, the ellipse interface, shows a new and different angle for organizing and structuring elements.

Based on the results of an informal usability test, both interfaces were evaluated and compared to each other and to two foreign faceted search interfaces, which were carefully selected to measure the success of the newly developed designs. The results have shown that a user's satisfaction is influenced greatly by structure and well-arranged elements. An interface that has been well thought through and contains well-organized components is preferred, even if it takes the user more time to finish the task and to obtain the desired results.

The new approach to arranging and organizing interface elements in the ellipse interface did not convince the participants enough to be successful. The block interface on the other hand did benefit from the combination of known interface elements and new and improved functionality and therefore turned out to be the most preferred interface of all tested interfaces.



# Kapitel 1

## Einleitung

Faceted Search Interfaces sind auf dem Weg, altbekannte Such-Interfaces abzulösen. Diese Art der Suche verbindet die Vorteile verschiedener Sucharten wie Stöbern, die gezielte Suche nach Produkten oder eine erweiterte Suche mit vielen Einstellungsmöglichkeiten und wird daher speziell auf E-Commerce Seiten immer mehr eingesetzt. Ein Faceted Search Interface bietet dem Benutzer viele Möglichkeiten, die vorhandenen Produkte, durch die Anwendung von Facetten und Werten als Filter, auf die gewünschte Ergebnismenge hin zu reduzieren.

Am Beispiel des Faceted Search Interfaces von Geizhals<sup>1</sup> kann man bereits während der Interaktion erkennen, dass diese Art von Interface nicht nur über Vorteile verfügt, sondern auch gewisse Probleme mit sich bringt. Besonders bei vielen Filtermöglichkeiten und Produkten im Datenbestand ist es eine große Herausforderung, das Such-Interface übersichtlich und strukturiert zu gestalten. Ein gutes Interface zielt darauf ab, dem Benutzer den Aufenthalt auf der Seite so angenehm wie möglich zu gestalten und ihn so zu unterstützen, dass die Suche nach Produkten erfolgreich ist.

Einige der Faceted Search Interfaces bieten die Möglichkeit, aus derselben Gruppe von Werten, den sogenannten Facetten, mehrere Werte als Filter anwenden zu können. Dabei ist es wichtig, den Benutzern der Seite alle Werte im Interface konstant anzuzeigen, ohne das Interface mit diesen Werten zu überfüllen. Bei der ausgiebigen Recherche von aktuellen Faceted Search Interfaces hat sich herausgestellt, dass dafür einige gute Lösungen gefunden wurden, jedoch in kaum einem dieser Interfaces die Umsetzung mit neuen Techniken gewählt und statt dessen auf alt bekannte Interface-Elemente gesetzt wurde.

Gerade diese Problematik der Mehrfachauswahl von Werten macht die Erstellung eines neuen Faceted Search Interfaces interessant. Der Einsatz von neuen Technologien ermöglicht es, die Organisierung von Werten und das Verhalten des Interfaces zu verbessern und dem Benutzer so einen noch

---

<sup>1</sup><http://www.geizhals.at>

leichteren Zugang zu einem Faceted Search Interface zu ermöglichen.

Faceted Search Interfaces haben sich auf vielen E-Commerce Seiten mittlerweile etabliert und die Weiterentwicklung dieser Interfaces ist in vollem Gange.

In diesem Dokument sollen neue Ideen und Techniken gefunden und präsentiert werden, wie Faceted Search Interfaces mit der Möglichkeit der Mehrfachauswahl erfolgreich konstruiert werden können.

Zu diesem Zweck werden zwei verschiedene Prototypen entwickelt. Das Blockinterface repräsentiert den Versuch, bereits gewonnene Kenntnisse im Bereich Faceted Search mit neuartigen Funktionen zu verknüpfen. Im Ellipseninterface hingegen wird ein gänzlich neuer Ansatz zur Strukturierung eines Faceted Search Interfaces implementiert.

Zur Evaluierung beider Interfaces wird eine Usability-Studie durchgeführt. In dieser Studie werden außerdem zu Vergleichszwecken zwei fremde Faceted Search Interfaces getestet, unter anderem das Interface von Geizhals.

Dieses Dokument umfasst sechs Kapitel, wobei die einzelnen Kapitel auf die vorhergehenden Kapitel inhaltlich aufbauen: In Kapitel 2 werden die Grundlagen und Herausforderungen von Faceted Search erklärt und einige aktuelle Faceted Search Interfaces präsentiert. Kapitel 3 bespricht den gesamten Entwicklungsprozess der neu entwickelten Interface-Komponenten, während in Kapitel 4 die Implementierung der einzelnen Interfaces beschrieben wird. In Kapitel 5 wird kurz das Design der Usability-Studie erörtert und anschließend die einzelnen Interfaces evaluiert und miteinander verglichen. Kapitel 6 beinhaltet neben einer kurzen Zusammenfassung der gewonnenen Erkenntnisse einen Ausblick auf die Weiterentwicklung der Faceted Search Interfaces.

## Kapitel 2

# Stand der Technik

Einen großen Aspekt der Navigation im Web stellt heutzutage die Suche dar. Ohne Suchmöglichkeit ist es mühsam, sich in der Vielfältigkeit des Webs zu orientieren und das zielgerichtete Finden von Informationen ohne Vorwissen ist fast unmöglich. Der erste Schritt auf der Suche nach Informationen führt meistens zu Suchmaschinen. Sind geeignete Seiten gefunden worden, wird zum Stöbern oder zur gezielten Suche die eingebaute Webseiten-Suche genutzt. Speziell auf Webseiten aus dem E-Commerce Bereich ist das Suchinterface von großer Wichtigkeit, da es den Ausgang der Suche enorm beeinflusst. Man möchte die Besucher dazu animieren, beim Stöbern interessante und passende Produkte zu entdecken, die zum Kauf animieren oder bei der zielgerichteten Suche die gewünschten Produkte schnell zu finden.

Dies sind zwei grundsätzlich verschiedene Vorgänge und verlangen jeweils nach einem geeigneten Interface. Für eine zielgerichtete Suche muss schnell ein Suchergebnis gefunden werden – das Interface soll einfach und verständlich gestaltet sein und über Grundfunktionen verfügen. Das Interface zum Stöbern sollte stattdessen viele Möglichkeiten bieten und den Benutzer unterstützend durch die Menge an Produkten leiten und durch die Art der Suchergebnisse zum Kauf animieren.

Ein Faceted Search Interface vereint die Vorteile dieser Interfaces und wird aus diesem Grund auf vielen E-Commerce Webseiten wie Amazon<sup>1</sup>, eBay<sup>2</sup> oder Geizhals) eingesetzt.

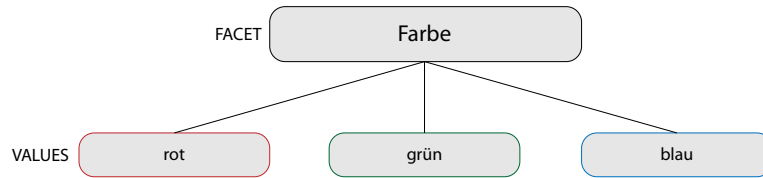
### 2.1 Was ist Faceted Search?

Faceted Search ist eine flexible Form der Suche, die dem Benutzer erlaubt, durch das Anwenden von Suchfiltern (Facetten) das Suchergebnis zu reduzieren und somit auf eine gewünschte Ergebnismenge einzuschränken. Eine Facette stellt eine Eigenschaft des Produkts oder des Objekts dar, welche

---

<sup>1</sup><http://www.amazon.at>

<sup>2</sup><http://www.ebay.at>



**Abbildung 2.1:** Logische Darstellung einer Facette (Farbe) und dessen Werte (rot, grün, blau).

über beliebig viele Werte verfügen kann.

Ein Produkt verfügt z.B. über die Eigenschaft Farbe. Die Eigenschaft Farbe (siehe Abbildung 2.1) beinhaltet mehrere Werte, in diesem Fall rot, grün und blau. Es können nun den Produkten bzw. Objekten beliebig viele Werte dieser Eigenschaft zugewiesen werden. Bei der Suche werden dann bei der Wahl eines (oder mehrerer) Wert(e)s genau die Objekte mit diesem Wert aus der vorhandenen Produktmenge gefiltert und angezeigt. Durch die Anwendung mehrere Werte als Filter wird somit eine Suche nach genau diesen Kriterien gestartet und die Ergebnismenge auf jene Produkte mit den gewünschten Eigenschaften eingegrenzt.

Der Benutzer der Webseite kann seine Suche somit sehr variabel gestalten. Er kann das Suchergebnis durch die Auswahl von Werten eingrenzen, oder durch das Entfernen von Werten wieder erweitern. Der Benutzer hat somit nicht den Druck, bereits im Vorhinein exakt über seine Wünsche Bescheid wissen zu müssen und wird auch speziell durch die Zusammensetzung der Suchergebnisse geleitet.

Diese Art der Suche eliminiert die Notwendigkeit, eine Suche komplett neu zu starten, weil plötzlich keine Suchergebnisse mehr gefunden wurden und der Benutzer sich in einer Sackgasse befindet. Es wird dem Benutzer ein Gefühl von Kontrolle vermittelt, weil er den Verlauf der Suche immer selbst bestimmen kann und durch gewisse Faktoren bereits vor der Auswahl von einzelnen Werten über die Auswirkung informiert werden kann (siehe Kapitel 3).

### 2.1.1 Arten von Werten und Facetten

Es gibt mehrere Ansätze, wie ein Faceted Search System aufgebaut wird bzw. wie genau die Werte zu den Produkten zugeordnet werden (für den technischen Ansatz zur Extrahierung von Facetten und Werten siehe Abschnitt 2.1.3).

Die Daten bzw. Facetten können hierarchisch oder flach definiert werden. Facetten in einer **flachen** Struktur haben keine Beziehung zueinander und können daher voneinander unabhängig ausgewählt werden. Eine Hose verfügt

z. B. über Farbe (blau), Material (Jeans) und Schnitt (regular), wobei die Eigenschaften unabhängig und getrennt ebenso informativ sind.

Andererseits können Facetten in einer **hierarchischen** Struktur nur voneinander abhängig ausgewählt werden bzw. müssen bestimmte Werte von Facetten aktiv sein, um andere Facetten (sowie Werte) erst zugänglich zu machen. Die Stadt Linz verfügt z. B. über die Eigenschaften Staat (Österreich), Bundesland (Oberösterreich), Bezirk (Linz Stadt) und Gemeinde (Linz), wobei die verfügbaren Werte von Bundesland, Bezirk und Gemeinde offensichtlich von ihren jeweils hierarchisch übergeordneten Werten abhängig sind.

Weiters ist zwischen Einfach- und Mehrfachzuweisung von Werten in Faceted Search Interfaces zu unterscheiden. Eine **Einfachzuweisung** liegt vor, wenn das Objekt genau einen Wert der Facette zugewiesen bekommt. Von **Mehrfachzuweisung** wird dann gesprochen, wenn einem Objekt aus einer Facette mehrere Werte zugeteilt werden. Eine Frucht ist z. B. von der Sorte Apfel (Einfachzuweisung, Sorte), hat aber die Farben grün und rot (Mehrfachzuweisung, Farbe).

### 2.1.2 Auswahl von Werten

Bei Faceted Search werden zwei verschiedene Möglichkeiten der Auswahl von Werten unterschieden: Einfachauswahl und Mehrfachauswahl. Bei einer **Einfachauswahl** wird dem Benutzer im Interface ermöglicht, genau einen Wert einer Facette auszuwählen und bei Bedarf abzuändern. Es ist jedoch zu jeder Zeit maximal ein Wert der Facette aktiv.

Gibt es im Suchinterface die Möglichkeit einer **Mehrfachauswahl**, kann der Benutzer durch die Wahl mehrere Werte aus einer Facette das Suchergebnis vergrößern, da alle Produkte mit dem einen oder dem anderen Wert (oder beiden) angezeigt werden.

### 2.1.3 Technische Aspekte

Um überhaupt eine Faceted Search anbieten zu können, müssen natürlich Facetten und Werte im Datenbestand vorhanden sein. Prinzipell kann jegliche Metainformation eines Objekts als Wert angesehen werden. Am Beispiel von benutzergeneriertem Inhalt im *Content Management System*(CMS)<sup>3</sup> Drupal<sup>4</sup> wären dies z. B. Erstellungsdatum, Autor, Kategorie usw. In diesem Fall können auch manuell zugewiesene Tags als Werte dienen, wobei der Überbegriff der Tags die Facette darstellt.

Suchquers können bei kleinen Inhaltsseiten über normale SQL-Abfragen durchgeführt werden, wobei natürlich darauf geachtet werden muss, dass die Abfragen bei vielen Daten performant bleiben.

---

<sup>3</sup>[http://en.wikipedia.org/w/index.php?title=Content\\_management\\_system&oldid=432471544](http://en.wikipedia.org/w/index.php?title=Content_management_system&oldid=432471544), Kopie auf CD-ROM (Datei docs/CMS.pdf vom 04.06.2011)

<sup>4</sup><http://drupal.org>

Für die Extraktion von Werten und Facetten aus einem bestehenden Datenbestand gibt es viele Ansätze und je nach Einsatzgebiet ist die Vorgehensweise sehr unterschiedlich. Sehr populär für die automatische Erstellung von Werten und Facetten aus Dokumenten ist im Moment der Einsatz von Apache Solr<sup>5</sup>, aufbauend auf Apache Lucene<sup>6</sup>. Während Lucene die Indizierung vorhandener Dokumente übernimmt, liefert Solr eine Suchplattform, in der unter anderem Faceted Search unterstützt wird. Diese Art der automatisierten Verarbeitung ist dann zu empfehlen, wenn große Datenmengen vorhanden sind, da Solr speziell für seine Skalierbarkeit bekannt ist.

Es gibt mittlerweile auch Ansätze, mehr als nur die üblichen Metadaten in Facetten und Werte umzuwandeln. Mehr dazu findet sich in [1].

## 2.2 Herausforderungen und Probleme von Faceted Search Interfaces

Wie bei allen anderen Interfaces gibt es auch bei Faceted Search Interfaces gewisse Herausforderungen, denen man sich bei der Entwicklung stellen muss. Generell ist es erstrebenswert, dass der Benutzer sich problemlos im Interface zurechtfindet und sich unter keinen Umständen in der Navigation verirrt oder das Gefühl hat, nicht mehr zu wissen was er tut oder wie das Interface reagiert.

Ben Shneiderman hat für die Entwicklung von User-Interfaces Design Kriterien entwickelt, welche auch für Faceted Search Interfaces erstrebenswert sind [19]:

- Streben nach Konsistenz
- Schneller Zugriff für Experten
- Informatives Feedback
- Abgeschlossenes Dialog-Design
- Einfache Handhabung von Fehlern
- Einfaches Widerrufen von Aktionen
- Einfluss des Benutzers unterstützen
- Optimierung für das Kurzzeitgedächtnis

Daraus schließend soll die Navigation bzw. Suche im Interface dem Benutzer so einfach wie möglich zugänglich gemacht werden. Das Interface soll für sich selbst sprechen und dem Benutzer die Angst nehmen, etwas falsch machen zu können bzw. mit unerklärlichem Verhalten konfrontiert zu sein. Gerade Konsistenz ist in diesem Sinne wichtig, so dass mögliche Aktionen nicht immer hinterfragt werden müssen, sondern logisch und eindeutig sind.

---

<sup>5</sup><http://lucene.apache.org/solr/>

<sup>6</sup><http://lucene.apache.org/>

Eine klare Darstellung aktiver Facetten ermöglicht dem Benutzer, bereits getätigte Aktionen immer im Blick zu haben und währenddessen neue Schritte zu planen, ohne sich die vorhergehenden merken zu müssen.

Neben diesen „Standard“ Usability-Kriterien gibt es weitere Faktoren, die ein solches Interface zusätzlich beeinflussen. Zu allererst sollte bei der Entwicklung des Interfaces bereits bekannt sein, mit welcher Art von Daten interagiert wird (siehe Abschnitte 2.1.1, 2.1.2, und 2.1.3).

Sogenannte „Query-Previews“ können dem Benutzer unter anderem helfen, die möglichen Auswirkungen bestimmter Werte und Facetten bereits im Vorfeld zu erahnen. Eine der am meist verbreitetsten Möglichkeiten ist hier die Darstellung der Anzahl der betreffenden Objekte in Klammer neben den einzelnen Werten. Benutzer wissen somit exakt, wieviele Objekte von einer bestimmten Auswahl betroffen sind und können sich daher im Vorhinein noch entscheiden, ob die Auswahl für ihre Zwecke Sinn macht oder nicht.

Die Art der Daten ist entscheidend für die Darstellung der Facetten und Werte im Interface. Wird dem Benutzer die Möglichkeit geboten, mit hierarchischen Facetten oder einer Mehrfachauswahl zu arbeiten, ergibt sich ein definitiv größeres Platzproblem als bei einer reinen Einfachauswahl. Es ist jedoch zu bedenken, dass die erweiterte Möglichkeit der Navigation bei Benutzern beliebter ist als die Limitierung der Navigation durch eine einzelne Facette [2] und deshalb die Mehrfachauswahl nicht automatisch ausgeschlossen werden sollte.

Dieses Platzproblem lässt sich auch bei Geizhals in Abbildung 2.2 gut erkennen. Es ist bekannt, dass von Beginn an sichtbare Werte einen klaren Vorteil über versteckten Werten haben [2]. Jedoch ist fraglich, ob diese Begründung genug ist, auch bei einer großen Anzahl von Werten keine Beschränkung einzubauen (siehe Kapitel 5), da dadurch das Interface übersichtlicher werden kann.

Der Benutzer sollte nach einer getätigten Auswahl von Werten einen Überblick über die Auswirkungen erhalten und daher ist es wichtig, dass zumindest Teile des Suchergebnisses sofort im Blickfeld erscheinen. Ist es nötig, dass dafür gescrollt werden muss, kann es dazu führen, dass der Benutzer die Seite verlässt [4].

Generell ist es wünschenswert, dass Facetten und Werte organisiert aufgelistet werden. Definitiv bevorzugt sind Darstellungsmöglichkeiten, in denen eine klare und sichtbare Abtrennung der einzelnen Punkte erkennbar ist [4].

### 2.2.1 Probleme bei Mehrfachauswahl

Wie bereits in Kapitel 2.2 angesprochen, ergibt sich bei einem Interface mit möglicher Mehrfachauswahl oft ein signifikantes Platzproblem. Zur Lösung dieses Problems gibt es verschiedene Ansätze aus [4], welche implementiert werden können.

Neben dem Platzproblem gibt es jedoch eine weitere Komplexität, mit



Abbildung 2.2: Faceted Search Interface bei Geizhals im Notebook-Bereich.



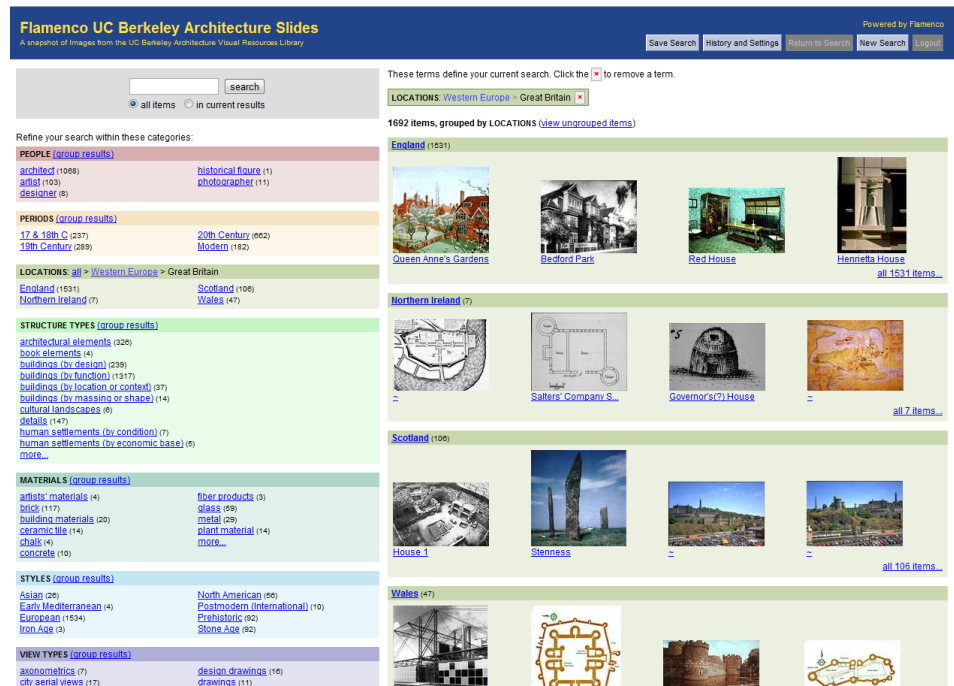
Abbildung 2.3: Bei einer Mehrfachauswahl ergibt sich eine komplexe Suchquery: Facetten werden per AND verknüpft, während Werte in den Facetten über eine OR-Beziehung verbunden werden. Im Fall dieses Beispiels ergäbe sich ein Ergebnis mit allen Objekten der Farben rot oder grün und der Größe mittel.

welcher der Benutzer unterbewusst konfrontiert ist. Ist es möglich, mehrere Werte aus verschiedenen Facetten auszuwählen, ergibt sich eine komplexe Suchanfrage. Wie in Abbildung 2.3 zu erkennen ist, stellen Benutzer unwissentlich eine boolesche AND-Verknüpfung von OR-Beziehungen zusammen. Gerade deshalb ist es bei diesen Interfaces wichtig, dem Benutzer einen guten und klaren Überblick über alle Möglichkeiten zu bieten und die Auswahl der Werte so einfach wie möglich zu gestalten.

## 2.3 Beispiele für Faceted Search Interfaces

Faceted Search Interfaces werden vielfach eingesetzt: sei es als unterstützendes Such-Interface für Bildersuche in [17], als Interface zur Suche auf mobilen





**Abbildung 2.4:** Screenshot der Suchergebnisse des finalen Interfaces des Such-Frameworks Flamenco nach Auswahl der Werte „Eastern Europe“ und „Great Britain“.

Geräten in [13] oder zur Suche nach Produkten im E-Commerce Bereich.

Um die in Abschnitt 2.2 besprochenen Probleme und Herausforderungen näher kennen zu lernen, werden in den folgenden Abschnitten drei ausgewählte Faceted Search Interfaces näher vorgestellt. Während das Framework Flamenco (siehe Abschnitt 2.3.1) einen Meilenstein in der Entwicklung von Faceted Search Interfaces darstellt, werden auch zwei aktuell auf E-Commerce-Seiten verwendete Interfaces besprochen.

### 2.3.1 Flamenco

Flamenco (siehe Abbildung 2.4) ist ein Such-Interface Framework und wurde unter der Leitung von Marti A. Hearst an der UC Berkeley von einem Forschungsteam entwickelt. Das primäre Ziel von Flamenco liegt darin, Benutzern das flexible Navigieren durch große Datenmengen zu ermöglichen [7].

