Universität Stuttgart

Institut für Parallele und Verteilte Systeme
Abteilung Anwendersoftware

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Fraunhofer IAO

Fraunhofer Institut
für Arbeitswirtschaft und Organisation IAO

Nobelstraße 12
D-70569 Stuttgart

Diplomarbeit Nr. 3171

Methodik zur automatisierten Extraktion und Klassifikation semistrukturierter Produkt- und Adressdaten aus Webseiten

Evgeny Baranovskiy

Studiengang:	Softwaretechnik
Prüfer:	PD Dr. rer. nat. Holger Schwarz
Betreuer:	PD Dr. rer. nat. Holger Schwarz Dipl.-Ing. Dipl.-Inf. Maximilien Kintz (Fraunhofer IAO) M. Sc. Andrea Horch (Fraunhofer IAO)
begonnen am:	28. März 2011
beendet am:	27. September 2011
CR-Klassifikation:	H3.1, H3.2, H3.3, I5.4, I7.5

Kurzfassung

Diese Arbeit stellt eine neue Methodik für die automatisierte Extraktion und Klassifikation von Daten aus Webseiten vor.

Die Methodik EH („Extraction Heuristics“) ist für die Domänen der Produkt- und Adressdaten konzipiert und erlaubt die Erweiterung um zusätzliche Domänen. Der Bedarf nach einer solchen Methodik ist groß, weil die Vielfalt von Informationen auf Websites eine lukrative Datenquelle darstellt. Mit den vorhandenen Werkzeugen und Verfahren lassen sich die Inhalte von Websites nur in einem begrenzten Umfang extrahieren, wobei sich eine Reihe von Nachteilen für den Benutzer ergeben. Zudem bieten die vorhandenen Werkzeuge keinerlei Möglichkeit zur Klassifikation der extrahierten Daten.

Die Methodik EH bietet einen einfachen und erweiterbaren Prozess, der alle Teilaufgaben der Extraktion und Klassifikation von Daten aus Webseiten abdeckt und durch das hohe Maß an Automatisierung den Benutzer entlastet. Mit der prototypischen Implementierung der Methodik EH in einer Anwendung xScraper wurden fünfzig Websites der Datenextraktion und Klassifikation unterzogen. Die Evaluation anhand von verschiedenen Kriterien hat die Wirksamkeit der Methodik bewiesen.

Abstract

A Method for Automated Extraction and Classification of Semi-Structured Product and Address Data from Websites

This thesis offers a new method for automated extraction and classification of data from websites.

The method EH ("Extraction Heuristics") is developed primarily for product and address data; however it can be enhanced to support other domains. The demand for such a method is high because the variety of information on websites is a much promising data source. Currently existing methods and tools for this purpose have certain limits in extracting the contents from websites and their usage offers little benefits. Furthermore the existing solutions do not offer any notable features for classifying the extracted data.

The method EH offers a simple and extendable process, which covers all the tasks while extracting and classifying data on a website. The usage of various heuristic algorithms and ontologies allows automated processing of the majority of tasks. In order to test the new method, a software tool xScraper was developed. This tool was used to extract product and address data from fifty websites. The results of the evaluation have proved the effectiveness of the developed method.

Danksagungen

Diese Diplomarbeit entstand in Zusammenarbeit des Instituts für Parallele und Verteilte Systeme, Abteilung Anwendersoftware der Universität Stuttgart (IPVS), und des Instituts für Arbeitswirtschaft und Organisation IAO der Fraunhofer Gesellschaft (Fraunhofer-Institut IAO).

An dieser Stelle möchte ich mich bei meinen Betreuern Maximilien Kintz und Andrea Horch vom Fraunhofer-Institut IAO für ihre wegweisende Unterstützung herzlich bedanken. Nicht nur die spannende Aufgabestellung, sondern vor allem ihr Engagement beim Thema hat mich höchst motiviert und inspiriert.

Besonderer Dank gilt Herrn Dr. Schwarz vom IPVS, dessen Anmerkungen und Vorschläge jederzeit konstruktiv und sehr hilfreich waren.

Bei Harald Kepschull bedanke ich mich für das Korrekturlesen dieser Arbeit und für seine Fragen, die dazu beigetragen haben, dass viele Stellen in dieser Arbeit klarer geworden sind.

Außerdem bedanke ich mich bei meiner Familie, die mich während des ganzen Studiums unterstützt, ermutigt und motiviert hat.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listingverzeichnis

1 Einleitung

Das Internet ist in moderner Welt zu einer der wichtigsten Informationsquellen geworden. Für Benutzer bringt das kontinuierlich wachsende Informationsangebot entscheidende Vorteile, z.B. beim Preisvergleich, bei der Produktauswahl oder bei der Suche nach einer passenden Dienstleistung. In der Unternehmenswelt spielen diese Vorteile eine noch größere Rolle, außerdem entdeckt die Wirtschaft immer neue Geschäftsmodelle, die auf die Vielfalt der Information auf zahlreichen Websites zurückgreifen.

1.1 Hintergrund

Um die Informationen aus dem Internet effizient zu nutzen, greifen viele Unternehmen zur automatisierten Extraktion von Daten aus den Webseiten. Die Vielfalt in Form und Inhalt der Information ist in diesem Zusammenhang die Ursache vieler Probleme und Herausforderungen: die meisten Websites werden für Menschen erstellt, nicht für Rechner. Ein Mensch interpretiert die visuelle Darstellung einer Webseite, ein Rechner ist dagegen auf den Quellcode angewiesen. Jede Webseite besitzt eine einzigartige Struktur und bietet eine eigene Informationsgestaltung und Gliederung. Hinzu kommt die Tatsache, dass sich zusammengehörende Daten (z.B. lange Listen) über mehrere Webseiten erstrecken können. All dies erschwert die automatisierte Datenextraktion aus den Webseiten.

Es gibt eine Reihe von Softwarelösungen zum Extrahieren von Daten aus Webseiten. Einige Produkte setzen voraus, dass ein Benutzer solide Kenntnisse im Programmieren hat. Andere Produkte sind leicht zu bedienen, können aber nur die einfachsten Aufgaben erledigen, weil sie z.B. keine Eingaben des Benutzers an die Webseite übermitteln können. Gemeinsam haben alle Produkte die Eigenschaft, dass sie sehr detaillierte Angaben des Benutzers benötigen, um die Daten zu extrahieren, und kaum Möglichkeiten für die Datenklassifikation anbieten.

1.2 Aufgabenstellung

Das Ziel dieser Arbeit ist die Entwicklung einer Methodik für die automatisierte Extraktion und Klassifikation von Daten aus Webseiten. Diese Methodik soll zunächst für die Domänen der Produkt- und Adressdaten konzipiert werden, aber auch eine Möglichkeit zur Erweiterung um zusätzliche Domänen bieten.