Einen großen Teil des Frameworks stellt die hierarchische Abbildung der Facetten dar. Benutzer können das Suchergebnis ganz normal über Anwendung oder Entfernung von Werten vergrößern oder verkleinern und zusätzlich die Ergebnismenge über Suchbegriffe einschränken.

Zu Entwicklungs- und Testzwecken wurde Flamenco auf einer Datenbank für Architektur-Fotos implementiert. Diese Datenbank beinhaltet ca. 40.000 Bilder und für geeignete Testbedingungen wurde speziell die Zielgruppe (Architekten, Stadtplaner, usw.) als Teilnehmer für die Usability-Studien hinzugezogen.

Die Entwicklung von Flamenco begann mit einem Prototypen, welcher in einer nicht-formalen Usability Studie evaluiert wurde. Anschließend folgten zwei Entwicklungszyklen mit jeweils zwei formalen Usability-Tests [7].

Wie bei jeder Entwicklung von Such-Interfaces wurde auch bei Flamenco speziell auf die einzelnen Komponenten und ihre Auswirkung auf das Suchverhalten der Benutzer geachtet. Es wurden mehrere mögliche Strukturen getestet [3], wobei sich eine Matrixstruktur in den Usability-Studien durchgesetzt hat [2].

Neben der grundlegenden Interface-Struktur wurden auch viele andere Interface-Komponenten evaluiert, wie z.B. die beste Darstellungsart von Werten und Facetten, die Integration von Stichwort-Suche in das Faceted Search Interface oder die ideale Möglichkeit zur Darstellung hierarchischer Facetten und Werte, ohne die Seite zu überfüllen [6].

Im Interface eingebunden ist auch ein Bereich, in dem alle aktiven Werte mit zugehörigen Facetten dargestellt werden (im Laufe des Dokuments als „Current Search“-Block bezeichnet<sup>7</sup>). Dieser Bereich bietet einen schnellen Überblick über den Verlauf der Suche und ermöglicht es, schnell und einfach bereits gewählte Werte aus der Suchanfrage zu entfernen.

Flamenco verfügt neben dem „Current Search“-Block zur Unterstützung der Benutzer auch über Query-Previews. Die voraussichtliche Anzahl der betreffenden Produkte zu den einzelnen Werten werden, wie bei vielen Interfaces üblich, rechts neben den Werten in Klammern dargestellt.

Die Forschungen und Entwicklungen von Marti A. Hearst haben zur Weiterentwicklung und Verbesserung von Faceted Search Interfaces maßgeblich beigetragen und werden auch verbreitet als Leitfaden oder Starthilfe genutzt (siehe Abschnitt 2.3.2).

Flamenco wirkt auf den ersten Eindruck hin etwas veraltet. Die Farben wirken blass und in der Ausgangssituation wirkt das Interface überfüllt. Dass das Interface jedoch theoretisch gut entwickelt wurde ist daraus ersichtlich, dass Flamenco im finalen Entwicklungsstadium sechs der acht Design Kriterien Shneidermans (siehe Abschnitt 2.2) erfüllen konnte [2].

### 2.3.2 eBay

Wie auch auf anderen bekannte E-Commerce Seiten, wird auch auf eBay Faceted Search zur Produktsuche angeboten. Große Teile des aktuellen Such-Interfaces (siehe Abbildung 2.5) wurden vor einiger Zeit von eBay Express<sup>7</sup>

---

<sup>7</sup><http://pages.ebay.com/express/>

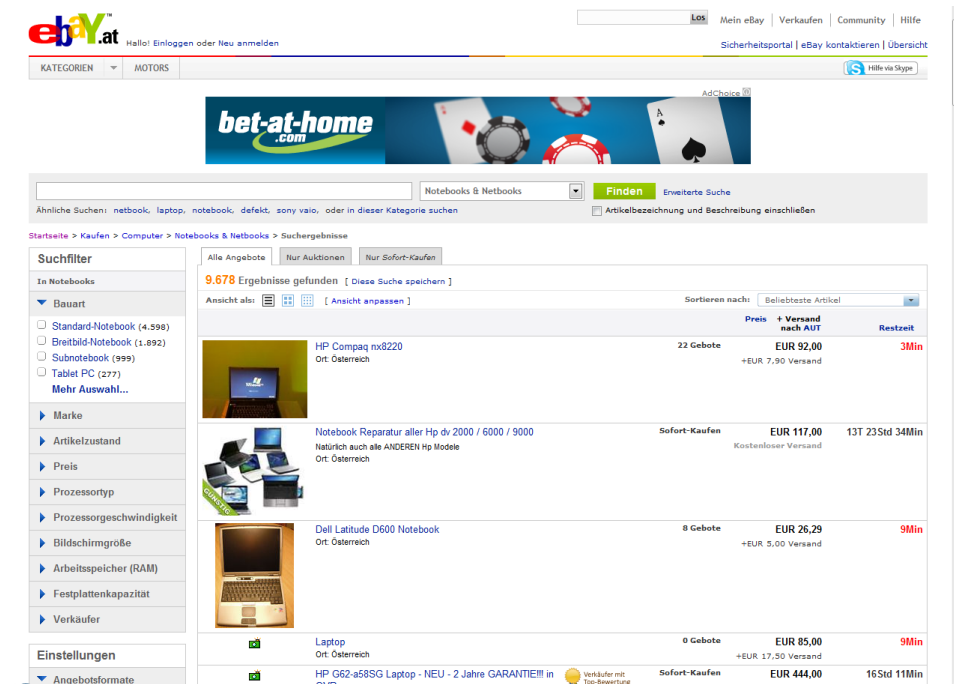


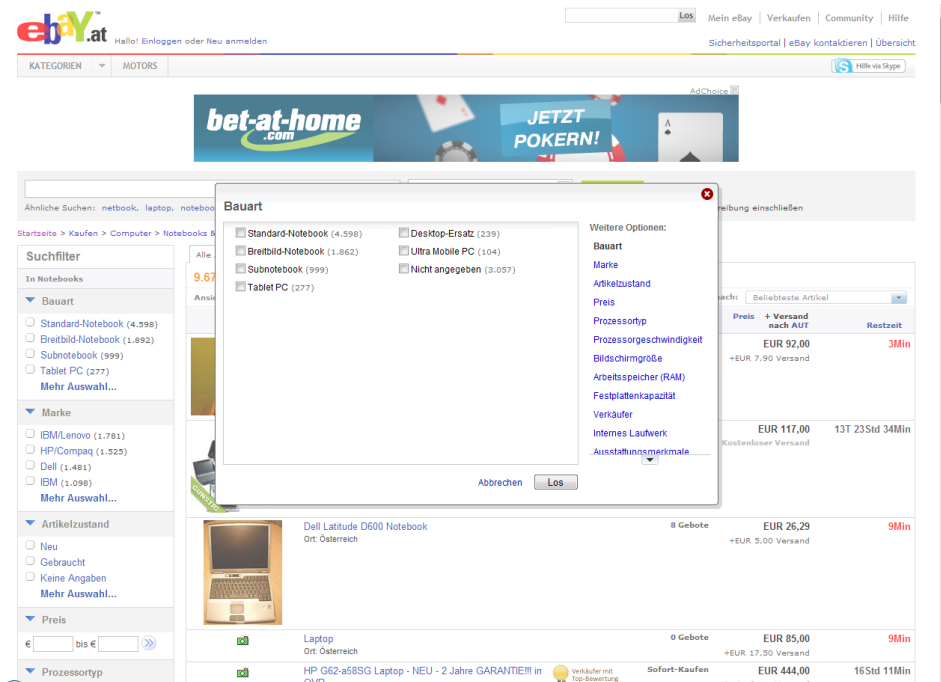
Abbildung 2.5: Das Such-Interface von eBay nach Navigation in den Notebook-Bereich.

übernommen. Dieses Such-Interface wurde im Rahmen eines Workshops in Zusammenarbeit mit Marti A. Hearst entworfen und anschließend – unabhängig von Flamenco – implementiert [8].

Interessant an diesem Interface ist vor allem die Handhabung mit der Menge an Werten. Wie in Abschnitt 2.2 erklärt wurde, stellt eine große Anzahl von Werten bzw. Facetten ein großes (Platz-)Problem dar. Bei eBay wurde dies auf zwei verschiedene Varianten gelöst:

Bei der **Einfachauswahl** werden die fünf relevantesten Werte in einer Listansicht als Links dargestellt, mit der Möglichkeit alle Werte per Klick auf einen „Mehr“-Link anzuzeigen und bei Klick auf einen „Weniger“-Link wieder zu verstecken. Die Wahrscheinlichkeit ist hoch, dass von der Mehrheit der Benutzer gesuchte Werte bereits in der Kurzansicht vorhanden sind.

Wird in Facetten eine **Mehrfachauswahl** angeboten, werden die vier relevantesten Werte in einer Listansicht mit Checkboxes angezeigt. Sollte sich der gewünschte Wert nicht in der Vorschau befinden, öffnet sich bei Klick auf den Link „Mehr Auswahl...“ ein Popup mit allen vorhandenen Werten (siehe Abbildung 2.6). Gewünschte Werte können dann per Auswahl der Checkboxes zum Suchfilter hinzugefügt werden.



**Abbildung 2.6:** Popup mit allen Werten mit einer Mehrfachauswahl-Option im Such-Interface von eBay.

### 2.3.3 Geizhals

Das Such-Interface von Geizhals (siehe Abbildung 2.7) ist ein gutes Beispiel für auftretende Probleme bei der Implementierung eines Faceted Search Interfaces mit der Möglichkeit der Mehrfachauswahl von Werten (siehe Abschnitt 2.2.1).

In diesem Interface werden alle Facetten untereinander aufgelistet und alle möglichen Werte zu den Facetten angezeigt. Dies führt bei einer großen Menge von Facetten und/oder Werten zu einem sichtbaren Platzproblem. Es ist hier nicht möglich, das komplette Interface ohne scrollen der Seite zu bedienen und es ist fraglich, ob sich Personen ohne Fachwissen in diesem Interface problemlos zurechtfinden.

Bei Geizhals werden prinzipiell alle Möglichkeiten der Wertauswahl angeboten. Je nach Facette bekommt der Benutzer die Möglichkeit zur Mehrfach- oder Einfachauswahl. Aktive Werte werden je nach Auswahlmöglichkeit verschieden hervorgehoben. Aktive Werte werden bei einer Mehrfachauswahl fett mit Link dargestellt, alle anderen Werte derselben Facette können per Link hinzugefügt werden. Aktive Werte bei einer Einfachauswahl werden auch fett dargestellt, jedoch sind die Hyperlinks von allen Werten in der Facette verschwunden.

The screenshot shows the Geizhals website's search interface for 'Notebooks & Tablets' with a filter for '13.1" bis 16.4" TFT ASUS'. The interface includes a navigation menu on the left, a search bar at the top, and a list of filter criteria on the right. The filter criteria are as follows:

- Hersteller:** ASUS (159), Acer, Apple, Archos, Belinea, Dell, Fujitsu, HP, Compaq, Lenovo, IBM, MSI, Nexos, Panasonic, Samsung, Sonosite, Sony, Toshiba, Alla
- Display-Größe:** 13.3" (14), 14" (9), 14.1" (3), 15.4" (9), 15.6" (108), 16" (16)
- Display-Format:** 16:10 (5), 16:9 (154)
- Display-Auflösung:** 1280x800 (4), 1366x768 (153), 1680x1050 (1), 1920x1080 (1)
- Display-Pixelabstand:** bis 0,180mm (1), bis 0,200mm (2), bis 0,220mm (14), bis 0,230mm (28), bis 0,260mm (159)
- Display-Typ:** LCD, aktivmatt (elare) (131), LCD, matt (non-glare) (28)
- CPU-Hersteller:** AMD (133), Intel (146)
- CPU-Kerne:** Single-Core (2), Dual-Core (135), Triple-Core (1), Quad-Core (21)
- CPU-Takt:** ab 1,5GHz (159), ab 1,5GHz (151), ab 2GHz (130), ab 2,3GHz (37)
- CPU-Typ:** Athlon II (10), Core 2 Duo (11), Core i3 (38), Core i5 (52), Core i7 (19), Pentium Dual-Core (11), Pentium P (15), Phenom II (3)
- Hauptspeicher:** ab 1GB (159), ab 2GB (157), ab 3GB (129), ab 4GB (122), ab 5GB (5), ab 8GB (3)
- Anzahl Speichermodule:** 1x (53), 2x (105), unbekannt (1)
- Festplattenkapazität:** ab 120GB (159), ab 250GB (156), ab 320GB (155), ab 500GB (78), ab 1000GB (3)
- Festplattentyp:** HDD (159)
- Optisches Laufwerk:** Blu-ray (BD-R/BE/RW) (1), Blu-ray (BD-ROM) (24), DVD±/RW (123), ohne (11)
- Grafik (Hersteller/Typ):** ATI (dediziert) (32), ATI (IGP) (4), Intel (IGP) (38), NVIDIA (dediziert) (70), NVIDIA (IGP) (12), unbekannt (3)
- GPU-Serie:** IGP (41), ATI Radeon HD (4), Mobility Radeon HD 3 (26), Mobility Radeon HD 6 (6), Intel GMA X4 (7), Intel GMA HD (26), GeForce 2 (3), GeForce 3 (1), GeForce GT (43), GeForce G72 (3), GeForce G73 (7), GeForce G74 (9), GeForce G75 (3), GeForce G76 (3)
- Anschlüsse:** USB (159), FireWire (12), Modem (6), GB-LAN (159), WLAN 802.11n (159), Bluetooth (62), eSATA (54), DVI (3), HDMI (136), DisplayPort (12), PCMCIA (1), ExpressCard (65)
- Anzahl USB:** ab 2x (159), ab 3x (147), ab 4x (43)
- Card Reader:** 2in1 (2), 3in1 (46), 4in1 (39), 5in1 (33), Zin1 (3), 8in1 (36)
- Auflösung Webcam:** ohne (24), 0,3 Megaapixel (59), 1,3 Megaapixel (15), 2,0 Megaapixel (53), unbekannt (8)
- Betriebssystem:** Windows 7 Home Premium (119), Windows 7 Professional (29), Windows Vista Home Premium (3), Windows Vista Ultimate (2), FreeDOS (5), ohne Betriebssystem (1), Linux (159)
- Akkutyp:** bis 2ka (15), bis 2.5ka (28), bis 3ka (144), bis 4ka (159)
- Gewicht:** bis 2ka (15), bis 2.5ka (28), bis 3ka (144), bis 4ka (159)
- Besonderheiten:** USB 3.0 (37), LED (140), 3D-Readv (1), Webcam (135), UMTS (7), HSPA/HSDPA/HSUPA (2), FingerPrint Reader (6), Multi-Touch Trackpad (57)

Abbildung 2.7: Startansicht des Such-Interfaces von Geizhals im Usability-Test.

Um Werte wieder zu entfernen zu können, muss der „alle“-Button am Ende der Facetten-Zeile betätigt werden, bzw. ist es möglich bei Werten mit Mehrfachauswahl dem Hyperlink am Wert erneut zu folgen, um ihn zu entfernen.

Das Such-Interface von Geizhals wurde als Ausgangsinterface für die Entwicklung der neuen Interfaces bestimmt. Wie die Interfaces im Endeffekt gegen das Interface von Geizhals abgeschnitten haben, kann in Kapitel 5 nachgelesen werden.

# Kapitel 3

## Design

Das Ziel dieser Arbeit ist, wie in Kapitel 1 angesprochen, die Entwicklung eines neuartigen Faceted Search Interfaces. Zu diesem Zweck musste im Vorfeld bestimmt werden, welche Komponenten des Interfaces wichtig sind und auf welche Faktoren speziell geachtet werden muss. Folgende Schlüsselkomponenten wurden daher zur Entwicklung eines neuen Interfaces identifiziert:

- Position des Interfaces
- Anordnung der Komponenten
- Darstellung der Facetten
- Darstellung der Werte
- Verhalten des Interfaces bei Interaktion
- Darstellung der Suchergebnisse

Um nun bei der anschließenden Evaluierung der selbstentwickelten Komponenten realistische Ergebnisse zu erzielen und bei Faktoren, wie der Position des Interfaces oder der Anordnung der Komponenten, Vergleiche ziehen zu können, wurden zwei verschiedene Faceted Search Interfaces entwickelt: das Blockinterface und das Ellipseninterface. Beide Interfaces verfügen über gewissen Ähnlichkeiten, wie die serverseitige Implementierung oder die Darstellung der einzelnen Facetten, unterscheiden sich jedoch in der Position des Interfaces und der Anordnung der einzelnen Komponenten.

Das **Blockinterface** stellt den Versuch dar, empfohlene Interfacekomponenten mit neuartigen Funktionen zu verknüpfen (siehe Abschnitt 3.2). Im **Ellipseninterface** hingegen wurde versucht, einen gänzlich neuen Ansatz zur Strukturierung der einzelnen Interfacekomponenten zu finden (siehe Abschnitt 3.3).

### 3.1 Übergreifende Designentscheidungen

Bereits vor der Implementierung der Interfaces wurden die meisten wichtigen Designentscheidungen getroffen. Einige der Entscheidungen haben sich auf

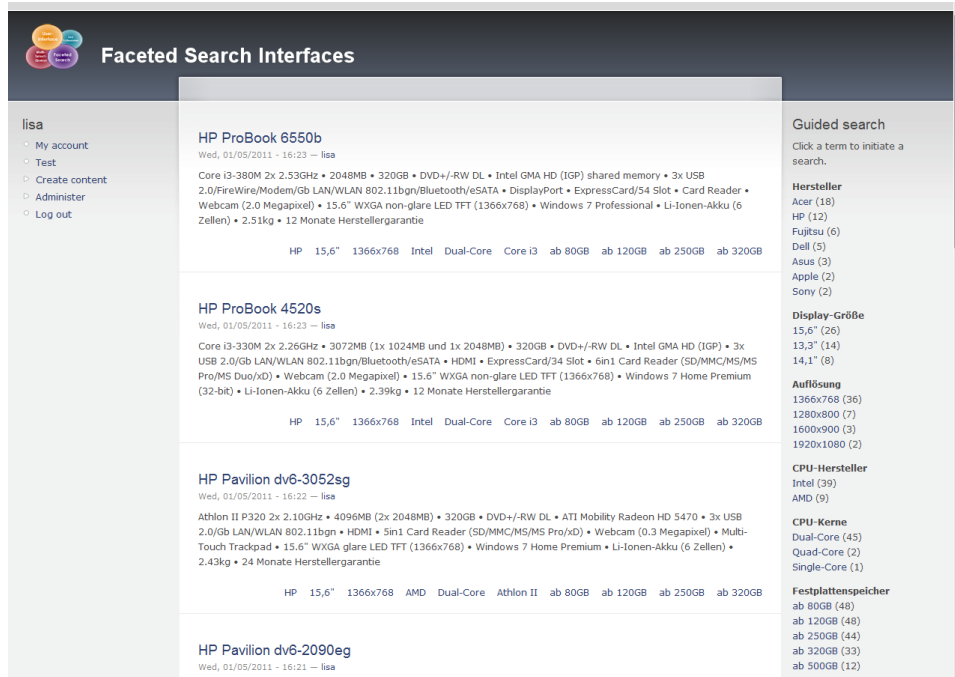
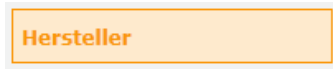


Abbildung 3.1: Such-Interface des Drupal Moduls Faceted Search.

beide Interfaces ausgewirkt, während andere Entscheidungen spezifisch für das jeweilige Interface getroffen wurden.

Beide Interfaces basieren im Design grundsätzlich auf dem Standardinterface des Drupal Faceted Search Moduls (siehe Abbildung 3.1). Von diesem Interface ausgehend wurden alle nötigen Änderungen durchgeführt, um die gewünschten Funktionen und Komponenten der Interfaces umsetzen zu können.

Da die Wahl der Auswahlmöglichkeiten im Interface die Anforderungen an das jeweilige Interface grundlegend verändern, wurde die Entscheidung für das Angebot einer Mehrfachauswahl von Werten in beiden Interfaces bald getroffen. Wie bereits in Abschnitt 2.2 angeführt, gibt es in Interfaces mit der Möglichkeit der Mehrfachauswahl gewisse Herausforderungen und Probleme, die während des Designprozesses berücksichtigt werden müssen. Auch in [9] wird bestätigt, dass für verschiedene Arten von Aufgaben auch verschiedene Interfaces bevorzugt werden – d. h. je nach Aufgabe verändern sich die Anforderungen an das Interface. Aus diesem Grund ist es wichtig, den Verwendungszweck des Interfaces bereits im Vorhinein zu definieren.



**Abbildung 3.2:** Die übersichtliche Block-Darstellung einer Facette.

### 3.1.1 Darstellung der Facetten

Die vermutlich größte Herausforderung stellt bei Interfaces mit Mehrfachauswahl oder hierarchischen Facetten das Platzproblem dar. Wird dem Benutzer ermöglicht, alle Werte einer Facette zur Suchanfrage hinzuzufügen, muss der Benutzer logischerweise auch Zugang zu allen Werten erhalten. Hat man alle Werte und Facetten konstant sichtbar im Interface, läuft man Gefahr, die Übersichtlichkeit des Interfaces zu opfern. Es muss also ein Weg gefunden werden, dass der Benutzer nicht lange nach den gewünschten Werten suchen muss, das Interface jedoch trotzdem übersichtlich und strukturiert erscheint.

Bei beiden Interfaces wurde daher der Weg gewählt, dem Benutzer die Entscheidung zu überlassen, ob er die zugehörigen Werte zu einer Facette nun anzeigen möchte oder nicht. Dafür wurden die Werte zu den Facetten gruppiert und können per Klick auf die jeweilige Facette angezeigt oder versteckt werden.

Für die Darstellung wurde basierend auf [4] entschieden, zumindest die Facetten-Überschriften in übersichtlichen Blöcken darzustellen (siehe Abbildung 3.2). Die Facetten werden so voneinander abgegrenzt und liefern durch die einheitliche Größe einen übersichtlichen und strukturierten Gesamteindruck. Geachtet wurde bei den Facetten auch darauf, den gesamten Facetten-Block klickbar zu machen und nicht nur die Facetten-Überschrift mit einem Hyperlink zu versehen.

Neben der blockweisen Anordnung der Facetten wurden die verschiedenen Facetten zur Verbesserung der Übersichtlichkeit farblich getrennt. Diese farbliche Trennung wurde bereits bei Flamenco überwiegend eingesetzt (siehe Abschnitt 2.3.1) und erhöht eine schnelle Assoziation der zugehörigen Werte, wenn diese zusätzlich farblich zugeteilt werden. Dies erscheint vorrangig nicht so wichtig, wenn man nur die Ansicht des normalen Suchinterfaces alleine betrachtet. Es ermöglicht jedoch den Benutzern, bei einer Auflistung der Werte und Facetten an einer anderen Stelle auf der Seite, wie z. B. im „Current Search“-Block (siehe Abschnitt 3.1.2), die Werte schnell und intuitiv den zugehörigen Facetten zuordnen zu können.

#### Aktive Facetten

Bei der Entwicklung der Interfaces wurden der Darstellung der aktiven Facetten und Werte eine hohe Bedeutung zugemessen. Ein signifikantes Problem des Interfaces von Geizhals ist die unterschiedliche Hervorhebung von aktiven Facetten und Werten (siehe Abschnitt 2.3.3). Es wird unterschieden,



<input type="checkbox"/>	<b>Hersteller:</b>	ASUS <b>Acer</b> (277) Apple Archos Belinea Dell Fujitsu HP Compag Lenovo IBM MSI Nexoc Panasonic Samsung <u>Sonstige</u> Sony Toshiba Alle
<input type="checkbox"/>	<b>Display-Größe:</b>	13.3" (28) <b>14"</b> (19) 14.1" (7) 15.4" (7) 15.6" (216)
<input type="checkbox"/>	<b>Display-Format:</b>	16:10 (7) <b>16:9</b> (270)
<input type="checkbox"/>	<b>Display-Auflösung:</b>	1280x800 (6) <b>1366x768</b> (257) 1680x1050 (1) unbekannt (13)
<input type="checkbox"/>	<b>Display-Pixelabstand:</b>	bis 0.200mm (1) <b>bis 0.220mm</b> (29) bis 0.230mm (54) bis 0.240mm (55) bis 0.260mm (277)
<input type="checkbox"/>	<b>Display-Typ:</b>	LCD glänzend (glare) (147) <b>LCD matt (non-glare)</b> (121) unbekannt (9)
<input type="checkbox"/>	<b>CPU-Hersteller:</b>	AMD (24) <b>Intel</b> (253)
<input type="checkbox"/>	<b>CPU-Kerne:</b>	Single-Core <b>Dual-Core</b> (277) Triple-Core Quad-Core Alle
<input type="checkbox"/>	<b>CPU-Takt:</b>	ab 1GHz (277) <b>ab 1.5GHz</b> (271) ab 2GHz (253) ab 2.5GHz (88)
<input type="checkbox"/>	<b>CPU-Typ:</b>	AMD C- (2) AMD E- (11) Athlon II (10) Celeron Dual-Core (1) <b>Core 2 Duo</b> (19) Core i3 (52) Core i3-2 (15) Core i5 (99) Core i5-2 (24) Core i7 (4) Pentium Dual-Core (24) Pentium P (15) Turion X2 (1)
<input type="checkbox"/>	<b>Hauptspeicher:</b>	ab 2GB (277) ab 3GB (235) ab 4GB (218) ab 6GB (18) ab 8GB (16)
<input type="checkbox"/>	<b>Anzahl Speichermodule:</b>	1x (143) <b>2x</b> (134)
<input type="checkbox"/>	<b>Festplattenkapazität:</b>	ab <b>250GB</b> (277) ab 320GB (259) ab 500GB (159) Alle

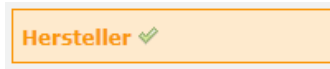
**Abbildung 3.3:** Darstellung von aktiven Werten und Facetten in Geizhals' Faceted Search Interface. Aktiv sind in diesem Fall die Werte „Acer“ der Facette „Hersteller“, „Dual-Core“ der Facette „CPU-Kerne“ und „ab 250GB“ der Facette „Festplattenkapazität“.

ob in der Facette eine Einfach- oder Mehrfachauswahl möglich ist. Je nach Art der Auswahl werden die aktiven Werte unterschiedlich dargestellt. Diese inkonsistente Darstellung kann bei Benutzern offensichtlich zu Verwirrung oder Unklarheit führen. Weiters werden aktive Facetten bei Geizhals optisch nicht hervorgehoben und es gibt auch keine separate Auflistung aktiver Facetten. Um nun zu bestimmen, welche Facetten aktiv sind, muss der Benutzer nach aktiven Werten suchen und die Werte den Facetten zuordnen (siehe Abbildung 3.3).

Bei der Entwicklung des Block- und Ellipseninterfaces wurde daher entschieden, zwischen den Darstellungsarten keinen Unterschied zu machen und dadurch die Konsistenz des Interfaces zu erhöhen. Außerdem soll der Benutzer mit einem Blick erkennen, welche Facetten aktiv am Suchergebnis beteiligt sind. Durch die Möglichkeit, die Werte von Facetten manuell verstecken zu können, muss also bereits in der Facetten-Überschrift ein Indikator vorhanden sein, der auf aktive Werte in der Facette hinweist. Dies soll gewährleisten, dass aktive Facetten immer sofort erkennbar sind, auch wenn der Benutzer zufällig alle Werte versteckt.

Die Indikation der aktiven Facette ist z. B. durch die Hervorhebung der Facette mittels einer grafischen Modifikation, wie Fettschrift oder eine Änderung der Farbe, möglich. Durch die bereits hohe Farbenvielfalt wurde entschieden, dass dies nur zu Verwirrung führen würde und auch die Überschrift in Fettschrift war nach einigen Versuchen im Prototyp als zu unscheinbar eingestuft worden.

Als Alternative für diese grafischen Modifikationen würde sich anbieten, neben der Facetten-Überschrift die Anzahl der aktiven Facetten in derselben Klammerdarstellung wie die Query-Previews der Werte darzustellen. Auch die Anzahl aller zutreffenden Produkte mit einem aktiven Wert der Facette



**Abbildung 3.4:** Kennzeichnung einer aktiven Facette durch die Aktivierung eines Icons in der Facetten-Überschrift.

in Klammerdarstellung stellt eine Möglichkeit dar. Beide Versionen würden jedoch sehr wahrscheinlich zur Verwirrung des Benutzers führen und die Konsistenz des Interfaces in Frage stellen. Die Zahlen in den Klammern in den Facetten-Überschriften hätten eine andere Bedeutung wie jene in den eigentlichen Query-Previews und es ist zweifelhaft, ob jeder Benutzer den Unterschied erkennen kann. Auch diese Varianten wurden deshalb nicht weiter verfolgt.

Im finalen Design wurden die aktiven Facetten nicht über passive Designeigenschaften, sondern durch die Anzeige eines Bildes grafisch hervorgehoben. Dafür wird für aktive Facetten in der Überschrift ein Icon aktiviert (siehe Abbildung 3.4), das die aktiven Facetten von den anderen trennen soll.

### 3.1.2 Komponenten der Interfaces

Das Interface des Faceted Search Moduls verfügt standardmäßig über einen „Guided Search“-Block (das Hauptinterface) und einen „Current Search“-Block. Neben dem Hauptinterface inklusive aller Funktionen wurde auch der „Current Search“-Block als hilfreich betrachtet und als Komponente in das Design der neuen Interfaces übernommen.

Ein Benutzer soll jederzeit das Verhalten des Interfaces verstehen und zusätzlich das Gefühl der Kontrolle vermittelt bekommen [4]. Dafür ist es notwendig, dem Benutzer klare und schnell erkennbare Hilfen zur Verfügung zu stellen. Der „Current Search“-Block ist eine solche Hilfe – er liefert dem Benutzer zu jeder Zeit einen schnellen Überblick, welche Werte von welchen Facetten im Moment das Suchergebnis beeinflussen. Dies ist wichtig, damit der Benutzer nicht plötzlich vor unerklärlichen Ergebnissen steht und gleichzeitig die Möglichkeit hat, die aktiven Werte direkt über den „Current Search“-Block zu entfernen.

Im Faceted Search Modul bietet der „Current Search“-Block, neben einer Auflistung aller aktiven Werte inklusive deren Facetten, auch die gerade angesprochene Möglichkeit zur Entfernung der Werte (siehe Abbildung 3.5). Es wurde angenommen, dass eine Entfernung der Werte im „Current Search“-Block sehr wahrscheinlich bevorzugt wird, im Gegensatz zur Entfernung der Werte im Interface. Der Grund dafür ist die kompakte Darstellung aller aktiven Werte auf einen Blick, da keiner der Werte zuerst im Interface gesucht werden muss.



**Abbildung 3.5:** Darstellung des „Current Search“-Blocks des Drupal Faceted Search Moduls.

Wie der „Current Search“-Block sind auch die Query-Previews ein Hilfsmittel zur Orientierung im Interface. Für beide neu entwickelten Interfaces ist die ursprünglich implementierte Version der Query-Previews des Drupal Moduls übernommen worden. Für jeden Wert wird bereits im Vorhinein die Anzahl der Produkte mit diesem, in Kombination mit bereits aktiven Werten, berechnet und in Klammer neben dem Wert angezeigt. Der Benutzer kann somit den Ausgang der Suchanfrage vorhersagen und kann – abhängig von den Werten im Query-Preview – zu erfolgreicherer Suchergebnissen geleitet werden.

Zur Unterscheidung der einzelnen Facetten sind diese, wie bereits in Abschnitt 3.1.1 kurz angesprochen, farblich voneinander getrennt worden. Neben der farbigen Darstellung im Interface wurde die einheitliche Farbgebung der einzelnen Facetten auch im „Current Search“-Block übernommen. Zwar wird im „Current Search“-Block bereits der Name der zugehörigen Facette zum Wert angezeigt, jedoch soll die farbliche Übereinstimmung noch einmal helfen, verschiedene Werte mit ihren zugehörigen Facetten schnell assoziieren zu können.

Während die Funktionsweise und die farbliche Gestaltung der Einträge im „Current Search“-Block in beiden Interfaces gleich ist, unterscheidet sich die Darstellung an sich. Wie genau die Darstellung in beiden Interfaces gelöst wurde, kann in den Abschnitten 3.2.1 und 3.3.1 nachgelesen werden.

### 3.1.3 Suchergebnisse

Ein Such-Interface sollte grundsätzlich zwei Schlüsselfunktionen erfüllen. Einerseits soll das Interface Unterstützung bei der Formulierung einer Suchanfrage bieten, andererseits die aus dieser Suchanfrage entstandenen Suchergebnisse darstellen [9]. Es gibt mittlerweile einige Ansätze, dass eine Suche zuerst über eine einmalige Suchanfrage gestartet wird und die restliche Navigation in den Suchergebnissen bzw. Erweiterung und Einschränkung der Suchanfrage über die Darstellung der Suchergebnisse durchgeführt werden kann. Genauer dazu kann in [10–12] nachgelesen werden.

Beide Komponenten können maßgeblich zur Erfolgsquote des Interfaces beitragen, jedoch wurde bei der Entwicklung der beiden neuen Interfaces nur



**Abbildung 3.6:** Darstellung der Suchergebnisse.

darauf geachtet, dass die Anzeige der Suchergebnisse lediglich eine unterstützende Funktion zum Interface an sich bietet. Die Suchergebnisse wurden rein für Testzwecke so gestaltet, dass der Benutzer einen oberflächlichen Überblick über die gefundenen Objekte bekommt, das Interface selbst aber im Mittelpunkt der Benutzeraktivität steht. Während die Interaktion mit den Suchergebnissen für den Benutzer wichtig ist, ist die Interaktion mit den Facetten mindestens genauso wichtig, wenn nicht wichtiger [16].

Für die Darstellung der Suchergebnisse wurde die Standardausgabe des Faceted Search Moduls direkt übernommen und nur Kleinigkeiten im Design geändert. Die von Drupal automatisch ausgegebenen Tags der Produkte (siehe Abschnitt 4.2) wurden in einer kleinen Schriftgröße neben die Produktüberschriften gesetzt (siehe Abbildung 3.6). Mit Hilfe der Tags soll schnell sichtbar sein, über welche Eigenschaften das jeweilige Produkt verfügt.

### 3.1.4 Konsistenz

Bei der Gestaltung eines berechenbaren Interfaces ist es wichtig, dass vor allem auf Konsistenz im Interface geachtet wird [15]. Diese Konsistenz wird in den neu entwickelten Interfaces neben der identen Darstellung von aktiven Werten (siehe Abschnitt 3.1.1) dadurch erreicht, dass das Interface in beiden neuen Entwürfen nie automatisch nachgeladen oder verändert wird, außer die Seite wird vom Benutzer aktiv im Browser neu geladen.

Durch den Einsatz von Ajax<sup>1</sup> (siehe Abschnitt 4.2.3) ist es möglich, die Interface-Komponenten statisch aussehen zu lassen, während spezifische Teile des Interfaces bei einer Änderung der Suchanfrage einzeln aktualisiert werden. Im Fall der beiden neuen Interfaces sind dies die Suchergebnisse, die Anzahl der zutreffenden Produkte in den Query-Previews und der „Current Search“-Block.

Wird eine Suche gestartet oder werden zu einer bestehenden Suchanfrage

<sup>1</sup>[http://en.wikipedia.org/w/index.php?title=Ajax\\_\(programming\)&oldid=434319538](http://en.wikipedia.org/w/index.php?title=Ajax_(programming)&oldid=434319538), Kopie auf CD-ROM (Datei docs/Ajax.pdf vom 23.06.2011)

Werte hinzugefügt oder entfernt, wird die aktuelle Suchanfrage im Hintergrund abgesendet und die Liste der Suchergebnisse auf die neue Suchanfrage angepasst. Gleichzeitig werden die Zahlen in den Query-Previews ausgehend von der neuen Suchanfrage neu berechnet und angezeigt. Zusätzlich wird der „Current Search“-Block aus allen aktiven Werten und Facetten neu generiert und die Anzeige aktualisiert.

Während der gesamten Interaktion mit dem Interface wird die Seite nie neu geladen. Der Benutzer findet sich daher nie in der Situation, dass seine momentane Position an der Seite automatisch durch das Nachladen der Seite verändert wird oder sich das Interface optisch verändert. Es werden immer dieselben Werte und Facetten angezeigt und die gesamte Darstellung (Anzeigen und Verstecken von Werten u. Ä.) kann rein nur durch Aktionen vom Benutzer gesteuert und verändert werden.

Gleichzeitig können damit andauernde Ladevorgänge bei einer großen Datenmenge eliminiert werden. Das Hinzufügen oder Entfernen von Werten kann direkt ohne Warten durchgeführt werden und bewahrt den Benutzer somit auch vor der Situation, dass Aktionen während langen Ladezeiten durchgeführt und dadurch nicht registriert werden.

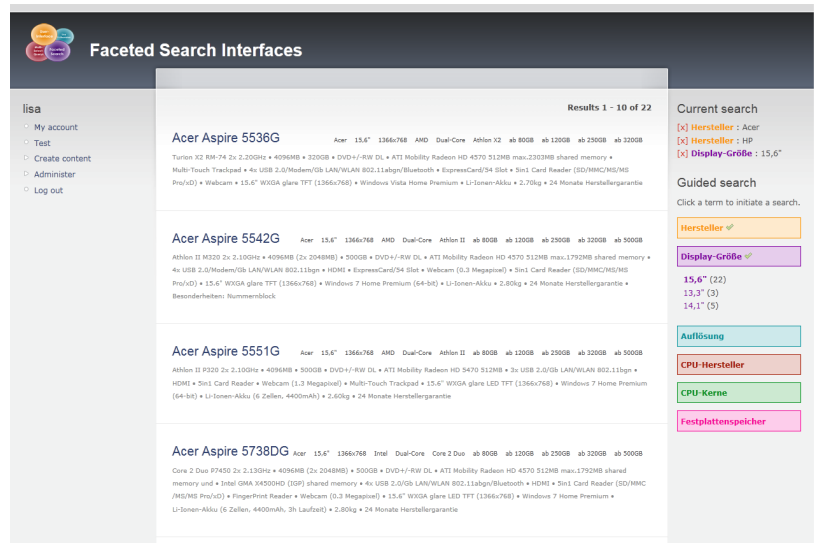
## 3.2 Blockinterface

Das Blockinterface (siehe Abbildung 3.7) ist das erste der zwei neu entwickelten Interfaces. Im Unterschied zum Ellipseninterface wurde bei der Komposition der einzelnen Interface-Elemente versucht, die klassische Richtung einzuschlagen und mit kleinen aber hilfreichen Änderungen ein besseres Interfaces zu gestalten.

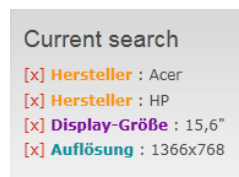
### 3.2.1 Anordnung der Elemente

Bei der Entwicklung eines Interfaces spielt neben anderen wichtigen Komponenten auch die Platzierung des Interfaces an sich eine große Rolle. Laut [4] ist die beste Position für Navigation und Suchinterfaces der obere, mittlere Bereich einer Seite. Wird das Interface dort platziert, sollte aber gewährleistet werden, dass das Interface nicht den gesamten Seitenbereich einnimmt und zur Ansicht der ersten paar Suchergebnisse schon gescrollt werden muss. Die Anzeige der Suchergebnisse im aktuellen Seitenbereich (ohne scrollen zu müssen) ist ein etabliertes Prinzip im Usability-Bereich [4] und sollte auch im Blockinterface eingehalten werden.

Für das Blockinterface wurde daher entschieden, das Interface nicht im oberen mittleren Bereich anzubringen, sondern, wie bei anderen bekannten E-Commerce-Seiten wie eBay oder Amazon, einen der zwei Seitenränder zu verwenden und statt dessen den gesamten mittleren Bereich für die Suchergebnisse zu reservieren.



**Abbildung 3.7:** Ansicht des Blockinterfaces mit bereits aktiven Werten in der Suchanfrage.



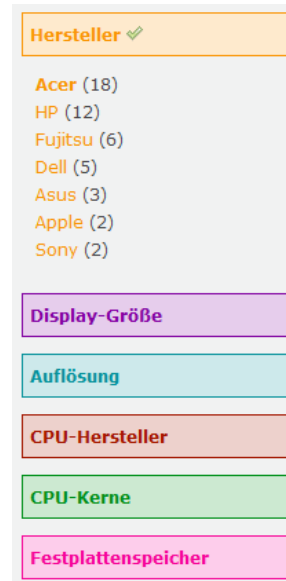
**Abbildung 3.8:** Darstellung der aktiven Werte und Facetten im „Current Search“-Block im Blockinterface.

Das Interface wurde schlussendlich am rechten Seitenrand platziert. In der Regel sind Interfaces wenn seitlich, dann links angeordnet (siehe Abschnitt 2.3.2). Grund für die doch unübliche Seitenwahl war die Überlegung, ob die Wahl des Seitenrandes nicht obsolet ist, solange das Interface an sich funktioniell und verständlich ist (siehe Abschnitt 5.4.5).

### Anordnung und Darstellung des „Current Search“-Blocks

Wie bereits in Abschnitt 3.1.2 angesprochen, wird in diesem Interface ein „Current Search“-Block für die zusätzliche Darstellung aktiver Facetten und Werte verwendet (siehe Abbildung 3.8).

Der „Current Search“-Block wird im Blockinterface direkt am oberen Ende des rechten Seitenbereiches über dem eigentlichen Interface angezeigt. Dies stellt sicher, dass die aktuell gewählten Werte immer genau an derselben fixen Position, nämlich ganz oben, gefunden werden können und der



**Abbildung 3.9:** Darstellung der Facetten in übersichtlichen Blöcken inklusive Anzeige der Facette „Hersteller“ mit aktivem Wert „Acer“.

Benutzer durch einen einzigen Scrollvorgang zum Block gelangen kann. Würde der „Current Search“-Block unter dem eigentlichen Interface positioniert werden, würde sich die Position des Blockes auf der Seite je nach Menge der angezeigten oder versteckten Werten verändern und die Konsistenz und Berechenbarkeit des gesamten Interfaces darunter leiden.

Im „Current Search“-Block werden die aktiven Werte, nach Facetten geordnet, untereinander angezeigt. Die Werte verfügen neben dem reinen Informationsgehalt auch über die Möglichkeit, per Klick auf [X] den zugehörigen Wert aus der Suchanfrage entfernen zu können.

### Anordnung der Facetten

Wie in Abschnitt 3.2 kurz erwähnt, wurden in diesem Interface alle Komponenten klassisch angeordnet. Die einzelnen geblockten Facetten werden vertikal an einer Linie ausgerichtet angezeigt (siehe Abbildung 3.9). Durch diese symmetrische Darstellung der Facetten wirkt das Interface klar und übersichtlich.

Die Werte einer Facette können durch Klick auf die verlinkte Facetten-Überschrift angezeigt oder versteckt werden. Wie in Abschnitt 3.1.1 besprochen wurde, kann der Benutzer somit steuern, welche Werte von welchen Facetten angezeigt oder versteckt werden.

Um eine unbestimmte Vergrößerung des Interfaces zu vermeiden, werden bei einem Klick auf eine bestimmte Facette die Werte aller anderen Facetten

versteckt. Dadurch soll gewährleistet werden, dass der Benutzer alle Facetten und auch den „Current Search“-Block immer im Überblick bzw. im Sichtfeld behält.

### 3.2.2 Darstellung der Werte

Im Ausgangszustand des Blockinterfaces sind alle Werte versteckt. Der Benutzer kann per Klick auf die gewünschte Facette alle Werte dieser Facette anzeigen oder verstecken.

Die Werte im Blockinterface werden, wie auch die Facetten, untereinander an einer Linie ausgerichtet aufgelistet. Für eine einfache Zuordnung zur übergeordneten Facette werden die Werte außerdem in der Farbe der Facette dargestellt (siehe Abbildung 3.9).

Zur Darstellung der Werte wurde in aktiven Faceted Search Interfaces recherchiert (siehe Abschnitt 2.3.2 und 2.3.1). Es scheint sich etabliert zu haben, Werte als einfache Hyperlinks dazustellen. Durch die gleiche Darstellung über mehrere verschiedene Interfaces hinweg erreicht man eine gewisse übergreifende Konsistenz. Durch die Verwendung von bekannten und verbreiteten Interface-Elementen kann neuen Benutzern somit den Einstieg in die Bedienung des Interfaces deutlich erleichtert werden.

Um einen bestimmten Wert der Suchanfrage hinzuzufügen, reicht es, dem Hyperlink des gewählten Wertes zu folgen. Die anschließende Darstellung von aktiven Werten wird im Blockinterface durch Fettschrift als Unterscheidung von anderen Werten erreicht. Auch aktive Werte verfügen immer noch über einen Hyperlink, welcher bei Klick den gewählten Wert aus der Suchanfrage entfernt.

## 3.3 Ellipseninterface

Bei der Entwicklung des Ellipseninterfaces (siehe Abbildung 3.10) ist absichtlich ein anderer Weg eingeschlagen worden. Während das Blockinterface von einer klassischen Darstellung abgeleitet wird, wurde im Ellipseninterface versucht, eine innovative Anordnung der Interface-Elemente umzusetzen.

### 3.3.1 Anordnung der Elemente

Wie bereits in Abschnitt 3.2.1 erwähnt, wird der obere mittlere Bereich einer Seite als beste Position für Navigation und Such-Interfaces empfohlen [4]. Im Ellipseninterface wurde versucht, das Interface genau in diesem Bereich zu positionieren.