Die Aufgabe umfasst folgende Arbeitsschritte:

1. State-of-the-Art-Analyse der vorhandenen Methoden, Technologien und Werkzeuge zur Extraktion von Daten aus dem Internet und Prüfung dieser auf Eignung für den Einsatz in der zu entwickelnden Methodik. Betrachtung beste-

hender Klassifikationssysteme und Strategien zur Beschreibung und Klassifikation von Adress- und Produktdaten. Bewertung geeigneter Technologien, Methoden und Klassifikationsstrategien.
2. Entwicklung einer Beschreibungsmethodik zur Systemkonfiguration bzgl. der automatisierten Datenextraktion und Klassifikation, z.B. XML-Beschreibung von Preisdaten.
3. Entwicklung einer Gesamt-Methodik (inkl. einzelner Technologien oder Methoden als Elemente) zur automatisierten Extraktion sowie Klassifikation der Daten. Hierbei sollen auch die Eingabe- und Eingriffsmöglichkeiten für Benutzer von Extraktionswerkzeugen bedacht und Methoden zur Erkennungsoptimierung entwickelt werden.
4. Entwicklung einer Systemarchitektur zur IT- Umsetzung.
5. Prototypische IT-Umsetzung ausgewählter Lösungselemente und Methoden zur automatisierten Klassifikation und Extraktion von Daten aus dem Internet.
6. Verifikation ausgewählter Lösungselemente und damit verbundener Methoden auf Basis von Beispielen.

Die Methodik soll Möglichkeiten zur Nutzung vorhandener Technologien und Werkzeuge einbeziehen (vor allem aus dem Open-Source-Bereich). Teil der Aufgabe ist es, die Methoden und domänenspezifischen Daten für die Extraktion zu beschreiben, die anschließend für eine automatisierte Extraktion und Klassifikation genutzt werden. Ist eine Lösung mit bereits existierenden Technologien und Werkzeugen nicht möglich, soll beschrieben werden, welche Methoden erfolgversprechend und welche Schritte notwendig sind, um die oben beschriebenen Aufgaben zu lösen.

1.3 Lösungsansatz

Diese Arbeit bietet die neue Methodik EH an, die eine automatisierte Extraktion und Klassifikation semistrukturierter Daten aus Webseiten am Beispiel von Produkt- und Adressdaten ermöglicht. Die Abkürzung EH steht für „Extraction Heuristic“. In dieser Methodik werden die zu extrahierenden Daten mithilfe von Ontologien beschrieben, eine Ontologie bildet dabei die Grundlage für die Klassifikation von Daten.

Die Methodik EH beinhaltet verschiedene Verfahren, um auf die Webseite mit den Informationen zu gelangen, die gewünschten Informationen zu extrahieren, die Datenklassifikation anhand einer Ontologie durchzuführen und die Ergebnisse zu speichern.

Um die neue Methodik zu prüfen, wird im Rahmen dieser Arbeit eine Anwendung „xScraper“ auf Basis von EH erstellt. Mit dieser prototypischen Anwendung wird eine Auswahl von Websites getestet.

1.4 Gliederung der Arbeit

Die Grundlagen zur Extraktion und Klassifikation von Daten aus Webseiten, sowie die aktuellen Probleme und die bereits existierenden Verfahren werden in Kapitel 2 be-

schrieben. Danach folgt in Kapitel 3 die Beschreibung der neuen Methodik. Kapitel 4 enthält eine detaillierte Beschreibung der prototypischen Implementierung. Im Anschluss folgen die Ergebnisse von der Evaluation des neuen Verfahrens. Das Kapitel „Zusammenfassung“ stellt die Ergebnisse der Arbeit und einen Ausblick vor.

2 Extraktion und Klassifikation semistrukturierter Daten

Die gezielte Extraktion von Informationen aus Webseiten wird oft und von vielen Unternehmen angewandt. Die Daten auf den Webseiten sind rund um die Uhr verfügbar, werden vielfach aktualisiert und sind häufig unentgeltlich. Viele Informationen liegen bereits in (semi-)strukturierter Form vor: Eine Studie aus dem Jahr 2008 schätzt allein die Anzahl von Tabellen mit relationalen Daten auf Websites im Web auf über 150 Millionen [10]. Das sind einige der Gründe, warum Webseiten eine lukrative und interessante Datenquelle darstellen.

Dieses Kapitel erklärt die Grundlagen von Webseiten und der Extraktion sowie der Klassifikation von Daten. Des Weiteren werden die aktuellen Probleme und die vorhandenen Methoden der Datenextraktion beschrieben.

2.1 Definition von Web Scraping

Die Datenextraktion aus Webseiten ist unter den Namen „Screen Scraping“ bzw. „Web Scraping“ bekannt. Cunningham bietet die folgende Definition für „Screen Scraping“ an [15]:

> *“Screen scraping is the process of deleting the presentational elements of data or text, e.g. as displayed on a Web page, in order to allow further processing of the data.”*

Andere Autoren verstehen unter „Screen Scraping“ nicht nur die Datenextraktion, sondern eine ganze Reihe von Aktivitäten und Techniken, die für die Extraktion von Daten aus Webseiten verwendet werden, so z.B. Stonebraker und Hellerstein [50]:

> *“Commercial screen-scraping is not merely intelligent parsing, however – it also includes the intricacies of navigating JavaScript pages, dealing with cookies and passwords, and interfacing with HTTPS-protected sites.”*

Diese Interpretation ist genauer, weil die Datenextraktion aus Webseiten in der Regel viel mehr Tätigkeiten einschließt, als lediglich die Analyse und die Verarbeitung einer Webseite. Sie ist jedoch unzureichend, weil z.B. so ein wichtiger Aspekt wie die Art der zu extrahierenden Daten kaum berücksichtigt ist.

Wegen dieser Einschränkungen wird Web Scraping im Rahmen dieser Arbeit wie folgt definiert:

> *Web Scraping umfasst alle Tätigkeiten, die für das Auffinden, das Abgrenzen, das Extrahieren, das Klassifizieren, das Umwandeln und das Speichern von Daten aus Webseiten erforderlich sind.*

Im Zusammenhang mit der Extraktion von Daten aus mehreren Websites mit Integration der Information anhand einer Ontologie wird der Begriff „Web Harvesting“ verwendet [19]. Das englische Wort „to harvest“ (Deutsch: „ernten“) impliziert in diesem Fall das Vorgehen: Aus einer großen Menge von Daten werden gezielt die domänenspezifischen Informationen extrahiert und integriert.

Ein weiterer Begriff, der für die Datenextraktion aus Webseiten von Bedeutung ist, lautet „Webcrawler“. Ein Webcrawler ist ein Computerprogramm, das Webseiten automatisch analysiert [27]. Solche Programme werden vor allem von Suchmaschinen eingesetzt.

2.2 Grundlagen von Webseiten

Eine Website ist eine Gesamtheit einzelner Webseiten, die normalerweise miteinander verlinkt sind und die von einer Person, einer Firma, einer Bildungseinrichtung, einer Regierung oder einer Organisation veröffentlicht sind (Def. basiert auf [41]).

Jede Website hat eine einzigartige Adresse, die „Uniform Resource Locator“ (URL). Um eine Website zu öffnen, gibt ein Benutzer diese Adresse im Browser ein. Der Browser baut eine Verbindung zum Webserver auf und lädt die Webseite herunter, anschließend wird die Webseite grafisch auf dem Bildschirm dargestellt. Dabei benutzt der Browser viele Technologien wie z.B. HTML, Scripting, DOM, usw. Im Folgenden werden die wichtigsten dieser Technologien kurz erklärt.