Wurde für das Interface diese Position festgelegt, muss ein Weg gefunden werden, auch bei einer großen Menge von Facetten und Werten trotzdem noch genug Platz für zumindest ein paar Suchergebnisse zu finden. Das große Problem von Geizhals ist, dass es dort eben nicht mehr möglich ist, während



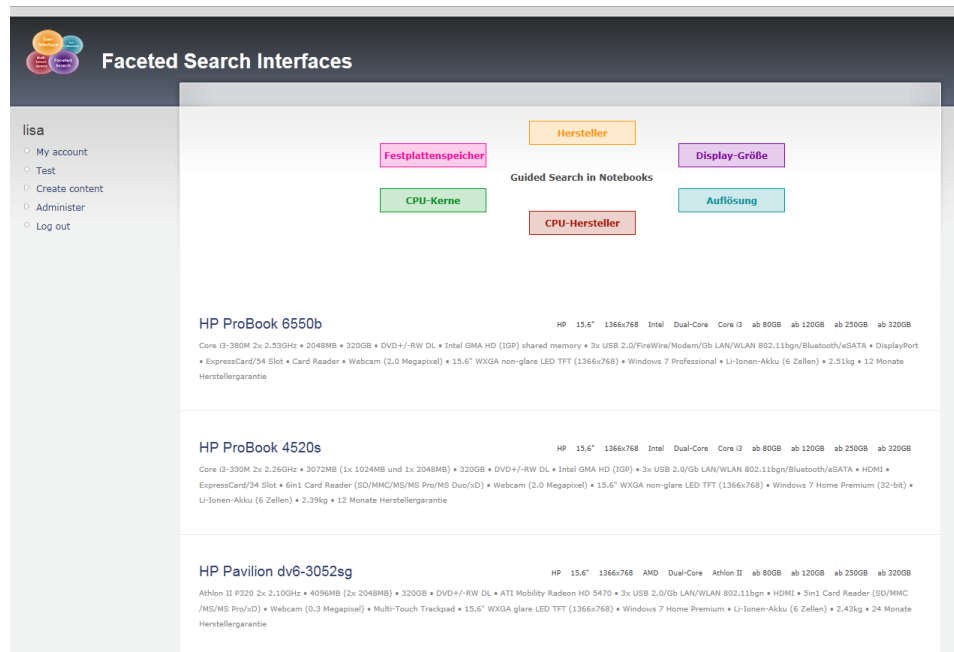


Abbildung 3.10: Startzustand des Ellipseninterfaces.

der Auswahl von neuen Werten einen Blick auf die Suchergebnisse werfen zu können (siehe Abschnitt 2.3.3). Für das Ellipseninterface war es also wichtig, die Facetten und Werte auf diese Weise anzuordnen, dass der genutzte Platz für das Interface so klein wie möglich bleibt.

### Anordnung und Darstellung des „Current Search“-Blocks

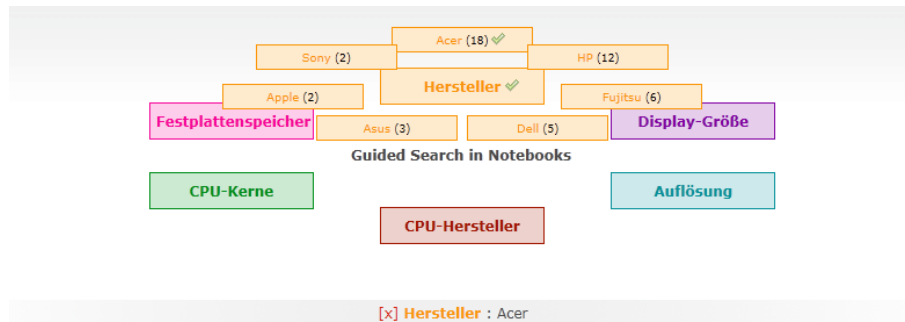
Da das Interface oben in der Mitte der Seite, statt der standardmäßigen Platzierung im rechten Seitenbereich, angeordnet und somit der rechte Seitenrand eliminiert wurde, musste auch für den „Current Search“-Block eine neue Position gefunden werden.

Im Gegensatz zum „Current Search“-Block des Blockinterfaces, wurde der Block im Ellipseninterface absichtlich unter die angeordneten Facetten und Werten positioniert (siehe Abbildung 3.12). Im Ellipseninterface passiert zu keiner Zeit eine Änderung in der vertikalen Ausrichtung, da angezeigter Wert absolut elliptisch um die Facette positioniert werden (siehe Abschnitt 3.3.2).

Da nach unten hin so viel Platz wie möglich gespart werden soll, um eine Vorschau auf die Suchergebnisse bieten zu können, werden die aktiven Werte im „Current Search“-Block des Ellipseninterfaces mit kurzem Abstand nebeneinander zentriert angeordnet (siehe Abbildung 3.11). Sollten so viele Werte ausgewählt sein, dass der Platz in einer Zeile nicht ausreicht, wird wieder von der Mitte ausgehend eine neue Zeile begonnen.

[x] Hersteller : Acer [x] CPU-Hersteller : Intel [x] CPU-Kerne : Dual-Core

**Abbildung 3.11:** Darstellung des „Current Search“-Blocks im Ellipseninterface.



**Abbildung 3.12:** Darstellung der Facetten und Werte im Ellipseninterface mit aktivem Wert „Acer“ (Hersteller).

Wie im Blockinterface, werden auch im Ellipseninterface die Facetten im „Current Search“-Block in der zugehörigen Farbe angezeigt und die Werte können aus dem aktuellen Suchergebnis per Klick auf das [X] entfernt werden.

### Anordnung der Facetten

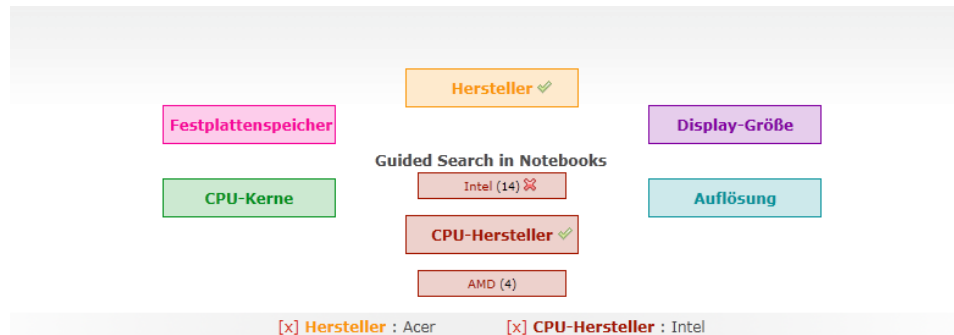
Die Facetten und Werte des Ellipseninterfaces werden, wie der Name andeutet, ellipsenförmig angeordnet (siehe Abbildung 3.12). Der genutzte Platz bei einer solchen Art der Anordnung bleibt bis zu einer bestimmten Menge von Facetten und Werten relativ klein und das Interface trotzdem übersichtlich.

Ursprünglich wurde überlegt, auch die Umrandung der Facetten elliptisch zu gestalten, damit das gesamte Interface einheitlich wirkt. Diese Idee wurde jedoch verworfen, da im Interface, zusätzlich zur ungewohnten Anordnung, gleichzeitig noch eine ungewöhnliche Darstellung der Facetten von den Benutzern verarbeitet werden müsste. Daher werden die Facetten, wie schon in Abschnitt 3.1.1 erklärt, in Blöcken dargestellt.

Auch im Ellipseninterface werden aktive Facetten per Icon von den nicht aktiven Facetten unterschieden.

### 3.3.2 Darstellung der Werte

Wie gerade in Abschnitt 3.3.1 besprochen, werden die Facetten im Ellipseninterface elliptisch angeordnet. Neben den Facetten werden außerdem alle



**Abbildung 3.13:** Darstellung der Facetten und Werte im Ellipseninterface mit aktivem Wert „Acer“ (Hersteller) und „Intel“ (CPU-Hersteller).

zugehörigen Werte rund um die zugehörige Facette in Ellipsenform angeordnet (siehe Abbildungen 3.12 und 3.13).

Zu Beginn der Entwicklung sind einige Darstellungsmöglichkeiten der Werte für dieses Interface überlegt worden, jedoch wirken z. B. alleinstehende Werte mit lediglich einem Hyperlink relativ verloren. Durch die unterschiedlichen Wortlängen und ohne sichtbare Abgrenzung zu anderen Werten ist es auch schwer, eine geometrische und symmetrische Figur zu erkennen. Deshalb werden die Werte, so wie die Facetten, in übersichtlichen Blöcken dargestellt. Um dem Benutzer zusätzlich die Zusammengehörigkeit von Facette und Werten zu vermitteln, werden die Werte in derselben Farbe wie die Facette angezeigt.

Während im Blockinterface der Unterschied von aktiven Werten zu nicht aktiven Werten leicht durch die Fettschrift gelöst werden konnte, stand man bei der Darstellung von aktiven Werten im Ellipseninterface vor demselben Problem wie bei allen Facetten – eine Fettschrift allein war zu wenig auffallend und eine Änderung der Farbe wurde bereits bei den Facetten ausgeschlossen (siehe Abschnitt 3.1.1). Den aktiven Werten im Ellipseninterface wurde daher dasselbe Icon wie den aktiven Facetten verliehen.

Das Icon hat jedoch bei den aktiven Werten im Ellipseninterface noch eine zusätzliche Funktion: Da die aktiven Werte per Klick auch wieder aus der Suchanfrage entfernt werden sollen und der Benutzer dazu auch animiert werden will, brauchen aktive Werte einen Indikator, dass ein Klick auf diesen Wert auch eine gewisse Funktion zur Folge hat. Daher werden die Icons in den Werten während des Überfahrens mit der Maus in ein Icon geändert, das die Entfernung aus dem Suchergebnis impliziert (siehe Abbildung 3.13). Damit soll gewährleistet werden, dass der Benutzer auch weiß, dass der erneute Klick überhaupt eine Funktion hat und die Funktion mit Hilfe des Icons beschrieben ist.

# Kapitel 4

## Implementierung

Um die theoretisch erarbeiteten Interfaces testen zu können, wurden zwei Prototypen entwickelt. Als Grundlage für die Implementierungen wurde im Bereich der CMS recherchiert und, abhängig von der bereits existierenden Unterstützung für Faceted Search in Form von Erweiterungsmodulen, Drupal installiert.

### 4.1 Designentscheidungen

Zur Erweiterung von Drupal gibt es bereits mehrere Module, welche unter anderem Faceted Search anbieten. Neben Apache Solr für Drupal<sup>1</sup> (siehe Abschnitt 2.1.3) gibt es auch ein Modul mit dem Namen *Faceted Search*<sup>2</sup>. Apache Solr wird speziell für Webseiten mit großen Datenmengen empfohlen, da es unabhängig als Webservice betrieben werden kann. Das Faceted Search Modul hingegen basiert rein auf PHP und MySQL und ist daher nur bis zu einer gewissen Datenmenge performant. Davon abgesehen lässt sich das Faceted Search Modul leicht installieren und bietet nach einem einfachem Setup ausreichend Einstellungsmöglichkeiten und Modifikationen für die neu entwickelten Interfaces. Aus diesem Grund wurde das Faceted Search Modul als Ausgangsprodukt gewählt.

Die Entscheidung für die verwendete Programmiersprache ergab sich aus der Wahl der grundlegenden Produkte, da Drupal, sowie auch das Faceted Search Modul auf PHP basieren. Für die nötigen Erweiterungen wurde daher im existierenden Daten- und Klassenmodell in PHP gearbeitet.

Wie in Kapitel 3 bereits angesprochen wurde, wird bei beiden neu entwickelten Interfaces JavaScript und Ajax verwendet. Gerade bei der Implementierung einer asynchronen Datenübertragung bedarf es vieler Zeilen von JavaScript-Code. Da die Nutzung einer Bibliothek diese Implementierungen um ein Vielfaches erleichtert, wurde hierfür *jQuery* verwendet. jQuery ist

---

<sup>1</sup><http://drupal.org/project/apachesolr>

<sup>2</sup>[http://drupal.org/project/faceted\\_search](http://drupal.org/project/faceted_search)

eine schnelle und präzise JavaScript-Bibliothek, die das Handling von Ereignissen sowie das Ansprechen von Elementen und Einrichten von asynchrone Datenübertragungen um ein Vielfaches vereinfacht<sup>3</sup>.

### 4.1.1 Drupal

Das CMS Drupal wurde unter anderem wegen seiner Vielseitigkeit für die neu entwickelten Interfaces als Basis gewählt. Drupal arbeitet modular, d. h. alle verfügbaren Funktionen von Drupal können in Module gegliedert werden. Die Grundfunktionen werden z. B. im Core-Modul abgehandelt, während auch selbst implementierte Module als solche behandelt werden [20]. Bei Aufruf einer bestimmten URL wie z. B. <http://master.lipoha.at/admin/content/taxonomy> wird das Admin-Modul angesprochen und dort die jeweiligen Funktionen über definierte Einsprungspunkte (sogenannte *Hooks*) aufgerufen. Angehängte Parameter können jederzeit in den jeweiligen Modulen ausgelesen werden. Möchte man Drupal um Funktionen erweitern, müssen diese zuerst in Form von Modulen installiert und anschließend je nach Bedarf aktiviert oder deaktiviert werden.

Neben dieser modularen Art der Funktionsverwaltung ist auch die Verwaltung des Inhalts bei Drupal speziell: Jeglicher Inhalt in der Datenbank wird grundlegend als *Node* erkannt und behandelt. Soll diesem Inhaltstyp Semantik verliehen werden, geschieht dies durch die Zuweisung eines bestimmten Inhaltstyps. Vordefinierte Inhaltstypen sind *Page* und *Story*, wobei mit Hilfe des Content Konstruktion Kit (CCK)<sup>4</sup> beliebige Inhaltstypen generiert werden können.

Drupal unterstützt auch die Semantik des Inhalts. Mit Hilfe der eingebauten Taxonomie-Verwaltung ist es möglich, verschiedene „Vocabularies“ als Begriff-Sammlungen anzulegen und diese Begriffe, je nach Bedeutung, neu generiertem Inhalt zuzuweisen.

## 4.2 Faceted Search Modul

Als Basis für die neuen Faceted Search Interfaces wurde das Faceted Search Modul (siehe Abbildung 4.1) ausgewählt und überarbeitet. Nach der Installation und Konfiguration wurde die Datenbank mit Testdaten befüllt und das Modul wurde neben vielen kleinen Änderungen um zwei wichtige Funktionen erweitert. Diese sind zum einen die Möglichkeit der Mehrfachauswahl von Werten und zum anderen die Bereitstellung der Daten für die asynchrone Datenübertragung.

---

<sup>3</sup><http://jquery.com>

<sup>4</sup><http://drupal.org/project/cck>

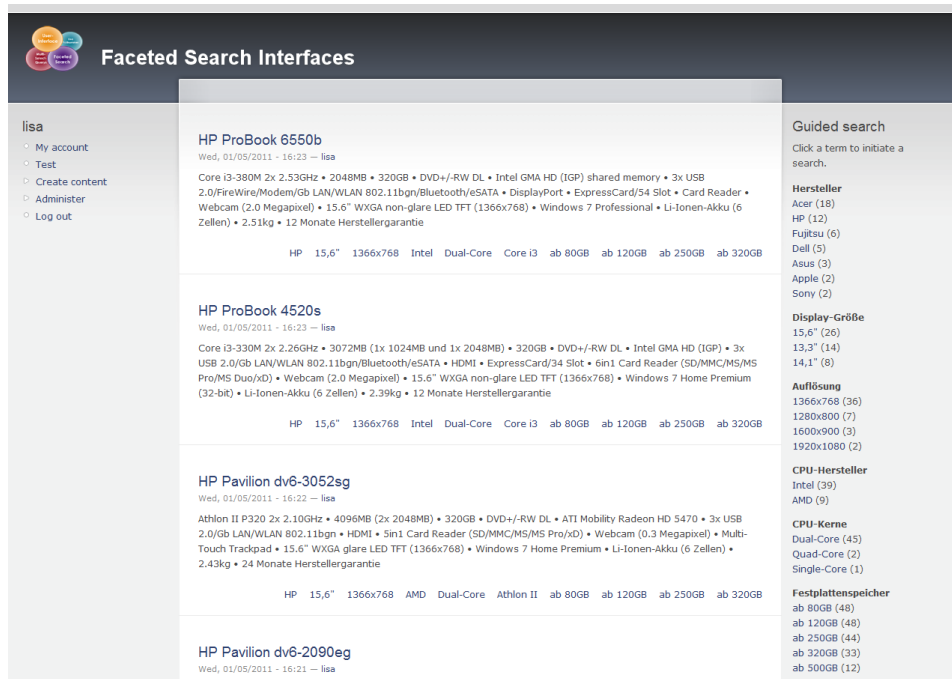


Abbildung 4.1: Standarddarstellung des Faceted Search Moduls von Drupal.

#### 4.2.1 Konfiguration und Testdaten

Das Faceted Search Modul bietet bereits standardmäßig einige Möglichkeiten, Metadaten von benutzergeneriertem Inhalt als Facetten und Werte zu verwenden. Neben dem Autor der Inhaltelemente oder dem Erstellungsdatum, kann auch die interne Taxonomie-Verwaltung von Drupal für die Faceted Search verwendet werden.

Dafür wurde eine repräsentative Menge an „Vocabularies“ angelegt, die im Interface die Facetten darstellen sollen (Hersteller, CPU-Hersteller, usw.). Diese Vocabularies wurden mit den zutreffenden Begriffen befüllt, die dann die Werte der Facetten bilden (bei Hersteller z. B.: Acer, Asus, usw.).

Die für die Evaluierung notwendigen Testdaten wurden von Geizhals aus dem Hardware/Notebook-Bereich übernommen. Für einen realistischen Vergleich in der Usability-Studie ist es von Vorteil, in allen Interfaces mit denselben Daten arbeiten zu können. Dafür wurden ca. 50 Datensätze von Geizhals in die Datenbank von Drupal übernommen und für jeden Datensatz die nötigen Zuweisungen in der Taxonomie-Verwaltung erstellt.

Für eine Trennung dieser Produkteinträge von normalen, inhaltsbezogenen Datensätzen im gesamten Bestand der Drupal Datenbank wurde ein neuer Inhaltstyp (siehe Abschnitt 4.1.1) speziell für diese Einträge erstellt.

Das Anlegen dieses neuen Inhaltstypen hat neben der Kategorisierung auch den Grund, dass für das Faceted Search Modul eine Umgebung angelegt werden muss, in welcher exakt die erlaubten Inhaltstypen für die Faceted Search definiert werden können. Die einzelnen Datensätze werden so vom restlichen Inhalt abgegrenzt.

Zur abschließenden Konfiguration des Faceted Search Moduls wurde in der Seitenkonfiguration diese neue Umgebung angelegt. Neben der Definition der bereits erwähnten erlaubten Inhaltstypen wurden genaue Einstellungen zu allen Komponenten, der Art der Facetten und den verschiedenen Ansichten für die Faceted Search Interfaces getroffen.

### 4.2.2 Grundfunktionen des Moduls

Das Faceted Search Modul unterstützt in der offiziellen Version, neben einer reinen Einfachauswahl von Werten bei entsprechender Konfiguration, auch eine hierarchische Beziehung der Facetten und Werte.

Standardmäßig wird im Modul zur Auswahl eines Wertes der Einsprungspunkt für die Taxonomie-Darstellung des Moduls angesprochen und die bestimmten Werte als Parameter an die URL angehängt. Es wird also bei jeder Auswahl für jeden Wert ein neuer Link erstellt und im Code des Moduls die Anzeige der Seite generiert.

Die Parameter in der URL identifizieren die jeweiligen Werte per Identifikationsnummer und werden bei unterschiedlichen Facetten (also so wie es standardmäßig implementiert wurde) per Beistrich getrennt. Nimmt man also einen typischen Link eines Wertes aus dem Faceted Search Modul, kann die URL wie folgt lauten: `http://master.lipoha.at/v2/faceted_search/results/taxonomy:9,24`. Als aktiv werden nun in der neu geladenen Version der Seite die auf Taxonomie basierenden Werte mit den Identifikationsnummern 9 und 24 betrachtet und die Suchergebnisse vor der Ausgabe mit exakt diesen Werten in der Suchanfrage generiert.

Im Gegensatz zu Drupal selbst ist im Faceted Search Modul objektorientiert programmiert worden. Dies bedeutet, dass zahlreiche Klassen vorhanden sind, welche das Auslesen der Werte, das Zusammenstellen der Suchanfrage und die Anzeige der Suchergebnisse, des „Current Search“-Blocks und des eigentlichen Suchinterfaces übernehmen.

### 4.2.3 Erweiterungen im Modul

Da die neu entwickelten Interfaces zwar keine hierarchischen Facetten und Werte, aber dafür eine Mehrfachauswahl von Werten erlauben sollen, mussten im Modul entsprechende Änderungen und Erweiterungen implementiert werden.

Diese bereits vorhandene Unterstützung für hierarchische Facetten und Werte wurde verändert, um für die neu entwickelten Interfaces die Mehrfach-

auswahl zu verarbeiten. Das Modul war bereits darauf ausgelegt, nicht nur einzelne Werte aus den Parametern auszulesen, sondern konnte auch eine Verbindung der Werte erkennen. Während einzelne Werte per Beistrich getrennt ausgelesen werden, können diese Werte innerhalb der Beistrichgrenzen auch durch die Verbindung von Punkten gruppiert werden. Ursprünglich ist damit die gemeinsame Hierarchie dargestellt worden. Nun werden auf diese Weise mehrere Werte der gleichen Facette an das Modul übergeben, wobei die Werte innerhalb von Beistrichen an eine bestimmte Facette gekoppelt sind. Es wurden also die generierten Hyperlinks der Werte so angepasst, dass die richtige Punkt- und Stichnotation der einzelnen Werte für die geplanten Zwecke verwendet wird.

Am Beispiel der URL [http://master.lipoha.at/v2/faceted\\_search/results/taxonomy:9.13,24.41,26](http://master.lipoha.at/v2/faceted_search/results/taxonomy:9.13,24.41,26) kann also folgende Suchanfrage konstruiert werden: Aus insgesamt drei Facetten wurden Werte ausgewählt, was durch die Beistrichtrennung ersichtlich ist. Die Werte 9 und 13, sowie die Werte 24 und 41 gehören jeweils zur gleichen Facette. Durch diese kleine aber ausschlaggebende Modifikation werden alle benötigten Werte bereits vom Modul ausgelesen. Für die Verwendung aller Werte müssen diese jedoch aus den übergebenen Informationen separat extrahiert werden, da vom Modul bei durch Punkt getrennten Gruppierungen nur der letzte Wert als aktiver Wert angesehen wird. Bei hierarchischen Werten ist der letzte Wert logischerweise an die vorigen Werte gekoppelt. Diese werden daher für die eigentliche Suchquery nicht benötigt – für die geplante Funktionalität sind jedoch alle Werte von gleicher Bedeutung.

Nach der Extrahierung aller aktiven Werte werden diese, in einem separaten Array, durch die einzelnen Funktionen mitgeschleust. An jeder Stelle, an der ursprünglich Werte eliminiert oder gruppiert wurden („Current Search“-Block oder Suchinterface), wurden die Funktionen so geändert, dass, statt den bisherigen Informationen, mit den neuen Daten gearbeitet wird.

Dies ist wichtig, da das Modul im „Current Search“-Block ursprünglich Werte in derselben Hierarchie zu einer einzelnen Facette zuordnet, aber für die Option der Mehrfachauswahl eine eigene Darstellung zur Entfernung einzelner Werte notwendig ist. Außerdem wurden ursprünglich die Werte einer Facette, in der bereits ein Wert aktiv ist, ausgeblendet. Diese Funktion wurde überarbeitet, da sich im Interface nichts verändern soll, damit auch nach der Auswahl eines Wertes trotzdem noch alle anderen Werte der betreffenden Facette dargestellt werden.

Um alle nötigen Funktionen für die Mehrfachauswahl ansprechen zu können, wurde auch die MySQL-Query angepasst. In der ursprünglichen Version des Moduls wird für die Zusammenstellung der Datenbank-Abfrage eine Klasse namens *faceted\_search\_query* verwendet. Im Konstruktor dieser Klasse werden zuerst die Grundelemente der Abfrage zusammengestellt. Anschließend wird das erstellte Objekt der Klasse je nach Art der Facetten (in diesem Fall Taxonomy-Facetten) an die *build\_results\_query*-Funktionen der



**Programm 4.1:** Die ursprüngliche Modifikation der Datenbank-Abfrage der Taxonomy-Facets im Faceted Search Modul.

```
1 function build_results_query(&$query) {
2     $query->add_table('taxonomy_facets_term_node', 'nid', 'n', 'nid', "
   term_node_{\${this->_tid}");
3     $query->add_where("term_node_{\${this->_tid}}.tid = %d", \${this->_tid});
4 }
```

verantwortlichen Klassen übergeben. In diesen Funktionen werden weiters die notwendigen Änderungen an der Abfrage durchgeführt. In Programm 4.1 ist ersichtlich, dass die nötige Tabelle, in welcher die Werte mit den Produkten verknüpft sind (*taxonomy\_facets\_term\_node*), zur Abfrage hinzugefügt wird und anschließend der betreffende Wert einen Teil der *WHERE*-Klausel in der Abfrage darstellen soll. Diese Funktion wird für jede einzelne gewählte Facette aufgerufen, bis schlussendlich alle gewählten Werte der Facetten in der Suchabfrage enthalten sind. Bei der Fertigstellung der Datenbank-Abfrage werden alle angegebenen Tabellen per *JOIN* in die Abfrage mit aufgenommen, während die registrierten Werte per *AND* verknüpft werden.

Bei einer Mehrfachauswahl ändert sich diese Logik der Verknüpfung der Werte grundlegend. Anstatt pro Facette genau einen Wert in die Abfrage mit aufzunehmen und diese Werte per *AND* zu verknüpfen, müssen auch alle anderen gewählten Werte der Facette für ein korrektes Suchergebnis berücksichtigt werden. Es ist also notwendig, die Werte derselben Facette per *OR* zu verknüpfen, während die einzelnen Facetten-Gruppen in weiterer Folge per *AND* in die Abfrage übernommen werden.

Da die *build\_results\_query*-Funktion einmal pro Facette aufgerufen wird, müssen in dieser Funktion alle aktiven Werte der Facette auf einmal abgearbeitet werden. Dafür wurden die bereits angesprochenen extrahierten Werte der Funktion übergeben, alle aktiven Werte der Facette ausgelesen, per *OR* verknüpft und an die Klasse der Datenbank-Abfrage übergeben (siehe Programm 4.2).

Auf diese Weise werden alle Werte je nach Facetten-Zugehörigkeit richtig in die Datenbank-Abfrage eingearbeitet und man erhält ein korrektes Suchergebnis, unabhängig von der Art der Auswahl der Werte.

### Asynchrone Datenübertragungen

Wie bereits in Kapitel 3 angesprochen wurde, sollen beide Interfaces einmalig geladen werden und notwendige Schlüsselemente, wie die Suchergebnisse oder die Query-Previews, mit Hilfe asynchroner Datenübertragungen aktualisiert werden.

Einerseits ist es möglich, die Datenbeschaffung über ein separates Script

**Programm 4.2:** Die überarbeitete Modifikation der Datenbank-Abfrage der Taxonomy-Facets im Faceted Search Modul.

```

1  function build_results_query(&$query, $filter) {
2    if (count($filter->_path) > 1) {
3      $query->add_table('term_node', 'nid', 'n', 'nid', "term_node_{
4        $this->_tid}");
5      $ors = array();
6      foreach ($filter->_path as $value) {
7        $ors[] = "term_node_{\$this->_tid}.tid = ".$value->_tid;
8      }
9      $queryString = "(" . implode(' OR ', $ors) . ")";
10     $query->add_where($queryString);
11   } else {
12     $query->add_table('term_node', 'nid', 'n', 'nid', "term_node_{
13       $this->_tid}");
14     $query->add_where("term_node_{\$this->_tid}.tid = %d", $this->_tid)
15     ;
16   }
17 }

```

zu lösen, welches die Ausgabe auf benötigte Elemente limitiert. In diesem Fall muss jedoch die gesamte Abfragelogik des Moduls nachimplementiert werden. Die andere Möglichkeit ist, das Modul direkt anzusprechen um dadurch die bereits bestehenden Funktionen des Moduls zu nutzen. Dafür werden die ursprünglich vom Modul generierten Hyperlinks für die Abfrage-Adressen manuell nachgebaut (siehe Abschnitt 4.2.2) und dem Modul dadurch eine getätigte Auswahl von Werten vorgetäuscht.

Da der erste Ansatz eine hohe Redundanz zur Folge hätte, wurde für beide Interfaces jene Methode gewählt, bei der das Modul in den Abfragen direkt angesprochen wird. Um nun die Suchergebnisse korrekt nachladen zu können, wurden diese im Design-Template in einen Container geschachtelt um anschließend diesen Container separat ansprechen und aktualisieren zu können.

Wie bereits erwähnt, wird durch das direkte Ansprechen des Moduls über die URL nun in der asynchronen Datenübertragung der Inhalt der gesamten Seite geladen. Hier bietet die *load()*-Funktion von jQuery neben der Datenanfrage und der Abwicklung der Antwort auch die Möglichkeit, die Antwort auf Fragmente der gesamten Seite zu limitieren, sofern diese in Containern geschachtelt sind. Wie in Zeile 2 und 14 im Programm 4.3 zu sehen ist, wird die Anfrage an eine normale Ausgabeseite gesendet, wobei der Inhalt des Containers *contentarea* auf der aktuellen Seite mit dem Inhalt des selben Containers auf der angefragten Seite ersetzt wird.

Vor jeder Anfrage müssen die Parameter aus den aktiven Werten zusammengestellt werden. Dafür werden alle aktiven Werte in einem globalen

**Programm 4.3:** Die Zusammenstellung der asynchronen Suchanfrage inklusive anschließender Modifikationen des Interfaces.

```
1 function startSearch() {
2   var link = 'http://master.lipoha.at/v2/faceted_search/results/';
3   var linkParams = new Array();
4   for(var i = 0; i < activeFacets.length; i++) {
5     if(activeFacets[i] != undefined && activeFacets[i].length > 0) {
6       linkParams.push(activeFacets[i]);
7     }
8   }
9
10  if (linkParams.length > 0) {
11    link += 'taxonomy:'+linkParams.join(',');
12  }
13
14  $('#contentarea').load(link+' #contentarea', function() {
15    buildCurrentSearchBlock();
16    formatSearchResults();
17  });
18
19  $('#temp_count_values').load(link+' #new_value_count', function(
20    content) {
21    $('#.faceted-search-value-count').each(function() {
22      var idParts = $(this).attr('id').split('_');
23      var tid = idParts[3];
24      var newCount = $('#nvc_'+tid).html();
25      if (newCount == null) {
26        newCount = 0;
27      }
28      $(this).html('&nbsp;('+newCount+')');
29    });
30  }
```

Array verwaltet, welches bei einer neuen Auswahl von Werten oder einer Entfernung je nach Identifikationsnummer des Wertes aktualisiert wird. Bei jeder Anfrage werden die aktiven Werte aus dem Array ausgelesen, verknüpft und an die URL angehängt, wie in Zeile 2-12 in Programm 4.3 ersichtlich ist.

Neben der Aktualisierung der Suchergebnisse werden auch die Zahlen in den Query-Previews nach jeder Aktion des Users neu berechnet und aktualisiert, um die Zusammenhänge zum aktuellen Suchergebnis widerzuspiegeln. Im Gegensatz zum Nachladen der Suchergebnisse stellte sich dieser Teil der Aktualisierung komplizierter dar. Auch bei den Query-Previews muss das Modul direkt angesprochen und die Zahlen aus einem versteckten Container ausgelesen werden. Dafür ist es erforderlich, die Zahlen im Modul an einer beliebigen Stelle auszugeben. Das Problem war in diesem Fall, dass im Modul



**Abbildung 4.2:** Die Darstellung des überarbeiteten „Current Search“-Block des Faceted Search Moduls.

kaum eine Funktion Zugriff auf alle (nicht nur die aktiven) Werte und deren Anzahl im Suchergebnis und gleichzeitig Zugriff auf das Ausgabe-Objekt hat.

Dafür wurde das Modul so modifiziert, dass an einer Stelle, an der alle Facetten und Werte ausgelesen werden, sofort die Anzahl der einzelnen Produkte zu den Werten berechnet und mitgeschleust wird. Wird eine Funktion erreicht, welche für einen Teil der Ausgabe verantwortlich ist, werden die Werte in einzelne Container verpackt um sie separat auslesen zu können und auf diese Weise an die Ausgabe angehängt.

Ab Zeile 19 im Programm 4.3 ist zu sehen, dass, wie bei den Suchergebnissen, nach der Anfrage wieder zuerst der nötige Teil aus der Antwort extrahiert wird und anschließend die einzelnen Query-Previews der betreffenden Werte mit den neuen Zahlen aktualisiert werden.

### „Current Search“-Block

Durch den Einsatz der asynchronen Datenübertragung musste auch der „Current Search“-Block überarbeitet werden (siehe Abbildung 4.2). In der ursprünglichen Funktionsweise des Moduls wird dieser Block direkt ausgegeben. Durch die Änderungen in der Datenübertragung ist dies nicht möglich und die Zusammenstellung wird daher manuell mit Hilfe gespeicherter Variablen aus dem JavaScript-Code realisiert.

Wie in Programm 4.4 zu sehen ist, werden dafür die konstant aktualisierten Datenarrays für aktive Facetten durchsucht und dabei geprüft, ob aktive Werte vorliegen. Ist dies der Fall, werden für diese Werte einzelne Container generiert, in welchen jeweils der Name der Facette, der Name des Wertes und der Link zur Entfernung enthalten ist. Diese Container werden anschließend nacheinander an den „Current Search“-Block angehängt. Neben der Darstellung der aktiven Werte wird auf diese Weise auch eine Reihung der Werte nach den Facetten erreicht, d. h. dass Werte der gleichen Facette direkt nebeneinander im „Current Search“-Block angezeigt werden.

**Programm 4.4:** Die Funktion zur Zusammenstellung des „Current Search“-Blocks aus den aktiven Werten und Facetten.

```
1 function updateCurrentSearchFloatContainer() {
2   $('#current_search_float_container').empty();
3   $('.facetcheck').hide();
4   var valueCount = 0;
5
6   for (var i = 0; i < activeFacets.length; i++) {
7     var facetValues = new Array();
8     var facetName = '';
9
10    if (activeFacets[i] != undefined && activeFacets[i].length > 0) {
11      facetName = $('#facet_' + i).html();
12      facetValues = activeFacets[i].split('.');
13      for (var j = 0; j < facetValues.length; j++) {
14        var valueName = $('#'+i+'_'+facetValues[j]).html();
15        $('#current_search_float_container').append(
16          buildNewCurrentSearchBlock(i, facetValues[j], facetName, valueName))
17        ;
18        valueCount++;
19      }
20    }
21  }
```

## 4.3 Erstellung der Interfaces

Die in Abschnitt 4.2 angesprochenen Änderungen am Faceted Search Modul betreffen das Blockinterface sowie das Ellipseninterface in gleichem Maße. Die serverseitige Funktionalität ist bei beiden Interfaces ident, während sich die clientseitigen Funktionen unterscheiden. Das Aussehen der einzelnen Interfaces und das Verhalten der Elemente wurde mit JavaScript, in Verbindung mit jQuery und CSS realisiert.

### 4.3.1 Blockinterface

Der erste Schritt bei der Implementierung des Blockinterfaces war die Unterscheidung der einzelnen Facetten und deren Werte durch aussagekräftige Farben. Dafür wurden Farben gewählt, die sich durch einen hohen Kontrast voneinander unterscheiden, aber im Gesamtbild passend in die restliche Seite integriert werden können (siehe Abbildung 3.7 in Abschnitt 3.2).

In der Standardausgabe des Faceted Search Moduls werden die Facetten automatisch in Container verpackt, während die Werte der Facetten als Liste im selben Container dargestellt werden. Um die Facetten und deren Werte konstant miteinander assoziieren zu können und auch bei Änderungen die

richtigen Identifikationsnummern (ID) der Werte und Facetten auslesen zu können, wurde direkt nach dem Laden der Seite die Container-Struktur des Interfaces verändert. Die Facetten wurden in Container verpackt und die ID des Containers direkt nach der ID der Facette benannt. Außerdem wurde um die Werte der Facetten ein Container gelegt, in dem wiederum die ID der betreffenden Facette in der eigenen Identifikationsnummer enthalten ist. So ist es möglich, alle Komponenten einer Facette (Überschrift und Werte) separat ansprechen zu können und im CSS spezifische Eigenschaften zu vergeben.

Nachdem die Struktur des Interfaces neu angelegt wurde, konnte über die Eigenschaften im CSS die Gestaltung vorgenommen werden. Dafür wurden die Farben den einzelnen Facetten und ihren Werte zugewiesen, die Überschriften der Facetten gestaltet und die Hyperlinks der Werte angepasst. Weiters wurden die ausgesuchten Farben auch im „Current Search“-Block angewandt, in welchem die einzelnen Werte auch über passende Identifikationsnummern den zugehörigen Facetten für die richtige Farbwahl zugewiesen werden konnten.

Wie bereits in Abschnitt 3.2 angesprochen, werden im Blockinterface zu Beginn nur die Facetten angezeigt. Durch Klick auf die Facetten können dann die Werte der Facette aufgeklappt und durch erneuten Klick wieder versteckt werden. Da die Werte in der neuen Struktur in einen eigenen Container verpackt wurden, kann dieses Anzeigen und Verstecken einfach durch Modifikation per JavaScript realisiert werden. Alle Facetten-Überschriften wurden mit einer Funktion belegt, welche bei Klick den Container sichtbar macht und bei erneutem Klick den Container wieder versteckt.

Da die einzelnen Container der Facetten und Werte nach- bzw. untereinander angeordnet wurden, erübrigt sich die aktualisierte Positionierung bei geöffneten Werten. Sind Werte sichtbar, verschieben sich die restlichen Container automatisch durch die Block-Eigenschaft der Container um den neu benötigten Platz nach unten und nach dem Verstecken wieder nach oben.

### 4.3.2 Ellipseninterface

Da das Ellipseninterface im Anschluss an das Blockinterface implementiert wurde, konnten für das Ellipseninterface einige Teile des Codes des Blockinterfaces übernommen werden. Neben der Farbgebung wurde vor allem die Einteilung und neue Strukturierung der Facetten und Werte direkt übernommen. Auch die Funktionalität des Anzeigens und Versteckens der Werte bei Klick auf die Facetten konnte mit kleinen Änderungen verwendet werden.

Neben diesen gemeinsamen Komponenten unterscheidet sich das Ellipseninterface vom Blockinterface vor allem durch die Anordnung der Facetten, Werte und des „Current Search“-Blocks. Wie in Abschnitt 3.3 besprochen wurde, war das Ziel des Ellipseninterfaces, so wenig Platz wie möglich in Anspruch zu nehmen und daher die einzelnen Facetten und Werte elliptisch anzuordnen. Dafür wurden die einzelnen Positionen der Facetten und Werte

**Programm 4.5:** Die JavaScript-Methode zur Berechnung der Position der Elemente in der Ellipse.

```

1 function arrangeFacets(facets, divX, divY, a, b, centerX, centerY,
   idPrefix) {
2   var x;
3   var y;
4   var t;
5
6   for (var i = 0; i < facets.length; i++) {
7     t = (((2*Math.PI)/facets.length)*i)-(Math.PI/2);
8     xM = centerX + (a * Math.cos(t));
9     yM = centerY + (b * Math.sin(t));
10
11    x = xM - (divX/2);
12    y = yM - (divY/2);
13
14    $('#'+idPrefix+facets[i]['id']).css('left', x).css('top', y);
15  }
16 }

```

mit der Ellipsengleichung

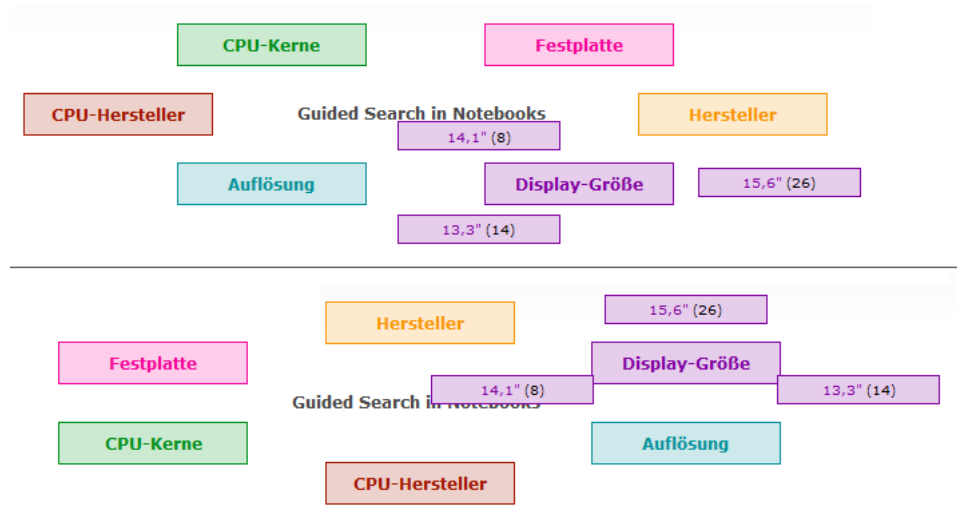
$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_0 + a \cos t \\ y_0 + b \sin t \end{pmatrix} \quad \text{mit } 0 \leq t \leq 2\pi \quad (4.1)$$

berechnet. Während  $a$  und  $b$  die Form der Ellipse bestimmen, ist  $t$  der Parameter zur Berechnung der Position der Elemente im Koordinatensystem.

Wie in Programm 4.5 erkenntlich ist, wird zur Berechnung und Positionierung der Elemente eine JavaScript-Funktion genutzt. An diese Funktion werden die zu positionierenden Facetten (und anschließend auch Werte) übergeben. Zusätzlich werden Breite und Länge der einzelnen Container ( $divX$  und  $divY$ ) und die gewünschte Größe und Form der Ellipse ( $a$  und  $b$ ) angegeben. Wichtig ist auch noch der Mittelpunkt der Ellipse ( $centerX$  und  $centerY$ ), da vom Mittelpunkt ausgehend die absoluten Positionen auf der X- und Y-Achse berechnet und anschließend direkt den betreffenden Elementen zugewiesen werden.

Dadurch dass  $\cos(0) = 1$  ist und  $\sin(0) = 0$  ergibt, ergibt sich mit  $t = 0$  die Positionierung des ersten Elements auf der positiven X-Achse. Wird dieser Punkt nun als Ausgangspunkt definiert, werden die restlichen Elemente von diesem Punkt abhängig gleichmäßig auf der Ellipse verteilt. Bei näherer Betrachtung dieser Verteilung der Elemente in Abbildung 4.3 oben, ergibt sich eine scheinbar ungleichmäßige Verteilung der Elemente.

Bei einer Rotation um einen Viertelkreis gegen den Uhrzeigersinn (siehe Abbildung 4.3 unten), wird das erste Element auf der positiven Y-Achse positioniert und die restlichen Elemente wiederum gleichmäßig auf der Ellipse



**Abbildung 4.3:** Die Aufteilung der Elemente in der Ellipse mit  $t = 0$  (oben) und einer Rotation um einen Viertelkreis gegen den Uhrzeigersinn (unten).

verteilt.

Bei einem Vergleich der Anordnung der Elemente in Abbildung 4.3 ist erkenntlich, dass Elemente in der unteren Variante, wo nach der Berechnung der Position eine Rotation vorgenommen wird, vom ersten Element ausgehend primär an der X-Achse angeordnet werden. Im Gegensatz dazu werden die Elemente in der Darstellung oben primär an der Y-Achse angeordnet. Während bei einer geraden Anzahl von Elementen kaum ein Unterschied erkennbar ist, ergibt sich bei einer ungeraden Anzahl von Elementen eine signifikant unterschiedliche Darstellung. Deutlich wird dies in der Anordnung der violetten Werte der Facette „Display-Größe“ in Abbildung 4.3.

Da auch auf gesamten Webseiten der bevorzugte Bereich oben mittig ist [4], wird angenommen, dass auch in Interfaces der obere mittlere Bereich vom Benutzer zu Beginn ins Auge gefasst wird. Für die Anordnung der Elemente wurde daher jene Variante gewählt, in der das erste Element oben mittig positioniert wird. Von diesem Element ausgehend werden in weiterer Folge die restlichen Elemente gleichmäßig verteilt.



# Kapitel 5

## Evaluierung

Nach Fertigstellung der Implementierung der zwei neuen Interface-Entwürfe wurde eine Usability-Studie durchgeführt. Nur durch die Studie mit breitgefächerten Teilnehmern konnte festgestellt werden, inwieweit die Komponenten und Funktionen der neu entwickelten Interfaces erfolgreich sind.

### 5.1 Testdesign

Ziel der Usability-Studie war es, Vor- und Nachteile der einzelnen Interfaces zu erkennen, um diese anschließend miteinander zu vergleichen. Um diesen Vergleich zu ermöglichen, wurden neben den zwei neuen Interface-Entwürfen auch das Standard-Design des Drupal Faceted Search Moduls und das Faceted Search Interface von Geizhals in den Usability-Test mit aufgenommen.