2.2.1 HTML

Für die Darstellung der Informationen auf Webseiten kommt die „Hypertext Markup Language“ (HTML) zum Einsatz [46]. Ein HTML-Dokument verfügt über eine bestimmte baumähnliche Struktur. Alle Texte und Bilder, die auf einer Webseite zu sehen sind, werden mithilfe von Tags beschrieben. Die Tags vermitteln einem Browser, wie die in ihnen enthaltenen Informationen darzustellen sind.

Der Inhalt jeder Webseite muss bestimmte Voraussetzungen erfüllen, damit ein Browser diese Webseite korrekt darstellen kann. So muss ein HTML-Dokument beispielsweise über eine korrekte Syntax verfügen, damit alle HTML-Elemente klar voneinander getrennt werden können und die einzelnen Attribute jedes Elements problemlos erkannt werden können.

Eine typische Webseite enthält neben der Dokumenttypdeklaration ein Tag-Paar „<html>…</html>“. Zwischen diesen beiden Tags befinden sich die Inhalte einer Webseite:

- Zwischen den Tags „<head>…</head>“ werden die Metadaten bereitgestellt. Diese Metadaten enthalten die Information über die Sprache und die Kodierung einer Webseite, die Hinweise für Suchmaschinen, usw.

- Zwischen den Tags „<body>…<body>“ befinden sich letztendlich die Informationen, die auf der Webseite angezeigt werden.

Die Zahl der Tags ist laut der HTML-Spezifikation begrenzt, ein HTML-Dokument kann aber beliebig lang sein [46]. Es gibt Tags zum Darstellen von Textblöcken, Bildern, Tabellen und Auflistungen. Die Tags können verschachtelt werden, so kann eine Tabelle z.B. eine Auflistung mit Bildern enthalten.

2.2.2 Skriptsprachen und Skripte

Skriptsprachen sind eine Gruppe von Programmiersprachen, die kleinere Programmieraufgaben realisieren lassen. Als Skripte werden die Programme bezeichnet, die in Skriptsprachen geschrieben sind.

Für das Web existiert eine Vielzahl von Skriptsprachen, die sich in zwei Kategorien unterteilen lassen:

- Serverseitige Skriptsprachen werden auf dem Webserver eigesetzt. Mit den serverseitigen Skripten können z.B. HTML-Dokumente oder Bilder generiert werden. Bekannte Beispiele von solchen Sprachen sind Perl [52] und PHP [4].
- Clientseitige Skriptsprachen werden im Browser benutzt, sie ermöglichen die Interaktion mit dem Benutzer und sie können Teile des HTML-Dokuments unmittelbar im Browser verändern. Die am meisten verbreitete Skriptsprache dieser Kategorie ist JavaScript [54].

2.2.3 DOM

Viele Webseiten sind dynamisch, d.h. sie können sich nach dem Öffnen im Browser verändern. Dies wird durch die Verwendung des „Document Object Model“ (DOM) und der Skripte möglich. Das Document Object Model ist eine Schnittstelle, die den dynamischen Zugriff und die Veränderung der Inhalte, der Struktur und der Styles eines Dokuments erlaubt [25]. Ein Skript, z.B. JavaScript, kann auf die Elemente des DOM zugreifen und, ausgelöst durch Benutzerhandlungen, die Inhalte einer Webseite verändern.

Das DOM stellt dem Webseitenentwickler eine Reihe von Objekten zur Verfügung, über die sich die Webseite und sogar (teilweise) der Browser steuern lassen. Jedes Objekt hat mehrere Eigenschaften, auf die ein Skript zugreifen kann. Abbildung 1 zeigt die Eigenschaften eines HTML-Knotens „p“.

Abbildung 1. Auszug aus der Liste von Eigenschaften eines Tags in DOM, angezeigt mit Firebug[1]

2.2.4 Ajax

Es gibt kaum eine andere Technologie, die das Web in den letzten Jahren mehr geprägt hat, als „Asynchronous JavaScript and XML“ (Ajax). Mit Ajax sind Websites, und vor allem die Webanwendungen, hinsichtlich ihrer Interaktionsmöglichkeiten den Desktopanwendungen ähnlicher geworden. Wie der Name sagt, ermöglicht Ajax den asynchronen Datenaustausch zwischen dem Browser und dem Webserver, und zwar im XML-Format [18]. Hierfür wird eine spezielle Komponente verwendet – die Ajax-Engine. Eine Webseite kann mithilfe von JavaScript auf diese Komponente zugreifen und asynchrone Abfragen ausführen, was bei der klassischen Interaktion mit Webseiten unmöglich ist (vgl. Abbildung 2).

Die Interaktion mit einer Webseite ohne Ajax funktioniert nach dem Request-Response-Paradigma: Der Browser führt eine Abfrage aus und wartet auf die Antwort, wie im oberen Teil der Abbildung 2 angezeigt. Während der Wartezeit kann keine Interaktion mit der Webseite erfolgen. Sobald die Antwort des Servers vollständig angekommen ist, wird der Inhalt der zuvor im Browser angezeigten Webseite komplett ersetzt.

Wenn Ajax eingesetzt wird, übernimmt die Ajax-Engine den Datenaustausch zwischen dem Browser und dem Webserver (siehe den unteren Teil der Abbildung 2). Die Abfragen erfolgen asynchron und transparent für den Benutzer. Durch die Verwendung von DOM und Scripting ist es möglich, nach einer Abfrage nicht die ganze Webseite, sondern nur einen Teilbereich davon zu aktualisieren. Diese Eigenschaft von Ajax reduziert das Datenvolumen beim Datenaustausch und beschleunigt die Aktualisierung der Webseite.

Für den Benutzer bietet Ajax eine Verbesserung im Vergleich zur traditionellen Interaktion mit den Webseiten: kürzere Wartezeiten und breitere Interaktionsmöglichkeiten.

[1] Firebug ist eine Erweiterung für den Browser Mozilla Firefox, die das Monitoring von Webseiten ermöglicht (https://addons.mozilla.org/de/firefox/addon/firebug/)