Die Usability-Studie wurde mit insgesamt sechs Teilnehmern geblockt durchgeführt. Nach dem ersten Durchgang mit vier Testpersonen waren einige Probleme gefunden worden, jedoch wurden im zweiten Block noch einmal zwei Testpersonen hinzugezogen. Wird hierfür der Prozentsatz der gefundenen Usability-Probleme laut [18] mit einer Anzahl von 6 Teilnehmern berechnet, erhält man ein Ergebnis von 89%, wobei 100% alle im Interface vorhandenen Usability-Probleme darstellen.

Als Teilnehmer wurden nicht speziell Personen aus gewissen Berufsfeldern oder Altersgruppen angeworben, sondern rein auf die Voraussetzung geachtet, dass alle teilnehmenden Personen zumindest über Grundkenntnisse im Umgang mit dem Internet verfügen. Dies ist laut [15] zum Finden von Usability-Problemen ausreichend. Die Teilnehmer stammen aus unterschiedlichen Berufsgruppen (Student, Webprogrammierer, zwei Software-Engineers, Sekretärin und Hortpädagogin), befinden sich jedoch in derselben Altersgruppe (24-30 Jahre). Jeder der Teilnehmer verfügt über Grundkenntnisse in der Navigation im Web und verbringt täglich mindestens eine Stunde am Computer. Drei der sechs Teilnehmer haben bisher noch keine Erfahrung im Umgang mit dem Such-Interface von Geizhals gesammelt, während die

anderen drei Teilnehmer regelmäßig mit diesem Interface arbeiten.

Für den Usability-Test wurden fünf Szenarien (siehe Abschnitt 5.1.1) zusammengestellt, welche in weiterer Folge für jedes Interface separat durchgeführt wurden. Um die Auswirkung des Lerneffekts der Szenarien zwischen den verschiedenen Interfaces so gering wie möglich zu halten, wurde die Reihenfolge der Interfaces bei jedem Teilnehmer anders festgelegt.

### 5.1.1 Szenarien

Um die Interfaces so breit wie möglich zu testen, wurden folgende fünf Szenarien entwickelt:

1. *Starte eine Suche nach Notebooks mit folgenden Eigenschaften: Hersteller (Acer), Auflösung (1366x768) und CPU-Hersteller (Intel).*

*Wie viele Notebooks entsprechen diesen Kriterien?*

Mit diesem ersten Szenario soll festgehalten werden, ob die Teilnehmer wissen, wie Faceted Search funktioniert bzw. den Einsprungspunkt in das Such-Interface finden. Weiters soll festgestellt werden, ob die Anzahl der Suchergebnisse schnell auffindbar ist.

2. *Finde alle Notebooks der Hersteller Acer, HP und Dell.*

*Limitiere das Suchergebnis auf Notebooks des Herstellers HP.*

Hier wird zum ersten Mal die Möglichkeit der Mehrfachauswahl getestet. Der zweite Teil des Szenario soll zeigen, ob die Teilnehmer wissen, welche Wege es zur Entfernung von aktiven Werten gibt und welcher Weg von den Teilnehmern bevorzugt wird.

3. *Wie viele Notebooks des CPU-Herstellers Intel gibt es (ohne eine Suche zu starten)?*

Dieses Szenario zeigt, ob die Teilnehmer die möglichen Query-Previews entdeckt haben und auch wissen, was diese bedeuten.

4. *Starte eine Suche mit sechs beliebigen Eigenschaften. Wie ist die Anzahl der Suchergebnisse?*

*Limitiere auf drei Eigenschaften – wie ist nun die Anzahl der Suchergebnisse?*

Auf diesem Weg soll noch einmal beobachtet werden, wie genau Werte ausgewählt und wie diese wieder entfernt werden. Die Frage nach den Suchergebnissen zielt darauf ab, zu sehen, wie die Teilnehmer auf die veränderten Suchergebnisse reagieren.

5. *Scrolle zum Anfang und sage, wieviele Eigenschaften ausgewählt wurden und zu welcher Kategorie diese Eigenschaften gehören.*

Für dieses Szenario wurden die Teilnehmer gebeten, sich kurz am Stuhl zur Wand zu drehen, damit eine Suche nach unbekanntem Werten gestartet werden kann. Start des Szenarios war am unteren Seitenende. Damit sollte überprüft werden, ob die Teilnehmer den Sinn der

„Current-Search“-Blöcke im Laufe der ersten vier Szenarien verstanden hatten.

Da die Szenarien mit jedem Interface durchgearbeitet wurden (also insgesamt vier Mal), wurde darauf geachtet, die Aufgabenstellungen so kompakt wie möglich zu halten.

## 5.2 Testablauf

Die Durchführung des Usability-Tests fand in einer ruhigen und störungsfreien Umgebung statt. Um stressfrei arbeiten zu können und auch Zeit für abschließende Fragen zu haben, wurden die Teilnehmer im 45min Takt zum Durchführungsort eingeladen.

Die Teilnehmer wurden zu allererst über die Grundlagen von Usability-Tests aufgeklärt und gebeten, bei der Durchführung der Szenarien laut mitzudenken.

Um anschließend exakt nachvollziehen zu können, wie die Teilnehmer gearbeitet haben bzw. was bei der Durchführung gesprochen worden ist, wurden die Tests mit einer Bildschirnkamera aufgezeichnet. Die Teilnehmer wurden gebeten ein Headset zu tragen, um die Tonaufnahme durch etwaige Hintergrundgeräusche nicht zu verfälschen.

Für die verschiedenen Teilnehmer wurden unterschiedliche Reihenfolgen der Interfaces festgelegt, um den Grad des Lerneffekts zwischen den Interfaces so niedrig wie möglich zu halten.

Zu Beginn wurde auf dem Bildschirm das erste Interface angezeigt und der Teilnehmer gebeten, mit den Szenarien zu beginnen. Die Szenarien wurden Zeile für Zeile abgearbeitet und die Teilnehmer wurden bei großen Problemen oder Unklarheit unterstützt, um die Motivation für die Durchführung der Tests so hoch wie möglich zu halten.

Nach Durcharbeitung der Szenarien in jedem Interface wurden den Teilnehmern abschließende Fragen gestellt. Diese Fragen dienten zur detaillierten Erklärung etwaiger Kommentare während der Durchführung. Die Teilnehmer wurden ebenso direkt gefragt, was erwartet wurde oder welches Interface bevorzugt wurde.

## 5.3 Testergebnisse

Während der Durchführung der Tests wurde unter anderem auch die benötigte Zeit gemessen. Erkennbar war, dass die Dauer der Durchführung mit dem Erfahrungsstand des Teilnehmers mit dieser Art von Interface direkt in Verbindung stand. Teilnehmer mit mehr Erfahrung absolvierten alle Szenarien in allen Interfaces in deutlich kürzerer Zeit (10-13min) als Teilnehmer mit weniger Erfahrung (18-21min).

**Tabelle 5.1:** Diese Tabelle zeigt die durchschnittliche Zeit zur Absolvierung der verschiedenen Szenarien pro Interface und die Rückmeldung der Teilnehmer betreffend die Zufriedenheit mit den verschiedenen Interfaces.

<i>Interface</i>	<i>Durchschnittliche Zeit</i>	<i>Zufriedenheit</i>
Geizhals(GH)	04:17min	+
Drupal Modul(DM)	03:09min	++
Blockinterface(BI)	03:44min	+++
Ellipseninterface(EI)	03:21min	++

**Tabelle 5.2:** Zeitmessung der Teilnehmer(T) 1-3 in den Test-Durchgängen inklusive der Reihenfolge der getesteten Interfaces.

<i>Interface</i>	<i>T1</i>	<i>T2</i>	<i>T3</i>
Geizhals(GH)	03:57min	03:26min	07:40min
Drupal Modul(DM)	02:38min	02:51min	03:07min
Blockinterface(BI)	03:05min	03:57min	06:48min
Ellipseninterface(EI)	03:20min	05:36min	02:42min
Reihenfolge	GH-DM-BI-EI	EI-BI-DM-GH	BI-DM-GH-EI

In Tabelle 5.1 kann erkannt werden, dass für das Interface des Drupal Faceted Search Moduls am wenigsten Zeit benötigt wurde, gefolgt vom Ellipseninterface und dem Blockinterface. Das Interface von Geizhals hat deutlich die längste Zeit in Anspruch genommen.

Der schnellste Durchlauf aller Szenarien wurde von Teilnehmer 6 im Blockinterface in 01:32min durchgeführt (siehe Tabelle 5.3) während Teilnehmer 3 die längste Zeit (07:40min) mit dem Interface von Geizhals verbracht hat (siehe Tabelle 5.2).

Laut [14] ist es nötig, zur Determinierung von guten oder schlechten Interfaces, abhängig von der benötigten Zeit, zu wissen, ob eine schnellere Zeit entweder durch fahrlässiges Verhalten oder größeren Erfolg im Umgang mit dem Interface erreicht wurde. Dies kann durch die Zusammenstellung und Art der Szenarien leider nicht exakt festgestellt werden, daher ist diese Präsentation der Zeitmessung rein informativ.

### 5.3.1 Interface 1: Geizhals

Um die neu entwickelten Interfaces mit bekannten und viel genutzten Faceted Search Interfaces realistisch vergleichen zu können, wurde zur Usability-Studie auch das Such-Interface von Geizhals hinzugezogen (siehe Abbildung

**Tabelle 5.3:** Zeitmessung der Teilnehmer(T) 4-6 in den Test-Durchgängen inklusive der Reihenfolge der getesteten Interfaces.

Interface	T4	T5	T6
Geizhals(GH)	04:54min	03:24min	02:23min
Drupal Modul(DM)	04:40min	03:08min	02:30min
Blockinterface(BI)	03:32min	03:33min	01:32min
Ellipseninterface(EI)	03:24min	02:18min	02:47min
Reihenfolge	DM-EI-GH-BI	BI-EI-DM-GH	DM-EI-GH-BI



**Abbildung 5.1:** Startansicht des Such-Interfaces von Geizhals im Usability-Test.

5.1).

Generell konnte erkannt werden, dass jeder Teilnehmer beim Anblick der Menge der Werte und Facetten Unmut äußerte. Von Personen mit Erfahrung mit Geizhals kam vermehrt der Kommentar „unübersichtlich“, Personen ohne Erfahrung mit Geizhals, aber Begriffskenntnissen, wirkten überrascht, während Personen ohne Erfahrung mit Geizhals und ohne Begriffskenntnis von der Menge der unbekanntenen Begriffe offensichtlich kurzzeitig überfordert waren.

Nachdem sich die Teilnehmer kurz mit dem Interface vertraut gemacht

hatten, stellte die Suche im ersten Szenario relativ wenig Probleme dar. Durch die große Menge von Werten musste teilweise lang nach den gewünschten Werten gesucht werden. Unklarheit entwickelte sich jedoch bei der Frage nach der Anzahl der Suchergebnisse. Diese wird in einem Tabellenkopf über den Suchergebnissen angezeigt und ist eindeutig schwer erkennbar und unauffällig. Zwei der sechs Teilnehmer nutzten die Anzahl der Suchergebnisse in der Query-Preview zur Antwort auf die Frage nach der Anzahl, obwohl sie sich nicht ganz sicher waren, dass die Zahl stimmte. Einer der sechs Teilnehmer konnte die Frage überhaupt nicht beantworten.

Fünf von sechs Teilnehmern äußerten bereits nach den ersten Klicks im ersten Szenario, dass die automatische Verlinkung nach jedem Seitenaufwurf zum Anker im Suchinterface „irritierend“ und „nervig“ ist.

Während der Durchführung des zweiten Szenarios kamen unter anderem die meisten Probleme zum Vorschein. Zwei der sechs Teilnehmer haben nicht realisiert, dass vom vorherigen Suchvorgang noch Werte ausgewählt waren und haben diese daher auch nicht entfernt. Nach der Suche nach den drei Herstellern gestaltet sich für diese Teilnehmer das Entfernen der bereits ausgewählten Werte schwierig. Ein Teilnehmer versuchte mehrmals vergeblich, über den Button „Zurücksetzen“, alle Filter zu entfernen. Erst nach einiger Zeit konnte auch dieser Teilnehmer die Werte erfolgreich durch erneuten Klick auf den Link entfernen. Zwei von sechs Teilnehmern versuchten auch über das kleine Such- und Sortierformular am Anfang der Seite die Werte zurückzusetzen, was jedoch nicht gelang.

Das dritte Szenario stellte für keinen Teilnehmer ein Problem dar. Jeder der Teilnehmer scheint mit dieser Art von Query-Preview vertraut zu sein.

Im vierten Szenario kam ein vermutlich größeres Problem des Interfaces zum Vorschein. Die Darstellung der aktiven Werte ist je nach Auswahlmöglichkeit verschieden. Ist eine Mehrfachauswahl möglich, bleiben die restlichen Werte in der Liste und können ganz normal per Klick auf den Link aktiviert werden, bzw. können aktive Werte per Klick deaktiviert werden. Handelt es sich jedoch um eine Einfachauswahl, werden die Links von allen Werten entfernt und der aktuelle Wert ist fett hervorgehoben. Der Wert kann in weiterer Folge nicht über einen Link entfernen werden. Dafür muss in diesem Fall ein „alle“-Button am Ende der Zeile verwendet werden.

Für alle Personen die nicht im Umgang mit Geizhals geübt waren, stellte diese nicht konsistente Art der Darstellung ein Problem dar. Bei der Auswahl von sechs beliebigen Werten in diesem Szenario kam es also vor, dass zuviele Werte ausgewählt wurden, da einige bereits aktive Werte übersehen und daher von den Teilnehmern nicht gezählt wurden. Dies äußerte sich auch bei der Einschränkung auf drei Werte, da, wie angesprochen, einige aktive Werte nicht als solches vom Teilnehmer identifiziert werden konnten.

Im letzten Szenario traten die bereits angesprochenen Probleme erneut auf. Die Teilnehmer sollten auflisten, welche Facetten mit welchen Werten aktiv sind. Durch die inkonsistente Darstellung konnten drei von sechs Teil-

nehmern die korrekte Antwort nicht nennen.

Zusammenfassend können bei diesem Interface einige Probleme identifiziert werden. Allen voran steht die riesige Menge an Werten und Facetten, die von Anfang an beinahe den gesamten Bildschirm einnimmt. Alle Teilnehmer der Studie waren sich einig, das dieses Interface sehr unübersichtlich ist und zu viele Informationen auf einmal präsentiert werden. Besonders Teilnehmer ohne Begriffskenntnisse merkten an, dass es nicht möglich ist, alle Informationen auf einmal aufzunehmen und eine gewisse Einschränkung der Werte wünschenswert wäre.

Weiters war es für einige Teilnehmer verwirrend, dass aktive Werte unterschiedlich dargestellt werden. Zwei von sechs Teilnehmern haben bis zum Ende der Szenarien nicht gemerkt, dass es überhaupt eine unterschiedliche Darstellung gibt.

Schließlich konnte auch noch die automatische Positionierung des sichtbaren Seitenbereiches als Problem identifiziert werden. Fünf von sechs Teilnehmern äußerten sich negativ über diese Funktion und würden das Interface ohne Ankersprung bevorzugen.

Bei der Frage nach einer Reihung der vier getesteten Interfaces wurde das Such-Interface von Geizhals von fünf der sechs Teilnehmer als Letztes genannt.

### 5.3.2 Interface 2: Drupal's Faceted Search Modul

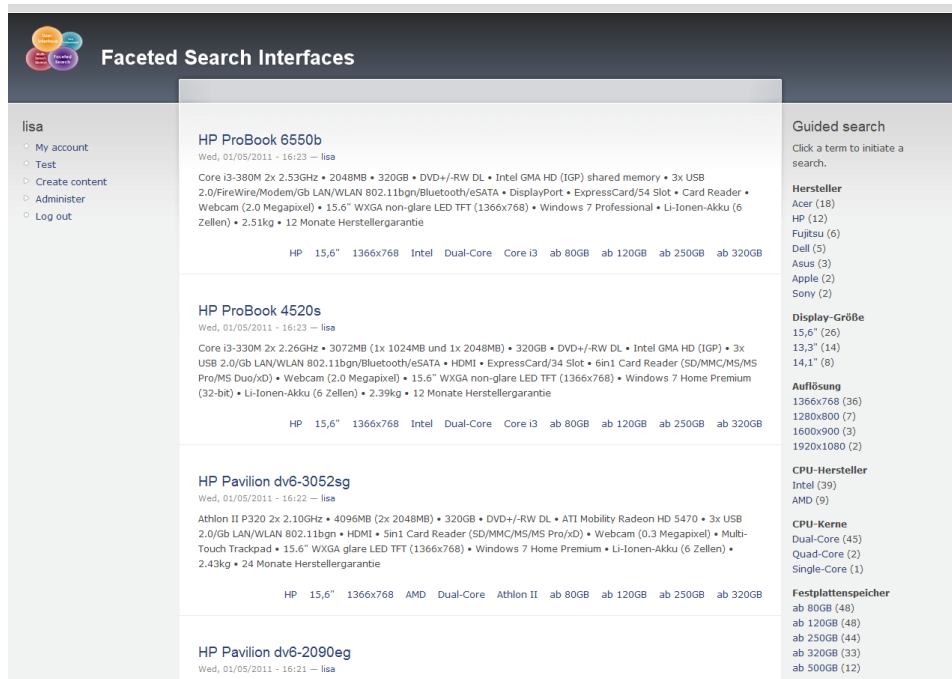
Da die neu entwickelten Interfaces auf dem ursprünglichen Interface des Faceted Search Moduls aufgebaut sind, wurde auch dieses Interface (siehe Abbildung 5.2) für einen Vergleich getestet.

Teilnehmer, welche durch die wechselnde Reihenfolge dieses Interface als erstes Interface zum Testen erhielten, mussten sich zu Beginn des ersten Szenarios erst einen Überblick über die gesamte Seite verschaffen. Durch die einfärbige Darstellung der Werte und Facetten sprang das Such-Interface als solches nicht sofort ins Auge. Teilnehmer, welche schon aus vorhergehenden Interfaces mit dem generellen Seitenaufbau vertraut waren, konnten auch dieses Suchinterface sofort identifizieren. Der Suchvorgang im Szenario an sich konnte von allen Teilnehmern erfolgreich durchgeführt werden.

Während der Durchführung des zweiten Szenarios traten speziell bei einem Teilnehmer Probleme und Unklarheit auf.

Dieses Interface hat, wie das Interface von Geizhals, von Anfang an alle Facetten und Werte aufgedeckt. Da bei den Drupal Interfaces nicht dieselbe Datenmenge wie bei Geizhals vorhanden ist, wirkt es anfangs nicht derart überfordernd. Ein Teilnehmer bemerkte jedoch, dass sich bei der gleichen Datenmenge wahrscheinlich exakt dasselbe Problem wie bei Geizhals entwickeln könnte.

Die Durchführung des dritten Szenarios hat keinem der Teilnehmer Probleme bereitet. Durch die bereits angesprochene Aufdeckung der Werte von



**Abbildung 5.2:** Such-Interface des Drupal Moduls Faceted Search in der Startansicht.

Anfang an, wurde die gesuchte Anzahl der Produkte ohne Verzögerung gefunden.

Im vierten Szenario kam es bei drei von sechs Teilnehmern zu einer kurzzeitigen Verwirrung. Dieses Interface ist so implementiert, dass Werte mit einer Anzahl von null Produkten im Query-Preview (also keiner Übereinstimmung mit bereits gewählten Produkten) aus der Liste entfernt werden. Dies stammt daher, dass das Interface ursprünglich nicht dafür gedacht ist, eine Mehrfachauswahl von Werten anzubieten, sondern lediglich eine Einzelauswahl mit gegebenenfalls hierarchischen Facetten ermöglicht.

Bei der Auswahl der sechs beliebigen Werte haben drei der Teilnehmer dies angesprochen. Sie haben nicht verstanden, warum die Werte plötzlich verschwunden waren. Von den Teilnehmern wurden letztendlich andere Werte gewählt und das Szenario konnte von jedem Teilnehmer erfolgreich abgeschlossen werden.

Das fünfte Szenario stellte für einen Teilnehmer ein großes Problem dar. Dieser hatte bereits bei zwei vorhergehenden Interfaces im Test die Funktion des „Current-Search“-Blocks nicht verstanden und tat sich daher auch bei diesem Interface schwer, die aktiven Facetten und Werte zu finden. Der Teilnehmer versuchte, über Blättern im Suchergebnis gemeinsame Eigenschaften herauszufiltern, gab aber nach einiger Zeit auf und konnte das Szenario nicht



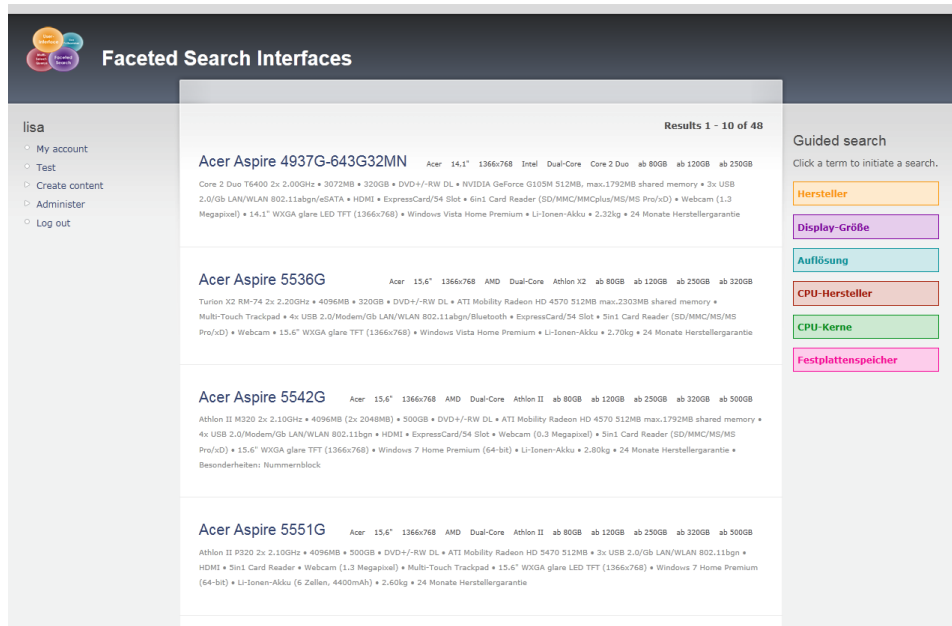


Abbildung 5.3: Ansicht des Blockinterfaces im Ausgangszustand.

erfolgreich abschließen. Die restlichen Teilnehmer konnten dieses Szenario ohne Probleme abschließen.

Generell konnte während des Usability-Tests bei diesem Interface erkannt werden, dass durch den ursprünglichen Verwendungszweck leitende Interface-Elemente gefehlt haben. Man merkte den Teilnehmern an, dass zwar ein oberflächliches Verständnis für das Interface vorhanden war, jedoch viele unterbewusste Entscheidungen vom Interface nicht abgenommen wurden.

Positiv am Interface wurde die dauerhafte Darstellung aller Werte und Facetten bemerkt. Ein Teilnehmer merkte an, dass es einfach ist, sich schnell einen Überblick über alle Suchmöglichkeiten zu verschaffen, aber dass dies auch von der Menge an Facetten abhängig ist. Würde man dieses Interface bei einem Datenumfang wie bei Geizhals anwenden, würde auch bei diesem Interface das Problem der Unübersichtlichkeit auftreten.

### 5.3.3 Interface 3: Blockinterface

Im Gegensatz zu den Such-Interfaces von Drupal und Geizhals, wussten die Teilnehmer beim ersten Anblick der gesamten Seite (siehe Abbildung 5.3) sofort, wo sich dieses Interface befindet. Die farbliche Hervorhebung stach allen Teilnehmern sofort ins Auge. Ein Teilnehmer merkte an, dass er den Rest der Seite anfangs gar nicht wahr nahm, da er sich sofort auf das Interface konzentrierte. Durch diese schnelle Identifikation des Interfaces stellte auch das erste Szenario kein Problem dar.

Ein Teilnehmer merkte beim ersten Eindruck des Interfaces an, dass die farbliche Gestaltung eher nicht seinem persönlichen Geschmack entsprach, aber er sich vorstellen könne, dass speziell die farblichen Unterschiede für den durchschnittlichen Benutzer eine Zuordnung der Werte zu den Facetten leichter machen würde.

Auch das zweite Szenario wurde von allen Teilnehmern erfolgreich gelöst, es konnten keine Probleme im Umgang erkannt werden.

Das dritte Szenario löste bei zwei von sechs Teilnehmern kurzzeitig Unsicherheit aus. Da die Werte durch Klick auf die Facette angezeigt werden, musste zur Durchführung dieses Szenarios ein Klick auf die Facette getätigt werden, um die geforderte Anzahl der Intel-Produkte herausfinden zu können. Durch die Formulierung des Szenarios (siehe Szenario 3 in Abschnitt 5.1.1) waren sich diese Teilnehmer nicht sicher, ob dieser eine Klick bereits als Start der Suche gilt oder nicht. Beide Teilnehmer fragten nach, ob dieser Klick „erlaubt“ sei und konnten anschließend wie die restlichen Teilnehmer auch dieses Szenario erfolgreich beenden.

In der Durchführung des vierten Szenarios konnte eindeutig der Vorteil des dynamischen Nachladens der Interface-Komponenten erkannt werden. Die Wahl der sechs beliebigen Eigenschaften sowie die anschließende Limitierung auf nur drei, wurde mit Abstand am schnellsten von allen Interfaces durchgeführt. Während die Teilnehmer bei normal implementierten Interfaces jedes Mal auf die neu geladene Seite warten mussten, wurden hier die Klicks intuitiv schneller eingesetzt. Speziell bei der Entfernung der Werte wurden die Entfernen-Links im „Current Search“-Block ohne Zeitverzögerung oder Warten seitens der Teilnehmer getätigt.

Ein Nachteil konnte jedoch durch die Schnelligkeit bemerkt werden. Drei der sechs Teilnehmer haben bei der Wahl der Eigenschaften nicht mehr aktiv überlegt, sondern die nächstbeste Eigenschaft gewählt. Dabei wurden auch teilweise Werte mit einer Anzahl von null Produkten im Query-Preview ausgewählt. Dies führte bei zwei Teilnehmern zu einem leeren Suchergebnis.

Während der Durchführung des fünften Szenarios wurde wieder deutlich, dass einer der sechs Teilnehmer die Funktion des „Current Search“-Blocks nicht verstanden hatte. Dieser Teilnehmer versuchte auch bei diesem Interface erfolglos, die aktiven Werte und Facetten über Blättern im Suchergebnis über Gemeinsamkeiten der dort angeführten Produkte zu finden.

Die restlichen Teilnehmer konnten dieses Szenario problemlos durchführen. Jeder dieser Teilnehmer scrollte sofort bewusst zum Anfang der Seite und nutzte den „Current-Search“-Block zur Beantwortung der Frage nach den aktiven Facetten und Werten. Zwei der Teilnehmer gruppieren sogar die Werte bei der mündlichen Auflistung intuitiv nach Facetten.

Dieses Interface wurde von fünf der sechs Teilnehmer als Erstes bei der Frage nach einer Reihung genannt. Positiv bezeichnet wurde die farbliche Trennung der einzelnen Facetten und die dadurch einfache Zuordnung der Werte zu den Facetten im „Current Search“-Block. Weiters wurde auch die

Möglichkeit geschätzt, Werte von Facetten beliebig verstecken oder anzeigen zu können. Die Teilnehmer bewerteten dieses Interface als sehr übersichtlich und gut strukturiert.

Als großer Vorteil konnte auch die Schnelligkeit der Bedienung des Interfaces erkannt werden. Die Teilnehmer haben den Unterschied in der Ladezeit der Interfaces nicht aktiv kommentiert, jedoch konnte man bei der Bedienung eindeutig mitverfolgen, dass alle Vorteile des dynamisch nachladenden Interfaces von den Teilnehmern unbewusst genutzt wurden.

Bei diesem Interface ist es (wie in Abschnitt 3.2 besprochen) möglich, ausgewählte Werte über zwei verschiedene Wege zu entfernen. Einerseits kann der Wert per Klick auf den Wert selbst im Interface entfernt werden. Andererseits liefert auch ein Klick auf den Entfernen-Link im „Current Search“-Block dasselbe Ergebnis. Die meisten Teilnehmer haben im Laufe der Szenarien beide Wege entdeckt, jedoch nutzten alle Teilnehmer vorzugsweise den „Current Search“-Block. Dies wurde bei Nachfrage auch von allen Teilnehmern bestätigt, wobei drei von sechs Teilnehmern anmerkten, dass es von Vorteil ist, beide Möglichkeiten nutzen zu können.

Zwei der sechs Teilnehmer würden sich bei diesem Interface zur Unterstützung noch einen „alle“-Button für die Werte der Facetten wünschen. Von zwei Teilnehmern wurde auch angesprochen, dass es interessant wäre, zu Beginn nicht alle Werte zu verstecken, sondern die relevantesten Werte anzuzeigen und in weiterer Folge den Rest der Werte gezielt verstecken oder anzeigen zu können.

#### 5.3.4 Interface 4: Ellipseninterface

Der Usability-Test brachte für dieses Interface (siehe Abbildung 5.4) die größte Überraschung. Wie in Abschnitt 3.3 besprochen, wurde dieses Interface mittig auf der Seite in Ellipsenform angeordnet. Speziell die eher ungewöhnliche elliptische Anordnung der Komponenten im Interface war für fast alle Teilnehmer ungewöhnlich, was auch zwei der sechs Teilnehmer direkt beim ersten Anblick des Interfaces bestätigten.

Das erste Szenario stellte bei diesem Interface grundsätzlich kein Problem für die Teilnehmer dar. Ein Teilnehmer (mit diesem Interface als erstes in der Test-Rotation) war kurzzeitig verwirrt mit der Fragestellung (siehe Szenario 1 im Abschnitt 5.1.1), da ihm nicht ganz verständlich war, wie genau die Suche gestartet wird. Wahrscheinlich ist jedoch, dass hier das Szenario etwas unglücklich formuliert war: Der Teilnehmer startete nach kurzer Überlegung die Suche richtig (per Auswahl eines Wertes) und hatte auch bei den folgenden Interfaces kein Problem mehr, den logischen Einsprungspunkt zu finden.

Auch das zweite Szenario wurde von allen Teilnehmern erfolgreich beendet. Während fünf von sechs Teilnehmer für das Entfernen der Werte wieder den „Current Search“-Block verwendeten, hat ein Teilnehmer die Entfernen-

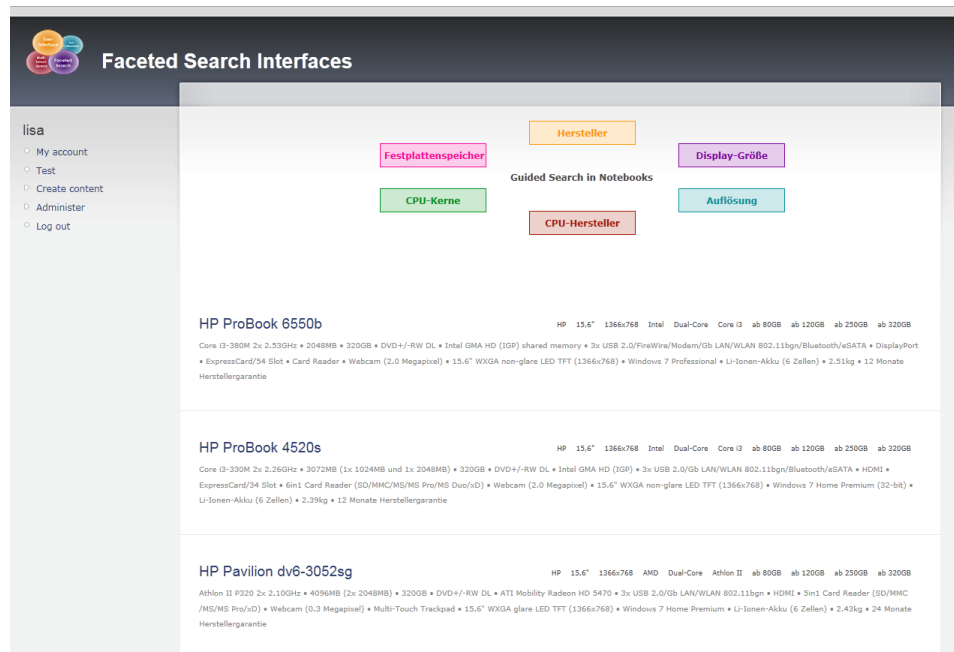


Abbildung 5.4: Ansicht des Ellipseninterfaces im Ausgangszustand.

Funktion durch erneuten Klick der Wert-Blöcke direkt im Interface genutzt.

Während der Durchführung des dritten Szenarios konnten keine Probleme im Umgang mit dem Interface erkannt werden – alle Teilnehmer fanden die geforderte Produktanzahl auf Anhieb.

Wie bereits bei den Testergebnissen der vorhergehenden Interfaces angemerkt wurde, gab es einen Teilnehmer der bis zum letzten getesteten Interface den Sinn des „Current Search“-Blockes nicht verstanden hatte. Dieser Teilnehmer bekam in der Testreihenfolge dieses Interface als Erstes zugeteilt. Im vierten Szenario konnte man das fehlende Verständnis zum ersten Mal sehr gut erkennen. Der Teilnehmer hatte bereits mehrere Werte der vorhergehenden Szenarien aktiv ausgewählt, hat dies aber nicht erkannt und zählte die neu gewählten Werte gedanklich mit. Zum Entfernen der Werte nutzte der Teilnehmer das Wegklicken der Werte im Interface und zählte wieder mit. Er war sich jedoch nie absolut sicher, wie viele Werte nun wirklich aktiv waren. Die restlichen Teilnehmer hatten bei der Durchführung dieses Szenarios kein Problem.

Das fünfte Szenario stellte wieder ein großes Problem für den eben angesprochenen Teilnehmer dar. Der Teilnehmer hat zu allererst den Begriff „scrollen“ nicht verstanden, hat aber auch nicht daran gedacht, die aktiven Werte eventuell im ursprünglichen Interface zu suchen. Stattdessen hat der Teilnehmer erfolglos versucht, durch Blättern im Suchergebnis Gemeinsamkeiten der verschiedenen angeführten Produkte zu finden und nannte diese

als aktive Werte. Die Werte konnten dann natürlich keinen Facetten zugeordnet werden und stimmten auch nicht mit den wirklich aktiven Werten überein.

Generell kann bei diesem Interface gesagt werden, dass die ungewöhnliche Anordnung der Komponenten in Ellipsenform für die Teilnehmer ungewohnt war. Alle Teilnehmer haben bestätigt, eine klassische vertikale Anordnung in Blockform dieser elliptischen Anordnung vorzuziehen. Auch die Platzierung des gesamten Interfaces oben in der Mitte der Seite hat daran nichts ändern können.

Ein Teilnehmer fand es störend, dass die Werte nach dem Aufklappen teilweise Facetten überlappen. Er merkte an, dass dies bei einer größeren Anzahl von Werten innerhalb den Facetten sicher sehr unübersichtlich wird und dass er sich nicht vorstellen könnte, wie das dann aussehen oder funktionieren soll.

Zwei der sechs Teilnehmer haben während der Durchführung aller Szenarien in diesem Interface den „Current Search“-Block bis zum Ende hin nicht entdeckt. Zur Entfernung der Werte nutzten diese Teilnehmer, wenn überhaupt, die Möglichkeit eines erneuten Klicks auf den Wert direkt im Interface. Auf die Frage hin, ob sie denn den „Current Search“-Block nicht entdeckt hätten, gaben diese zwei Teilnehmer tatsächlich an, diesen Bereich bis zum Ende hin übersehen zu haben. Der Block wäre laut den Teilnehmern nicht auffallend genug und „gehe neben den farbigen Facetten und Werten unter“.

Durch die wechselnden Icons in den Werten wurde die Möglichkeit der Entfernung über erneuten Klick von allen Teilnehmern relativ schnell erkannt, jedoch begrenzt genutzt. Selbige Icons in den Facetten-Überschriften haben jedoch eher verwirrt als geholfen.

Der Vorteil des schnellen und dynamischen Nachladens konnte auch bei diesem Interface beobachtet werden.

Bei der Frage nach einer Reihung wurde dieses Interface von fünf von sechs Teilnehmer hinter allen anderen aber vor Geizhals gereiht. Auf die Frage nach dem Grund, gaben die Teilnehmer an, hauptsächlich von der ungewöhnlichen Anordnung irritiert gewesen zu sein und eine klassische Anordnung, wie diese im Blockinterface, bevorzugen.

## 5.4 Diskussion

Bei der Entwicklung eines Interfaces für Faceted Search muss man sich oft die Frage stellen: „Ist diese Funktionalität überhaupt nötig und wie wirkt sie sich auf das Verhalten der Benutzer aus?“. Viele der existierenden Interfaces nutzen keine neuen Technologien, da Benutzer offensichtlich gerne mit altbekannten Methoden arbeiten. Es muss abgewogen werden, ob die Neuerung wirklich hilfreich ist, oder der Benutzer verwirrt wird.

Bei Design und Implementierung der neuen Interfaces (siehe Kapitel 3) wurde versucht, gut funktionierende Komponenten bestehender Interfaces zu übernehmen und mit neuer und optimierter Funktionalität auszustatten. Diese neuen Funktionen betreffen unter anderem die Verwendung von neuen Technologien wie Ajax und jQuery, die bei der Entwicklung eines Interfaces viel mehr Möglichkeiten bieten als herkömmliche Technologien.

Obwohl vor der Evaluierung das Ellipseninterface als neuartiger und dynamischer angesehen wurde, hat sich im Usability-Test definitiv das Blockinterface durchgesetzt. Dieses Interface überfällt den Benutzer nicht mit vielen Neuerungen, sondern lehnt sich an klassische Interfaces an, bietet jedoch im Detail neue Funktionalität.

#### 5.4.1 Darstellung der Facetten und Werte

Ein großer Vorteil des Geizhals Interfaces liegt darin, dass die Facetten und Werte von Anfang an sichtbar sind und auch bleiben. Das Interface verliert dadurch zwar an Übersichtlichkeit und löst bei Benutzern ohne Fachkenntnis teilweise Unsicherheit aus (siehe Abschnitt 5.3.1). Andererseits ist jedoch anzumerken, dass ein schneller Zugriff auf die Werte möglich ist, ohne vorher in verdeckten Facetten nach den gewünschten Werten suchen zu müssen.

Diese fixe Darstellung der Werte und Facetten fehlt im Blockinterface sowie auch im Ellipseninterface. Für Benutzer, die einen schnellen Zugang zu den Suchergebnissen wünschen, da die betreffende Suchanfrage schon im Vorhinein bekannt und nur das Ergebnis relevant ist, ist die Auflistung der relevantesten Werte sinnvoll. Die Relevanz muss im Vorhinein bestimmt werden und es gibt mehrere Möglichkeiten diese abzuleiten (Menge der Produkte, Meistgeklickt, usw.).

Bezogen auf das Blockinterface wäre es ideal, die Werte mit den meisten Produkten als Vorschau anzuzeigen und die restlichen Werte mit Hilfe eines „mehr“-Links anzuzeigen oder verstecken zu können. Die aktuelle Funktionalität für das Anzeigen und Verstecken von kompletten Facetten bzw. deren Werten hat sich während der Usability-Studie definitiv als hilfreich und praktisch erwiesen. Die gruppierte Darstellung der Werte und Facetten erhöht die Übersichtlichkeit enorm.

Die elliptische Anordnung des Ellipseninterfaces dagegen ist definitiv nicht sinnvoll. Die Teilnehmer der Studie zeigten sich irritiert und konnten sich mit dieser neuen Darstellung nicht anfreunden. Die grundsätzliche Idee der elliptischen Anordnung war die Eliminierung des Platzproblems bei der Mehrfachauswahl von Werten (siehe Abschnitt 2.2) und der Positionierung des Interfaces im sogenannten „Sweet-Spot“ des Bildschirms (siehe Abschnitt 3.3). Diese Kombination hat leider nicht funktioniert. Soll das Interface unbedingt oben mittig positionieren werden, müsste eine andere Lösung gefunden werden, um das Interface übersichtlich und funktionell zu gestalten. Inwieweit die Übersichtlichkeit eines Interfaces wichtig ist, zeigt die Tatsache, dass das

Ellipseninterface trotz der ungewohnten Anordnung im Usability-Test in der Reihung immer noch vor Geizhals genannt wurde.

### 5.4.2 Hervorhebung aktiver Werte

Eine gute Darstellung der aktiven Facetten und Werte ist enorm wichtig um zu gewährleisten, dass dem Benutzer zu jeder Zeit klar gemacht wird, was er gerade macht und welche Werte nun wirklich sein Suchergebnis beeinflussen haben.

Im Blockinterface sowie auch bei der Entwicklung des Ellipseninterfaces wurde speziell darauf geachtet, dem Benutzer so viele Hinweise wie möglich zu liefern, welche Werte und Facetten im Moment aktiv sind. Neben dem „Current Search“-Block (siehe Abschnitt 5.4.3) wurden daher auch die Werte und Facetten im Interface selbst durch gewisse Methoden hervorgehoben.

Die aktiven Facetten werden im Blockinterface fett dargestellt und die Überschriften bei mindestens einem aktiven Wert mit einem Icon gekennzeichnet. Dies wurde von den Teilnehmern zwar aktiv nicht kommentiert, jedoch konnte beobachtet werden, dass diese Methoden definitiv hilfreich sind.

Im Ellipseninterface dagegen werden die Werte auch durch Icons als aktiv gekennzeichnet. Dies funktionierte prinzipiell, jedoch trat Verwirrung auf, als bei aktiven Werten auch die Facetten-Überschriften selbiges Icon zur Kennzeichnung verwendet wurde. Das Icon in den Wert-Blöcken soll eine Funktion implizieren, während das Icon in den Facetten-Überschriften rein als optischer Indikator für aktive Werte in der Facette verwendet wird.

Im Geizhals Interface und dem Interface des Drupal Faceted Search Moduls gab es eindeutig die größten Schwachstellen in der Darstellung aktiver Werte. Beide Interfaces sind nicht konsistent in der Hervorhebung von aktiven Werten. Während im Interface von Geizhals ein Unterschied für die Art der Auswahl gemacht wird, sind im Interface des Faceted Search Modul die aktiven Werte normalerweise fett dargestellt. Wählt man bei diesem Interface jedoch mehrere Werte aus, sind einige Werte fett hervorgehoben und einige nicht. Dieses Problem hat einige Teilnehmer der Studie irritiert und ist ein schönes Beispiel für die Notwendigkeit von Konsistenz im Verhalten von Interfaces.

Die Darstellung der aktiven Werte im Interface von Geizhals schneidet im Vergleich mit den restlichen besprochenen Interfaces merklich am Schlechtesten ab. Teilweise konnten von den Teilnehmern nicht alle aktiven Werte erkannt werden und die Facetten sind bei aktiven Werten überhaupt nicht hervorgehoben.

### 5.4.3 „Current Search“

Wie soeben in Abschnitt 5.4.2 angesprochen, sind Hervorhebungen von Zustandsänderungen (z. B. Auswahl von Werten) wichtig um dem Benutzer das Gefühl von Kontrolle im Interface zu vermitteln.

Als gute Lösung hat sich hier der „Current Search“-Block des Blockinterfaces etabliert. Jeder der Teilnehmer der Studie, der die Funktionsweise des Blockes erkennen konnte, hatte auch bei versteckter Suche sofort einen Überblick über alle aktiven Werte und konnte diese auch problemlos den Facetten zuordnen. Die farbliche Unterteilung der Facetten bewährte sich besonders.

Während der Usability-Studie konnte, speziell im Ellipseninterface, festgestellt werden, dass bei Verwendung eines „Current Search“-Blocks, dieser auch prominent genug gestaltet sein muss. Diese Funktionalität ist umsonst, wenn sie vom Benutzer nicht erkannt wird.

Generell ist zum „Current Search“-Block zu sagen, dass es definitiv von Vorteil ist, die Facetten und Werte noch einmal separat vom eigentlichen Suchinterface aufzulisten. Es sollte möglich sein, Werte im Interface und im „Current Search“-Block zu entfernen, um so den bestmöglichen Arbeitsfluss zu garantieren.

### 5.4.4 Suchergebnisse

Wie bereits in Abschnitt 3.1.3 angesprochen, lag das Hauptaugenmerk auf dem Interface an sich und die Suchergebnisse wurden eigentlich außen vor gelassen. Trotzdem ist es notwendig, ein übersichtliches Suchergebnis zu präsentieren, damit das Interface an sich angemessen getestet werden kann.

Da die Szenarien des Usability-Tests eher weniger auf den Umgang mit den Suchergebnissen abzielte, kann zur Auswirkung der Darstellung wenig gesagt werden. Lediglich bei dem Teilnehmer, der Probleme mit dem Verständnis der aktiven Facetten hatte (siehe Abschnitt 5.3.4), konnte beobachtet werden, dass die bei den Drupal Interfaces angezeigten vorhandenen Eigenschaften neben den Produkten (siehe Abbildung 5.5) eher zur Verwirrung führen als dass sie helfen. Hilfreich wäre hier vermutlich, eine, wie im Interface von Geizhals zu sehende, tabellarische Darstellung (siehe Abbildung 5.6) der Produkte einzuführen und die Eigenschaften in weiterer Folge in den Detailansichten oder unter den kurzen Beschreibungstexten anzuzeigen. An der derzeitigen Position fallen sie zu sehr auf und stören die Ansicht.