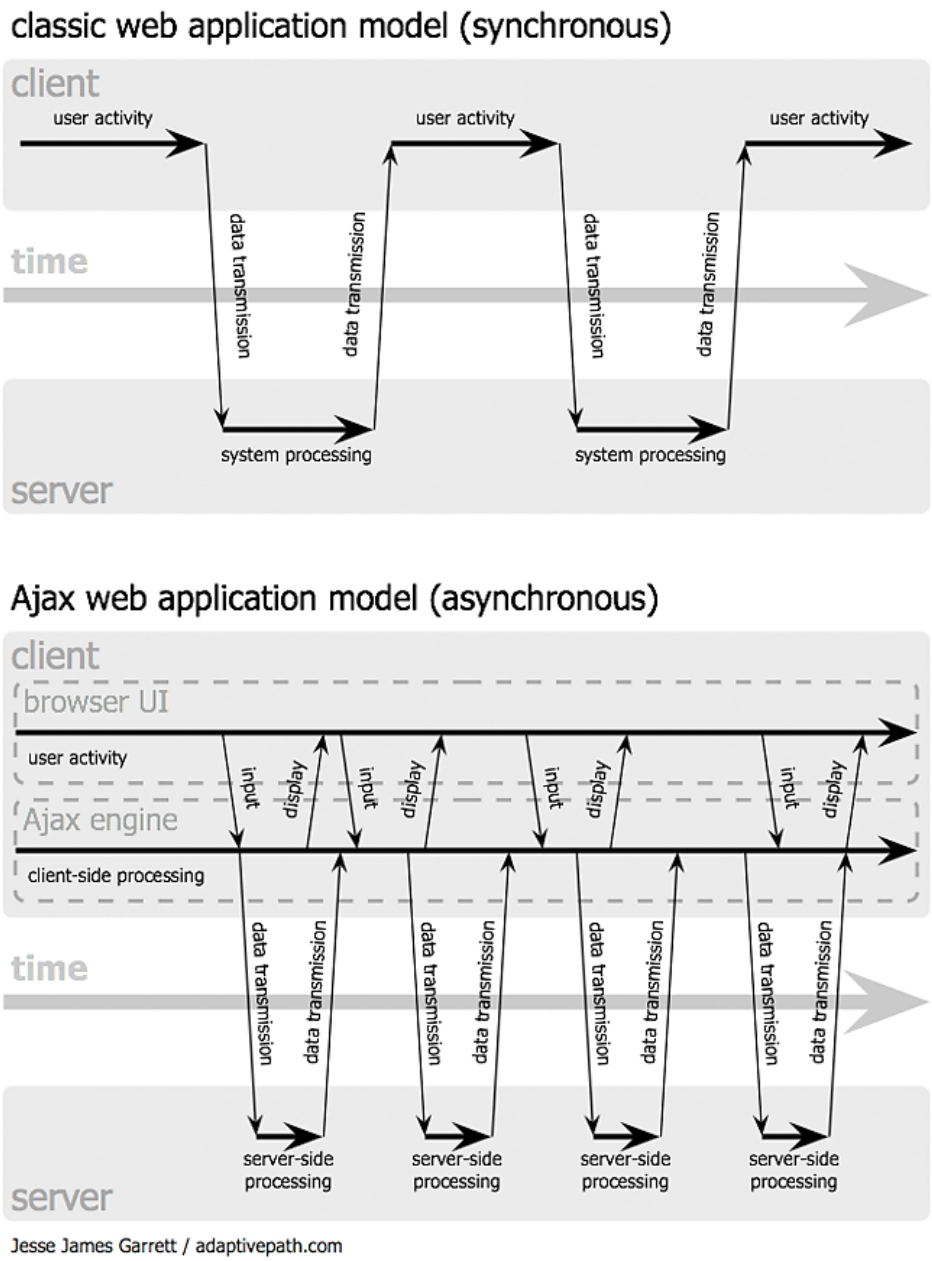


Abbildung 2. Zeitlicher Ablauf der klassischen Interaktion mit Webseiten und der Interaktion mit Ajax [18]

Bei der Datenextraktion ist die Verwendung von Ajax eher nachteilig, die Einzelheiten dafür werden in Abschnitt 2.5 erläutert.

2.3 Datenextraktion

Es gibt zahlreiche Anwendungsfälle, in welchen die Extraktion von Daten aus Webseiten eingesetzt wird, z.B.:

- Ermittlung der aktuellen Preise eines Produkts oder einer Dienstleistung bei verschiedenen Anbietern (Produktvergleich, Anbietervergleich),
- Erkennung neuer Veröffentlichungen auf einer Webseite (Webmonitoring),
- Regelmäßige Abfrage bestimmter Daten (z.B. Wetterbeobachtung),
- Verfolgung der Veränderungen auf Webseiten.

Ohne Automatisierung ist der Prozess der Datenextraktion sehr aufwendig. Wenn alle Aufgaben manuell ausgeführt werden, muss der Benutzer sämtliche benötigte Informationen manuell aus dem Webbrowser in sein Dokument übertragen. Je öfter die Daten abgefragt werden, desto häufiger muss der Vorgang wiederholt werden. Bei einem großen Datenvolumen kann die Datenextraktion eine entsprechend längere Zeit in Anspruch nehmen.

Ein weiteres Problem ist die monotone Weise einer manuellen Datenextraktion. Da eine solche Tätigkeit sehr eintönig ist, sind Fehler unvermeidbar. Um sicherzustellen, dass die extrahierten Daten fehlerfrei sind, müssen die Daten gegen die Quelle abgeglichen werden, was zu einer Steigerung des Aufwands führt.

Diese Probleme lassen sich durch die automatisierte Datenextraktion lösen. Dabei übernimmt ein Computerprogramm das Aufrufen einer Webseite, das gezielte Auslesen der Informationen und das Speichern der gewonnenen Daten in einem passenden Format.

2.3.1 Wrapper

Die automatisierte Datenextraktion basiert auf „Wrapper“. Kushmerick definiert einen Wrapper als „eine Prozedur zum Extrahieren von Inhalten einer bestimmten Ressource“ [29].

Das älteste Verfahren für die automatisierte Datenextraktion aus Webseiten ist das manuelle Erstellen von „Wrappern“. Zunächst untersucht ein Programmierer die Struktur einer Webseite und notiert die Elemente des DOM, die die gewünschten Informationen enthalten. Anhand dieser Elemente schreibt er dann einen Wrapper. Beim Ausführen extrahiert ein Wrapper die Inhalte der notierten Elemente.

Der Vorteil dieser Methode besteht darin, dass sie auf alle Webseiten angewendet werden kann und gute Ergebnisse liefert. Es gibt aber auch eine ganze Reihe von Nachteilen. Der größte Nachteil ist der hohe Aufwand, der zum Erstellen eines Wrappers benötigt wird. Da jede Webseite einen eigenen Wrapper braucht, kann man mit akzeptablem Aufwand lediglich eine begrenzte Zahl der Webseiten einbeziehen. Des Weiteren wird die Struktur der Webseiten nicht selten geändert, z.B. beim Redesign einer Website, was einen Wrapper ungültig machen kann, weil die Elemente, die ein Wrapper ausliest, anders geordnet oder gar entfernt werden können.

Manuell erstelle Wrapper sind also nur dann sinnvoll, wenn ein Benutzer Informationen aus einer konkreten Webseite braucht, deren Struktur sich selten ändert.

2.3.2 Wrapper Induction

Die Weiterentwicklung der Datenextraktion mit Wrappern führt logischerweise zum Versuch, Wrapper automatisch erstellen zu lassen: Diese Methode nennt sich „Wrapper Induction“ [30]. Es gibt mehrere Verfahren der Datenextraktion mithilfe von

Wrapper Induction. Die wichtigsten von ihnen werden im Folgenden vorgestellt. Die Klassifikation der Verfahren stammt von Laender [32], der die Verfahren in verschiedene Gruppen eingeordnet hat. Als Hauptkriterium für die Einordnung gilt die Art und Weise wie ein Wrapper erstellt wird. Demnach gibt es folgende Gruppen verschiedener Verfahren:

1. *Sprachen für die Entwicklung von Wrappern*

Diese Gruppe enthält Verfahren mit speziellen Sprachen, die für die Datenextraktion entwickelt wurden. Solche Sprachen stellen eine Alternative zu den weit verbreiteten Programmiersprachen wie Perl oder Java dar. Sie bieten spezielle Konzepte, die das Erstellen von Wrappern erleichtern. Eine Reihe von Tools bieten solche Sprachen an: Minerva [13] kombiniert beispielsweise die Verwendung von deklarativen Grammatiken und prozeduralen Programmiersprachen. Web-OQL [3] ist eine deklarative Abfragesprache, die auf HTML angewendet werden kann. Jedi [24] verwendet Attributgrammatik, um Wrapper zu erstellen.

2. *Tools, die HTML benutzen*

Eine Reihe von Tools nutzen die strukturellen Merkmale von HTML für die Erstellung von Wrappern. Zunächst wird aus dem HTML-Dokument eine baumbasierte Datenstruktur mit allen Tags erstellt. Im nächsten Schritt erfolgt das (semi-)automatische Erstellen von Extraktionsregeln. Diese Regeln werden dann auf die Webseiten zwecks Datenextraktion angewendet.

W4F ist ein Toolkit zum Erstellen von Wrappern [48]. Dieses Tool unterteilt den Prozess des Erstellens von Wrappern in drei Teile: die Beschreibung der Navigation zur Webseite, die Beschreibung der Datenbereiche für die Extraktion und die Beschreibung der Datenstrukturen zum Speichern der extrahierten Informationen

Eine weitere Anwendung, XWRAP [36], führt den Benutzer durch mehrere Schritte, um einen Wrapper zu erstellen. Dem Benutzer stehen sechs Heuristiken für die automatische Erkennung der Datenbereiche auf der Webseite zur Verfügung. Am Ende generiert XWRAP einen Wrapper in Java.

RoadRunner [14] vergleicht zwei oder mehr Webseiten der gleichen Klasse[2] und sucht nach Ähnlichkeiten in ihrer Struktur. Anhand der Ergebnisse versucht RoadRunner ein Datenschema für den Inhalt von Webseiten zu erstellen. Der ganze Prozess verläuft völlig automatisch.

[2] Bei den Webseiten der „gleichen Klasse“ handelt es sich um Webseiten mit ähnlichen strukturellen Merkmalen, wie es z.B. bei Webseiten, die nach einer Vorlage erstellt sind, der Fall ist.

3. NLP-basierte Tools

Verarbeitung der natürlichen Sprache (engl. Natural Language Processing, NLP) wird eingesetzt, um die Webseiten mit textuellem Inhalt zu verarbeiten (wie z.B. ein Stellenangebot). Dabei finden verschiedene linguistische Mittel Anwendung, um die Objekte und die Beziehungen im Text zu erkennen. Diese Kategorie besteht aus den Tools wie RAPIER [11], SRV [17] und WHISK [49].

4. Wrapper Induction Tools

Wrapper Induction Tools erstellen Extraktionsregeln anhand von trainierten Beispieldaten. Bei diesen Verfahren spielt das „Labeling“ eine zentrale Rolle: ein Benutzer markiert die zu extrahierenden Informationen auf einer Webseite und ein Algorithmus versucht anhand dieser Angaben einen Wrapper zu generieren. Dabei werden die strukturellen Merkmale von Webseiten analysiert. Zu den Wrapper Induction Tools gehören folgende Programme: WIEN [30], SoftMealy [23] und STALKER [42].

5. Modeling-basierte Tools

Mit Modeling-basierten Tools kann die Struktur der Zieldaten definiert werden. Ein Tool versucht dann die zu dieser Struktur passenden Informationen aus einer Webseite zu extrahieren. Beispiele für solche Tools sind NoDoSE [1] und DEByE [31].

6. Ontologie-basierte Tools

Die Verwendung von Ontologien bietet eine neue Sicht auf die Datenextraktion. Ontologien sind formal beschriebene Darstellungen von Begrifflichkeiten und Beziehungen zwischen diesen Begrifflichkeiten in einer bestimmten Domäne (für eine genauere Beschreibung von Ontologien siehe Abschnitt 2.4.1).

Eine Ontologie macht es möglich, bei der Datenextraktion nicht auf der strukturellen, sondern auf der inhaltlichen Ebene zu operieren. Der Vorteil dabei ist, dass die Struktur einer konkreten Webseite nicht mehr im Vordergrund steht. Ein Algorithmus versucht zusammenhängende Daten anhand einer Ontologie zu Extrahieren.

Als Beispiel für diese Gruppe kommt die Anwendung, die von Embley et al. [16] entwickelt wurde. Sie nutzt eine zuvor erstellte Ontologie, um die Daten aus einer Webseite zu extrahieren und zu speichern. Für die Datenextraktion wird auch NLP verwendet.

2.4 Datenklassifikation

Klassifikationen sind „hierarchische Strukturen, die zum Zweck der Organisation größerer Mengen von Objekten verwendet werden“ [20].

Im Kontext der Datenextraktion wird die Datenklassifikation benutzt, um die einzelnen Informationen aus den Webseiten anhand ihrer inhaltlichen und strukturellen Merkmale den vorher definierten oder den vorher unbekannten Klassen zuzuordnen. Hier-

für existieren mehrere Verfahren wie Entscheidungsbäume, neuronale Netze, Bayes-Klassifikatoren, usw. [2]. Die Auswahl eines passenden Verfahrens hängt davon ab, welche Art von Daten klassifiziert werden müssen und ob die Klassifizierung überwacht oder unüberwacht erfolgt.

In dieser Arbeit liegt der Fokus auf der Extraktion von Produkt- und Adressdaten, was einen gewissen Rahmen festlegt, wie die zu extrahierenden Daten aussehen können. Die Eigenschaften der Gruppen, in die die extrahierten Daten eingeordnet werden können, sind somit bekannt. Anhand von Merkmalen der extrahierten Daten kann eine Zuordnung in die am besten passende Gruppe erfolgen. Für die Lösung dieser Aufgabe eignet sich aufgrund seiner schnellen Berechenbarkeit der Naive Bayes-Klassifikator [39] am besten.

Für die Beschreibung der zu klassifizierenden Daten gibt es verschiedene Möglichkeiten, z.B. das XML Schema [51], das Entity-Relationship-Modell [45] oder die Ontologien. XML Schemata (XSD) ermöglichen die Beschreibung von atomaren und komplexen Datentypen. Entity-Relationship-Modelle (ERM) können semantische Beziehungen darstellen. Ontologien bieten jedoch mehr Flexibilität bei der Umsetzung von komplexen semantischen Beziehungen, und können die Konzepte von XSD und ERM realisieren.

2.4.1 Ontologien

Ontologien spielen eine wichtige Rolle im Zeitalter des Semantischen Webs. Berners-Lee schreibt dazu Folgendes [8]:

> *"Ontologies can enhance the functioning of the Web in many ways...With ontology pages on the Web, solutions to terminology (and other) problems begin to emerge. The meaning of terms or XML codes used on a Web page can be defined by pointers from the page to an ontology."*

Es ist also möglich Ontologien einzusetzen, um die Informationen auf den Webseiten mit zusätzlichen semantischen Merkmalen zu ergänzen.