Für Benutzer ist interessant, ob die Suchanfrage Ergebnisse brachte und wenn ja, wieviele Objekte gefunden werden konnten. Im Interface von Geizhals ist es eines der größten Probleme, dass das Interface mehr als die gesamte Bildschirmfläche in Anspruch nimmt und der Benutzer nach dem Absenden einer Suchanfrage zuerst überhaupt keinen Hinweis erhält, ob überhaupt Suchergebnisse gefunden wurden. Dafür muss der Benutzer scrollen und auch



<b>HP ProBook 4520s</b>	HP	15,6"	1366x768	Intel	Dual-Core	Core i3	ab 80GB	ab 120GB	ab 250GB	ab 320GB
Core i3-330M 2x 2.26GHz • 3072MB (1x 1024MB und 1x 2048MB) • 320GB • DVD+/-RW DL • Intel GMA HD (IGP) • 3x USB 2.0/Gb LAN/WLAN 802.11bgn/Bluetooth/eSATA • HDMI • ExpressCard/34 Slot • 6in1 Card Reader (SD/MMC/MS/MS Pro/MS Duo/xD) • Webcam (2.0 Megapixel) • 15.6" WXGA non-glare LED TFT (1366x768) • Windows 7 Home Premium (32-bit) • Li-Ionen-Akku (6 Zellen) • 2.39kg • 12 Monate Herstellergarantie										
Herstellergarantie										

<b>HP Pavilion dv6-3052sg</b>	HP	15,6"	1366x768	AMD	Dual-Core	Athlon II	ab 80GB	ab 120GB	ab 250GB	ab 320GB
Athlon II P320 2x 2.10GHz • 4096MB (2x 2048MB) • 320GB • DVD+/-RW DL • ATI Mobility Radeon HD 5470 • 3x USB 2.0/Gb LAN/WLAN 802.11bgn • HDMI • 5in1 Card Reader (SD/MMC/MS/MS Pro/xD) • Webcam (0.3 Megapixel) • Multi-Touch Trackpad • 15.6" WXGA glare LED TFT (1366x768) • Windows 7 Home Premium • Li-Ionen-Akku (6 Zellen) • 2.43kg • 24 Monate Herstellergarantie										

Abbildung 5.5: Darstellung der Suchergebnisse in Drupal.




Bild	2449 Artikel ▾	Bewertung	Einträge	LZ	Preis in €	Händler mit Bestpreis
	<a href="#">Acer Aspire 3750-2314G50Mnkk (LX.RGR02.009/LX.RGR02.021)</a> Core i3-2310M 2x 2.10GHz • 4096MB • 500GB • DVD+/-RW DL • Intel GMA HD 3000 (IGP) shared memory • 3x USB 2.0, 1x USB ...	(zu wenige)	11	<input type="checkbox"/>	ab 577,68	<a href="#">Handytuner.at</a>
	<a href="#">Acer Aspire 3750-2414G50Mnkk (LX.RGV02.007)</a> Core i5-2410M 2x 2.30GHz • 4096MB • 500GB • DVD+/-RW DL • NVIDIA GeForce GT520M 1024MB • 3x USB (2x USB 2.0, 1x USB ...	(zu wenige)	1	<input type="checkbox"/>	704,67	<a href="#">Amazon.at</a>
	<a href="#">Acer Aspire 3750G-2414G50Mnkk (LX.RGV02.022)</a> Core i5-2410M 2x 2.30GHz • 4096MB • 500GB • DVD+/-RW DL • NVIDIA GeForce GT520M 1024MB • 3x USB (2x USB 2.0, 1x USB ...	(zu wenige)	1	<input type="checkbox"/>	704,67	<a href="#">redcoon.at</a>

Abbildung 5.6: Darstellung der Suchergebnisse bei Geizhals.

dann dauert es teilweise lange, bis die Anzahl der Suchergebnisse gefunden wird (siehe Abschnitt 5.3.1).

Im Gegensatz zu Interface von Geizhals waren bei den drei anderen getesteten Interfaces die Suchergebnisse immer sofort im Bild und auch die Anzahl der Suchergebnisse konnte bei jedem Interface sofort ohne scrollen gefunden werden. Die Teilnehmer der Studie bekamen sofort Rückmeldung auf getätigte Suchanfragen und haben auch gleich bemerkt, wenn keine Produkte zu den angegebenen Werten gefunden werden konnten.

### Leere Suchergebnisse

Ein Designkriterium bei der Erstellung von Faceted Search Interfaces ist das Vermeiden von sogenannten leeren Suchergebnissen [5]. Wie in Abschnitt 3.1.2 beschrieben, wurden bei allen Interfaces Query-Previews eingesetzt, um die Anzahl der zugehörigen Produkte zu den Werten im Vorhinein bestimmen zu können. Bei allen Interfaces ist es jedoch möglich, auch Werte mit einem Preview von null zur Suchquery hinzuzufügen. Lediglich im Standardinterface des Drupal Faceted Search Moduls tritt der Fall ein, dass Werte mit einer Produktanzahl von null ausgeblendet werden.

Dieses Ausblenden der Werte sorgte unter den Teilnehmern der Studie zwischenzeitlich für Verwirrung (siehe Abschnitt 5.3.2). Den Teilnehmern war nicht klar, warum nach der Wahl eines bestimmten Wertes plötzlich weit

weniger Werte zur Auswahl standen als vorhin. Fraglich ist, ob es besser ist, den Benutzern die Möglichkeit der Wahl von Werten mit null Ergebnissen zu bieten, oder diese kleine Verwirrung in Kauf zu nehmen.

Da dieser Aspekt nicht Teil der Usability-Studie war, kann nur eine Vermutung angestellt werden, welche Methode nun zielführender ist. Interessant ist die Darstellung der Werte mit der gewohnten Query-Preview aber gleichzeitig das Verhindern der Auswahl dieses bestimmten Wertes durch Entfernung des Links. So würden alle Werte konsistent in der Ansicht verbleiben, der Benutzer jedoch vor einem leeren Suchergebnis geschützt sein.

#### 5.4.5 Position der Interfaces

Während der Tests konnte bei den Teilnehmern keine Präferenz für eine bestimmte Position des Suchinterfaces auf der Webseite erkannt werden.

Wie in Kapitel 3 besprochen, wurden die beiden neu implementierten Interfaces absichtlich unterschiedlich positioniert. Theoretisch sollte das oben mittig positionierte Ellipseninterface die Benutzer von der Position her mehr ansprechen als das seitlich rechts positionierte Blockinterface. Praktisch war es jedoch so, dass die elliptische Anordnung der Werte im Ellipseninterface einen Vergleich nicht möglich machen. Die Teilnehmer der Studie waren mit dieser Art der Darstellung nicht vertraut und gaben im Nachhinein an, klassische Darstellungsmethoden zu bevorzugen. Auch auf die direkte Frage nach der bevorzugten Positionierung, gaben die Teilnehmer an, das Blockinterface seitlich rechts besser zu finden. Fraglich ist, ob dies wirklich von der Positionierung an sich abhängig ist oder die elliptische Anordnung den Gesamteindruck negativ beeinflusst hat.

Hilfreich für eine Evaluierung genau dieser Interface-Komponente wäre ein Vergleich zwischen zwei Interfaces, in denen die restlichen Komponenten ident, jedoch die Positionierung unterschiedlich ist. Auf diesem Weg wäre es einfach, die Auswirkungen der Position genau zu bestimmen, was bei dieser Evaluierung leider nicht möglich war.

#### 5.4.6 Dynamisches Nachladen von Interface-Elementen

Das dynamische Nachladen von Elementen in Interfaces wird definitiv immer häufiger gesehen. Hier können leider keine definitiven Zeitvergleiche angestellt werden, da die Interfaces, zusätzlich zum Unterschied der Ladezeit, signifikant unterschiedlich sind.

Beobachtet werden konnte jedoch mit Sicherheit, dass sich diese Funktionalität eindeutig positiv auf das Verhalten der Benutzer auswirkt. Generell wurden Klicks in kürzeren Abständen getätigt, während bei der Benutzung des Interfaces von Geizhals und jenes des Drupal Faceted Search Moduls jedes Mal auf das Laden der Seite gewartet werden musste.

Zusätzlich zur Erhöhung der Schnelligkeit kann bestätigt werden, dass

durch das dynamische Nachladen von Elementen auch die Konsistenz des Interfaces verbessert werden kann. Da das Interface an sich nur zum ersten Aufruf der Seite geladen wird und anschließend nur wichtige Komponenten einzeln nachgeladen werden, verändert sich das Interface zwischen einer Wahl von Werten kaum. Lediglich die Zahlen in den Query-Preview werden geändert, sowie das Suchergebnis jedes Mal neu nachgeladen und der „Current Search“-Block neu generiert und aktualisiert. Hiermit kann erfolgreich verhindert werden, dass sich das Interface, in einer für den Benutzer undurchsichtigen Weise, verändert.

#### 5.4.7 Zeit und Zufriedenheit

Sehr interessant ist der Vergleich der benötigten Dauer zur Durchführung der verschiedenen Szenarien in den Interfaces mit der generellen Zufriedenheit der Teilnehmer der Usability-Studie mit diesen Interfaces.

Wie in Abschnitt 5.3 bereits besprochen wurde, erreichte das Blockinterface unter den Teilnehmern die größte Zufriedenheit. Im direkten Vergleich schnitt dieses Interface mit Abstand am Besten ab und die Teilnehmer kommentierten speziell die Übersichtlichkeit und behaupteten von sich selbst, am Besten mit diesem Interface arbeiten zu können.

Betrachtet man jedoch die durchschnittlich benötigte Zeit pro Interface, liegt das Blockinterface deutlich hinter dem Ellipseninterface und dem Interface von Drupal. Dies kann nun mehrere Gründe haben. Sehr wahrscheinlich ist die Tatsache, dass das Interface des Faceted Search Moduls alle Werte bereits zu Beginn darstellt. Im Blockinterface muss also bei jeder Auswahl eines einzigen Werts ein Klick mehr gemacht werden, wenn die Werte zu diesem Zeitpunkt noch nicht sichtbar sind. Wird dies auf die große Menge von Klicks die getätigt wurden hochgerechnet, kann man einen deutlichen Zeitverlust erkennen.

Weiters ist in Tabelle 5.2 in Abschnitt 5.3 ersichtlich, dass speziell Teilnehmer 3 eine deutlich längere Zeit für dieses Interface benötigt hat. Dieser Teilnehmer hatte nachweislich am wenigsten Erfahrung mit dieser Art von Interface und begann die Testreihe mit dem Blockinterface. Daraus kann geschlossen werden, dass während des ersten Durchlaufs der Szenarien auch eine gewisse Zeit benötigt wurde, mit den Szenarien und vor allem der Funktionsweise von Faceted Search Interfaces vertraut zu werden. Ableiten lässt sich das auch aus der Tatsache, dass der Teilnehmer für die Durchführung der Szenarien in den restlichen Interfaces (Geizhals ausgenommen) deutlich weniger Zeit benötigte. Der Teilnehmer hat zu keinem Zeitpunkt geäußert, Probleme mit den Interfaces oder Szenarien zu haben. Lediglich zum Interface von Geizhals wurde kommentiert, dass es unübersichtlich wäre und es auch deutlich länger gedauert hat, benötigte Werte zu finden.

Daraus kann also geschlossen werden, dass es einem Benutzer generell egal ist, wie lang der Suchvorgang dauert und wieviele Klicks dafür nötig

sind, solange sich der Benutzer bei der Arbeit mit dem Interface wohl fühlt und generell ein Gefühl von Kontrolle, Konsistenz und Übersichtlichkeit vermittelt bekommt.

## Kapitel 6

# Schlussbemerkungen

Bei der Entwicklung von Faceted Search Interfaces mit der Möglichkeit einer Mehrfachauswahl müssen einige Herausforderungen bewältigt werden. Der Benutzer soll zu jeder Zeit ein Gefühl der Kontrolle über das Interface vermittelt bekommen, immer wissen oder erahnen können wie sich das Interface auf seine Eingaben und Aktionen verhält oder wie gerade getätigte Aktionen rückgängig gemacht werden können. Zusätzlich soll das Suchergebnis die Erwartungen des Benutzers erfüllen und die korrekten Produkte zu den gewählten Facetten und Werten enthalten.

Bereits seit Jahren wird versucht, Faceted Search Interfaces bestmöglich zu optimieren und auch in diesem Dokument werden neue Ansätze zur Entwicklung eines Faceted Search Interfaces diskutiert.

Zur Evaluierung dieser neuen Ansätze wurden zwei Prototypen entwickelt: das Blockinterface und das Ellipseninterface. Im Blockinterface wurde versucht, konventionelle Techniken der Zusammensetzung solcher Interfaces mit neuen Technologien und überarbeiteten Komponenten zu verknüpfen. Das Ergebnis ist ein gut strukturiertes Interface, im Seitenbereich rechts angeordnet mit einer übersichtlichen Blockdarstellung der einzelnen Facetten. Die Werte der Facetten können durch Klick auf die Facetten-Überschrift nach Belieben angezeigt oder versteckt werden. Der Benutzer hat somit jederzeit Zugriff auf alle verfügbaren Werte aller Facetten, ohne dass der Seitenbereich des Interfaces überfüllt oder unübersichtlich wirkt.

Zur Erhöhung der Konsistenz im Interface werden im Blockinterface gewisse Schlüsselkomponenten dynamisch nachgeladen, während das restliche Interface nach dem erstmaligen Ladevorgang nicht wieder nachgeladen wird. Dies betrifft vor allem das Aktualisieren der Suchergebnisse, der Anzahl der Produkte in den Query-Previews und die automatische Generierung des „Current Search“-Blocks.

Wie im Blockinterface werden auch im Ellipseninterface die bereits angesprochenen Komponenten dynamisch aktualisiert, während das restliche Interface im selben Zustand verbleibt. Von dieser ähnlichen Verhaltensweise

abgesehen unterscheidet sich das Ellipseninterface in der Darstellung jedoch merklich vom Blockinterface. Bei der Entwicklung des Ellipseninterfaces wurde versucht, eine neuartige Anordnung der einzelnen Facetten und Werte in elliptischer Form zu implementieren. Das Interface als Ganzes wurde im mittleren, oberen Seitenbereich über den Suchergebnissen platziert und benötigt durch die elliptische Anordnung der Elemente vergleichsweise wenig Platz. Die Werte der Facetten sind wie im Blockinterface im Ausgangszustand versteckt und können durch Benutzeraktionen entweder angezeigt oder versteckt werden.

Durch die Durchführung der Usability-Studie konnten wertvolle Kenntnisse gewonnen werden. Eine ungewohnte Anordnung der Elemente, wie im Ellipseninterface, verursachte bei den Testpersonen Unsicherheit und zeigt sich als sehr gewöhnungsbedürftig. Optimal dagegen ist die Einteilung der Facetten und Werte in Blöcken wie im Blockinterface. Positiv wurde auch die Übersichtlichkeit dieses Interfaces bemerkt. Eine konstante Darstellung aller Werte und Facetten wie im Interface von Geizhals wirkt für die Testpersonen unübersichtlich. Bemängelt wurde auch die fehlende Konsistenz in der Darstellung aktiver Werte.

Die Usability-Studie hat zum Teil unerwartete Ergebnisse geliefert. Nach Fertigstellung der Interfaces wurde prinzipiell damit gerechnet, dass das Ellipseninterface großen Anklang unter den Testpersonen findet. Genau das Gegenteil war jedoch der Fall und zeigt, dass bei der Erstellung eines Faceted Search Interfaces mit Mehrfachauswahl ein Mittelweg gewählt werden sollte. Eine Kombination aus alt-bekanntem Interface-Elementen in Verbindung mit neuen Funktionen scheint sich hier am Besten anzubieten, komplett neue Darstellungen können den Benutzer überfordern.

Faceted Search Interfaces sind keinesfalls veraltet und laufend gibt es Ideen, diese zu verbessern. Der Einsatz neuer Technologien ist definitiv ein Schritt in die richtige Richtung und kann die Effektivität, Vielseitigkeit und Einsetzbarkeit noch um ein Vielfaches erhöhen.

# Kapitel 7

## Inhalt der CD-ROM/DVD

**Format:** CD-ROM, Single Layer, ISO9660-Format

### 7.1 PDF-Dateien

**Pfad:** /

DA.pdf . . . . . Diplomarbeit im PDF-Format  
(Gesamtdokument)

**Pfad:** /docs

Ajax.pdf . . . . . Kopie der Wikipedia-Seite zu Ajax vom  
23.06.2011

CMS.pdf . . . . . Kopie der Wikipedia-Seite zu Content  
Management Systemen vom 04.06.2011

Zeitmessung\_Usability\_Studie.pdf Aufzeichnung aller Zeitmessungen  
im Laufe der Usability-Studie

### 7.2 Quellcode

**Pfad:** /source

drupal.zip . . . . . Überarbeitetes Faceted Search Modul für  
Drupal

blockinterface.zip . . . . Das Faceted Search Modul mit dem  
Blockinterface

ellipseninterface.zip . . . Das Faceted Search Modul mit dem  
Ellipseninterface

### 7.3 Sonstiges

**Pfad:** /images

- \*.pdf . . . . . Original Adobe PDF-Dateien
- \*.png . . . . . Original Rasterbilder



# Literaturverzeichnis

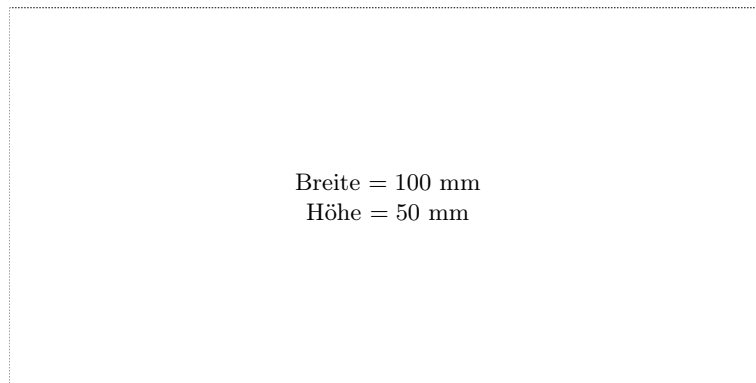
- [1] Ben-Yitzhak, O., S. Yogev, N. Golbandi, N. Har'El, R. Lempel, A. Neumann, S. Ofek-Koifman, D. Sheinwald, E. Shekita und B. Sznajder: *Beyond basic faceted search*. In: *WSDM '08 Proceedings of the International Conference on Web Search and Web Data Mining*, S. 33–44, New York, NY, 2008. ACM.
- [2] English, J., M. A. Hearst, R. Sinha, K. Swearingen und K. P. Yee: *Flexible search and navigation using faceted metadata*. Techn. Ber., University of Berkeley, School of Information Management and Systems, Berkeley, CA, 2002. <http://flamenco.berkeley.edu/papers/flamenco02.pdf>.
- [3] English, J., M. A. Hearst, R. Sinha, K. Swearingen und K. P. Yee: *Hierarchical faceted metadata in site search interfaces*. In: *CHI '02 Extended Abstracts on Human Factors in Computing Systems*, S. 628–629, New York, NY, 2002. ACM.
- [4] Hearst, M. A.: *Design recommendations for hierarchical faceted search interfaces*. In: *SIGIR 2006: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, S. 26–30, New York, NY, Aug. 2006. ACM.
- [5] Hearst, M. A.: *Uis for faceted navigation – recent advances and remaining open problems*. In: *HCIR 2008 Proceedings of the Second Workshop on Human-Computer Interaction and Information Retrieval*, S. 13–17, Redmond, WA, 2008. Microsoft Research.
- [6] Hearst, M. A.: *Search User Interfaces*. Cambridge University Press, New York, NY, 2009.
- [7] Hearst, M. A., A. Elliott, J. English, R. Sinha, K. Swearingen und K. P. Yee: *Finding the flow in web site search*. *Communications of the ACM*, 45:42–49, 2002.
- [8] Hearst, M. A., P. Smalley und C. Chandler: *Faceted metadata for information architecture and search*. Kursunterlagen CHI Course for CHI 2006, 2006. [http://flamenco.berkeley.edu/talks/chi\\_course06.pdf](http://flamenco.berkeley.edu/talks/chi_course06.pdf).

- [9] Hilal, A., F. Scholer, J.A. Thom und M. Wu: *User interaction with novel web search interfaces*. In: *OZCHI '09 Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group: Design: Open 24/7*, S. 301–304, New York, NY, 2009. ACM.
- [10] Hoeber, O., D. Brooks, Michael an Schroeder und X. Dong Yang: *The-hotmap.com : Enabling flexible interaction in next-generation web search interfaces*. In: *WI-IAT '08 Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, S. 730–734, Washington, DC, Dez. 2008. IEEE Computer Society.
- [11] Hoeber, O. und X. Dong Yang: *A comparative user study of web search interfaces: Hotmap, concept highlighter, and google*. In: *WI '06 Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, S. 866–874, Washington, DC, Dez. 2006. IEEE Computer Society.
- [12] Hoeber, O. und X. Dong Yang: *A unified interface for visual and interactive web search*. In: *CIIT '07 The Sixth IASTED International Conference on Communications, Internet, and Information Technology*, S. 16–22, Anaheim, CA, 2006. ACTA Press.
- [13] Karlson, A.K., G.G. Robertson, D.C. Robbins, M.P. Czerwinski und G.R. Smith: *Fathumb : A facet-based interface for mobile search*. In: *CHI '06 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, S. 711–720, New York, NY, 2006. ACM.
- [14] Käki, M.: *Proportional search interface usability measures*. In: *NordiCHI '04 Proceedings of the third Nordic Conference on Human-Computer Interaction*, S. 365–372, New York, NY, 2004. ACM.
- [15] Krug, S.: *Don't make me think*. New Riders, Berkeley, CA, 2. Aufl., 2006.
- [16] Kules, B., R. Capra, M. Banta und T. Sierra: *What do exploratory searchers look at in a faceted search interface?* In: *JCDL '09 Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, S. 313–322, New York, NY, 2009. ACM.
- [17] Müller, W., M. Zech und A. Henrich: *Visualflamenco: Faceted browsing for visual features*. In: *Ninth IEEE International Symposium on Multimedia Workshops ISMW 2007*, S. 71–72, Washington, DC, Dez. 2007. IEEE Computer Society.

- [18] Nielsen, J. und T.K. Landauer: *A mathematical model of the finding of usability problems*. In: *CHI '93 Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, S. 206–213, New York, NY, 1993. ACM.
- [19] Shneiderman, B.: *Designing the user interface: strategies for effective human-computer interaction*. Addison Wesley, Boston, MA, 4. Aufl., 2004.
- [20] VanDyk, J.A. und M. Westgate: *Pro Drupal Development*. Springer Verlag, New York, NY, 2007.

# Messbox zur Druckkontrolle

— Druckgröße kontrollieren! —



— Diese Seite nach dem Druck entfernen! —