Gruber bietet die folgende Definition von Ontologien an [21]:

> *"An ontology is an explicit specification of a conceptualization."*

Wenn ein Benutzer z.B. Autopreise aus einer Webseite extrahieren möchte, kann eine Ontologie beschreiben, was für Eigenschaften (Modell, Farbe, Herstellungsdatum) ein Auto haben kann und wie ein Preis aussieht (Währung, numerisches Format, usw.). Mit diesen Angaben kann ein Algorithmus die Daten auf der Webseite analysieren und die Informationen zu einzelnen Autos korrekt interpretieren.

Ontologien operieren mit „Begriffen" (im Englischen: „concepts") um die Objekte der realen Welt abzubilden. Die Begriffe können weitere übergeordnete und untergeord-

nete Begriffe besitzen. Objekttypen werden in Ontologien durch die „Typen“ repräsentiert. „Instanzen“ ermöglichen das Erstellen von individuellen Objekten eines bestimmten Typs. Mit „Relationen“ können Verbindungen zwischen den Objekten hergestellt werden.

Ontologien werden in zwei Gruppen unterteilt:

- Lightweight-Ontologien beschreiben Begriffe, Taxonomien (gemeint sich Klassifikationen von „Begriffen“) und Beziehungen („Relationen“ zwischen den Objekten)
- Heavyweight Ontologien erweitern die Lightweight-Ontologien um Axiome (wahre Aussagen innerhalb einer Ontologie) und Einschränkungen.

Für die Beschreibung der Ontologien existieren verschiedene Notationen, z.B. das RDF-Schema [9], DAML+OIL [22] und OWL [43].

Im Fokus dieser Arbeit werden Ontologien eingesetzt, um die Produkt- und die Adressdaten zu beschreiben.

2.4.2 Produktdaten

Der Begriff „Produktdaten“ wird wie folgt definiert [37]:

> *“Als Produktinformationen werden alle Produktbeschreibungen in unterschiedlichen Sprachen, Preise und Rabatte, technische Attribute und vor allem Produktbeziehungen bezeichnet.“*

Wie aus dieser Definition folgt, enthalten die Produktdaten zwei Typen der Informationen:

- Die allgemeine Beschreibung, die Eigenschaften sowie die Preise.
- Die Beziehungen zu anderen Produkten und Produktgruppen.

Im Kontext unserer Aufgabe betrachten wir nur den ersten Aspekt von Produktdaten. Die Berücksichtigung von Beziehungen zwischen den Produkten bei der Datenextraktion ist ein anspruchsvolles Thema, das jedoch nicht zur Aufgabenstellung gehört und deshalb nicht verfolgt werden kann.

2.4.3 Adressdaten

Adressdaten sind verschiedene Zielangaben bezüglich einer Person oder einer Organisation. Diese Daten enthalten die Bezeichnung (den Namen) einer Person oder einer Organisation und eine oder mehrere der folgenden Angaben:

- Anschrift,
- Telefonnummer, Faxnummer,
- E-Mail-Adresse,
- Website-Adresse,

- Geografische Koordinaten, usw.

Einige dieser Angaben haben eine fest definierte Struktur. Eine E-Mail-Adresse wird beispielsweise überall auf der Welt gleich geschrieben.

Andere Angaben wie Anschrift können je nach Land verschiedene Bestandteile oder eine andere Reihenfolge der Bestandteile besitzen.

2.5 Probleme bei der Extraktion und Klassifikation von Daten

Die Vielfalt der Webseiten in Form und Inhalt sowie die vielen Technologien im Web führen zu Schwierigkeiten bei der Extraktion und Klassifikation von Daten aus Webseiten. Im Folgenden werden die aktuellen Probleme beschrieben.

2.5.1 Veränderungen in der Struktur von Webseiten

Falls sich die Struktur einer Webseite ändert, nachdem ein Wrapper erstellt wurde, kann der Wrapper u.U. nicht mehr korrekt funktionieren.

2.5.1.1 Aktualisierung von Webseiten

Im Verlauf der Zeit werden die meisten Webseiten aktualisiert. Dabei kann es sich um inhaltliche und/oder strukturelle Veränderungen handeln. Solche Veränderungen können den Wrapper unbrauchbar machen, falls die Elemente, die als Orientierung für den Wrapper dienen, nicht mehr vorhanden sind.

2.5.1.2 Verwendung von Scripting

Wie im Abschnitt 2.2.3 bereits erwähnt, kann sich die Struktur einer Webseite im Browser unter dem Einfluss von Skripten verändern. Für den Benutzer können solche Veränderungen absolut unbemerkt bleiben, weil nicht jede Veränderung im Browser sichtbar ist.

Die Gefahr solcher Veränderungen besteht darin, dass der Wrapper sein Ziel verfehlt und auf die gewünschten Informationen nicht zugreifen kann (ähnlich wie in Abschnitt 2.5.1.1).

Es ist zwar möglich, die Nutzung von Skripten zu deaktivieren, bei den meisten Websites führt das jedoch dazu, dass die Interaktionsmöglichkeiten dann nur in einem sehr eingeschränkten Umfang verfügbar sind.

2.5.2 Verwendung von Ajax

Die Verwendung von Ajax ist vorteilhaft für Benutzer, bei der Extraktion von Daten aus einer Webseite ist sie jedoch eher nachteilig. Ohne Ajax bleiben die Interaktionsmöglichkeiten ziemlich übersichtlich (Buttons anklicken, Text in ein Textfeld eintippen, usw.). Mit Ajax können komplexe Steuerelemente wie z.B. ein Kalender erstellt werden, die die Interaktionsmöglichkeiten drastisch erweitern. Die Verwendung solcher Steuerelemente erschwert jedoch die automatisierte Navigation zu der Zielwebseite.

Ein weiteres Problem, das mit der Benutzung von Ajax erscheint, ist die dynamische Veränderung der Inhalte von Webseiten, die schwer zu verfolgen und zu erkennen ist. Durch das asynchrone Nachladen von XML-Daten können beliebige Teile eines HTML-Dokuments entfernt oder erweitert werden.

2.5.3 Grafische Darstellung von Daten

Manche Anbieter schützen die Daten ihrer Webseite vor maschineller Extraktion, indem sie diese Daten in grafischer statt in textueller Form veröffentlichen. Dies wird zunehmend bei den Produktpreisen beobachtet. Für die Besucher von Webseiten stellt eine grafische Darstellung von Daten meist kein Problem dar, bei der automatisierten Datenextraktion führt sie jedoch zu Schwierigkeiten: Die Rechner sind auf spezielle Algorithmen angewiesen, um Texte in Bildern zu erkennen.

2.5.4 Daten in anderen Formaten

Die Verwendung von Daten in anderen Formaten ist ein häufiges Problem im Web. Viele Webseiten setzen Technologien wie Adobe Flash oder Microsoft Silverlight ein. Für den Benutzer bieten solche Technologien viele Vorteile durch eine verbesserte Interaktion. Aus technischer Sicht sind diese Erweiterungen eigenständige Anwendungen, die im Browser ausgeführt werden. Die Informationen, die sie im Browser anzeigen, sind in diesen Anwendungen integriert und stammen nicht aus dem HTML-Dokument. Somit hat ein Wrapper keinen Zugriff auf diese Daten.

2.5.5 Deep Web

Das Web kann in zwei ungleiche Teile differenziert werden: „Surface Web“ und „Deep Web“ [7]. Das Surface Web besteht aus den Webseiten, die durch Suchmaschinen auffindbar sind. Das Deep Web besteht aus den Webseiten, auf die kein Zugriff erfolgen kann. Dafür gibt es mehrere Gründe: Die Webseiten können sich z.B. in einem passwortgeschütztem Bereich einer Website befinden, oder sie können Inhalte darstellen, die nur durch das Ausfüllen eines Formulars dynamisch erstellt werden. Im Jahr 2001 schätzte Bergman das Volumen des Deep Web 400 bis 550-fach größer als das Surface Web [7].

Im Kontext der Datenextraktion aus Webseiten hebt diese Relation die Rolle der automatisierten Navigation über eine Website hervor (gemeint ist das Ausfüllen von Formularen, das Anklicken von Buttons, usw.).

2.5.6 Extraktion semistrukturierter Daten

Daten können strukturiert, semistrukturiert oder unstrukturiert sein. Chang et al. [12] bieten die folgende Differenzierung an:

Art von Daten	Beispiel
Strukturierte Daten	Relationale Datenbanken, XML
Semistrukturierte Daten	HTML
Unstrukturierte Daten	Freitext

Tabelle 1. Klassifikation von Daten nach Chang et al.

Die maschinelle Datenverarbeitung funktioniert im Allgemeinen besser mit strukturierten Daten als mit unstrukturierten Daten. Dasselbe gilt für die Datenextraktion aus den Webseiten. Die aus HTML bestehenden Webseiten verfügen zwar über eine baumähnliche Struktur (siehe Abschnitt 2.2.1), diese Struktur kann aber fehlerhaft sein. So werden z.B. oft die abschließenden Tags wie „
“ oder „ “ weggelassen, für andere Tags wie „
“ ist ein abschließender Tag gar nicht notwendig. Die Browser sind in Bezug auf die fehlenden abschließenden Tags fehlertolerant, bei der Datenextraktion könnte ein fehlender Tag jedoch zu Problemen führen: Der Wrapper wird das Ende eines Datensatzes nicht erkennen.

Um dieses Problem zu lösen, wurde die „Extensible Hypertext Markup Language“ (XHTML) entwickelt [44]. XHTML definiert eine strengere, auf XML basierende Struktur, die das Fehlen von abschließenden Tags verbietet.

Rogers untersuchte im Januar 2011 ca. 360.000 Webseiten auf 70.000 Websites. Dabei hat er festgestellt, dass etwa 12% der Webseiten HTML verwenden und etwa 62% XHTML nutzen [47]. Diese Ergebnisse zeigen, dass HTML immer noch auf einer Vielzahl von Websites eingesetzt wird.

Ein weiteres Problem, dass sowohl HTML als auch XHTML in gleichem Maß betrifft, ist die zweckfremde Verwendung bestimmter Tags. Für die Darstellung tabellarischer Daten existieren z.B. die Tags „<table>…</table>“. Diese Tags werden aber auf vielen Webseiten zum Zweck der Gestaltung eingesetzt (z.B. um eine Webseite in mehrere Bereiche aufzuteilen).

Andererseits kann eine Tabelle ohne die Tags „<table>…</table>“ definiert werden, z.B. mit wiederholten Tags „ … „.Ein Mensch wird in diesem Fall die tabellarische Struktur optisch erkennen, für die Maschine stellt dieser Fall lediglich eine Sequenz der Textblöcke dar.

2.5.7 Zugriffsbeschränkungen

Der Betreiber eines Webservers hat mehrere Möglichkeiten, den Zugriff zu bestimmten Inhalten auf seinem Webserver zu begrenzen.

2.5.7.1 Zugriffsperre wegen zu vieler Abfragen

Jeder Zugriff auf eine Webseite wird auf dem Webserver protokolliert. Fällt dem Betreiber des Webservers auf, dass von einem bestimmten Rechner zu viele Abfragen kommen, so kann er diesen Rechner auf die „schwarze Liste“ setzen, wodurch der Zugriff für diesen Rechner gesperrt wird.

Um dieses Problem zu umgehen, können Proxy-Server verwendet werden.

2.5.7.2 Authentifizierung eines menschlichen Benutzers

Wenn der Betreiber das automatisierte Auslesen von Informationen aus seiner Website verhindern will, kann er den Zugriff zu den Webseiten mit dem Challenge-Response-

Verfahren schützen. Dabei muss der Benutzer eine bestimmte Frage richtig beantworten [40].

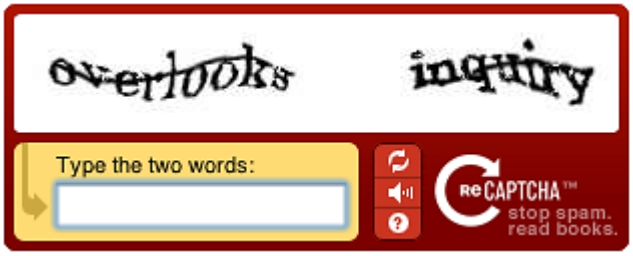


Abbildung 3. Beispiel des Challenge-Response-Verfahrens: CAPTCHA[3]

Dieses Verfahren basiert auf der Feststellung, dass es Aufgaben gibt, die ein Mensch sehr schnell lösen kann, aber die für eine Maschine nur schwer lösbar sind. Im Beispiel auf der Abbildung 3 müssen verzerrte Wörter richtig erkannt und in das Eingabefeld eingetippt werden.

2.5.7.3 Authentifizierung mit einem Passwort

Viele Websites enthalten passwortgeschützte Bereiche. Wenn die Inhalte aus diesen Bereichen extrahiert werden, muss sich der Benutzer zuvor auf dieser Website registrieren.

Die Datenextraktion kann in diesem Fall nur eingeschränkt möglich sein, weil viele Anbieter z.B. die parallelen Zugriffe eines angemeldeten Benutzers nicht zulassen.

2.5.7.4 Zugriffsbeschränkung mit robots.txt

Laut dem Robots-Exclusion-Standard [28] kann der Betreiber einer Website eine Liste von Regeln für Webcrawler festlegen. Diese Liste befindet sich in der Datei „robots.txt“ im Stammverzeichnis einer Website. Mit ihrer Hilfe kann der Betreiber bestimmen, welche Bereiche seiner Website ein Crawler auslesen darf, und welche nicht. Diese Regeln haben jedoch einen lediglich hinweisenden Charakter.

2.5.8 Rechtliche Aspekte der Datenextraktion aus Webseiten

Die Datenextraktion aus Webseiten an sich ist aus rechtlicher Sicht meist zulässig (zumindest wenn sie nicht in den allgemeinen Geschäftsbedingungen einer Website ausdrücklich verboten ist), es gibt aber Einschränkungen in der Art und Weise, wie die extrahierten Daten verwendet werden dürfen[4].Es müssen auf jeden Fall die Urheberrechte beachtet werden.

Die rechtliche Situation in unterschiedlichen Ländern variiert sehr stark: Was in den USA verboten ist, kann beispielsweise in Deutschland durchaus zulässig sein. Deshalb ist es wichtig zu beachten, in welchem Land eine Website registriert ist.

[3] http://www.captcha.net/
[4] http://www.scrapingweb.com/legal-issues.html

Manche Anbieter, die auf ihrer Webseite Informationen aus einer Datenbank veröffentlichen, sehen die Datenextraktion als einen Angriff auf ihre Datenbestände [26].

In dieser Arbeit wurde lediglich die technische Seite der Datenextraktion aus Webseiten untersucht, die rechtlichen Aspekte wurden nicht betrachtet.

2.6 Tools für die Datenextraktion aus den Webseiten

Im Abschnitt 2.3.2 wurde bereits eine Auswahl von Techniken für die Datenextraktion im Zusammenhang mit Wrappern bereits vorgestellt. Die betrachteten Tools stammen jedoch aus dem Bereich der Forschung und sind meist nicht für die Endnutzer geeignet. Es gibt aber mehrere Produkte, die die Datenextraktion für die Endbenutzer ermöglichen. Dieses Kapitel bietet einen Überblick über die aktuell vorhandenen Tools.

2.6.1 Web-Harvest

Web-Harvest[5] ist eine Open-Source-Software für die Datenextraktion aus dem Web. Dieses Tool ermöglicht die Verarbeitung von XML- und HTML-Dokumenten mit XPath, XQuery und regulären Ausdrücken (siehe Abbildung 4).

Mit Web-Harvest kann der Benutzer eine Konfiguration erstellen, die die Adresse der Zielwebseite sowie die Angaben zu den zu extrahierenden Informationen enthält.

Die Datenextraktion erfolgt in zwei Schritten. Im ersten Schritt werden eine oder mehrere Webseiten abgerufen und die Inhalte ausgelesen. Dann erfolgt die Transformation von HTML in XML. Anschließend wird eine interne Liste der mit XPath extrahierten XML-Knoten erstellt.

Im zweiten Schritt wird diese Liste verarbeitet. Hierfür erstellt der Benutzer eine Art Skript, das auf XML basiert. Mit XQuery können die extrahierten Daten bereinigt und in eine XML-Datei mit vordefinierter Struktur gespeichert werden.

Die Vorteile von Web-Harvest sind seine Erweiterbarkeit, eine klare Struktur der Konfiguration für die Datenextraktion und eine gute Dokumentation. Die Möglichkeit, sowohl Text als auch Bilder aus den Webseiten zu extrahieren, trägt auch zu den Vorteilen dieser Software bei.

[5] http://web-harvest.sourceforge.net/

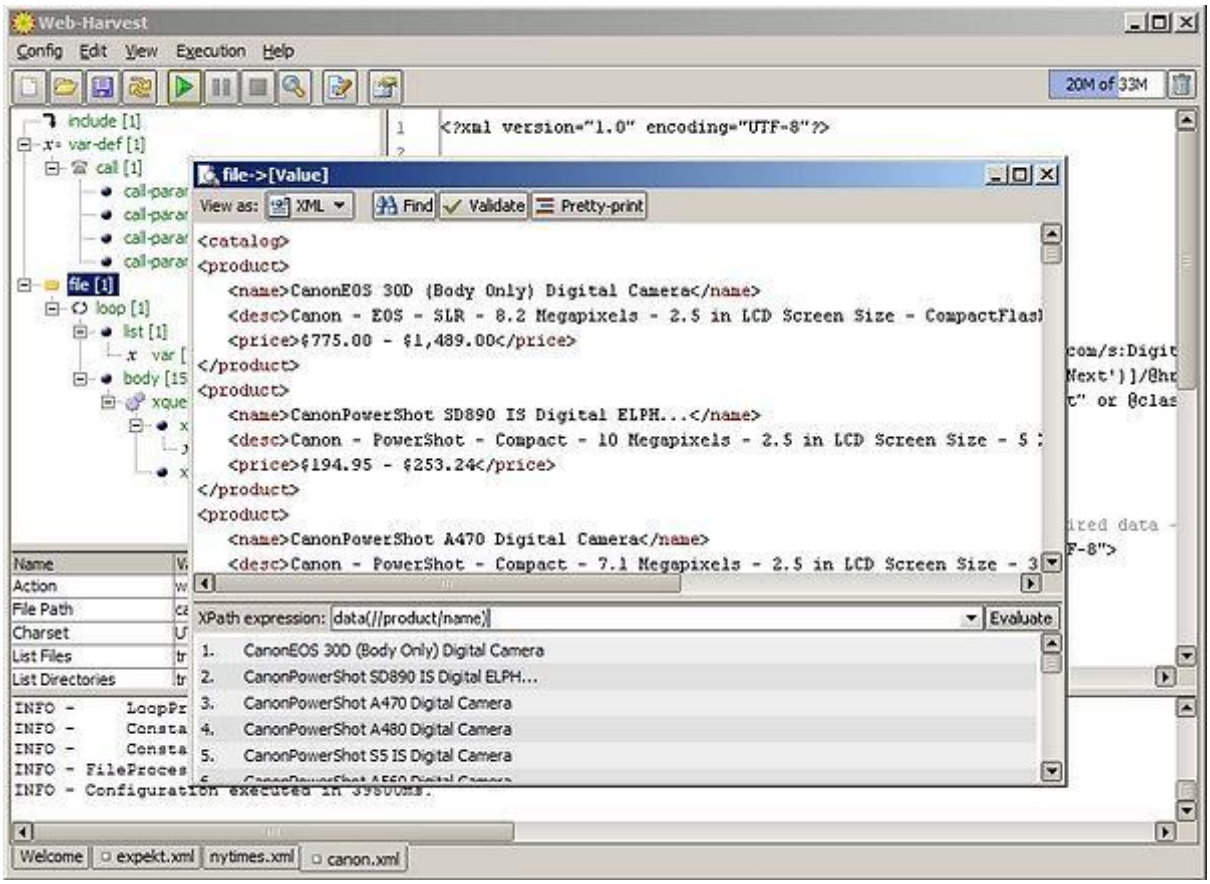


Abbildung 4. Auswahl von Knoten mit den Daten anhand eines XPath-Ausdrucks in Web-Harvest[6]

Andererseits gibt es eine Reihe von Nachteilen:

- Die Konfiguration muss in einem Editor manuell geschrieben werden.
- Das Ausfüllen von Formularen sowie das Anklicken von Buttons sind nicht vorgesehen, dem Benutzer steht lediglich die Möglichkeit der parametrisierten GET- und POST-Abfragen zur Verfügung.
- Ajax wird nicht unterstützt.
- Die Datenklassifikation ist nicht vorhanden, der Benutzer kann lediglich die Struktur der XML-Datei für die Ausgabe der extrahierten Daten definieren.
- Das Erstellen von Skripten setzt Programmierkenntnisse voraus.

2.6.2 Mozenda

Mozenda[7] ist ein kommerzielles Produkt, das das Extrahieren von Daten aus dem Web als SaaS[8] anbietet.

Um Informationen aus Webseiten zu extrahieren, erstellt der Benutzer einen „Agenten“. Dieser Agent enthält die Adresse der Eingangsseite sowie alle Aktionen, die für

[6] http://web-harvest.sourceforge.net/screenshots.php
[7] http://www.mozenda.com/default
[8] „Software as a Service“ (Saas) ist ein Modell der Softwarenutzung, das eine nutzungsabhängige Bezahlung der Software anbietet.

die Extraktion von Daten notwendig sind: Das Anklicken von Buttons, das Ausfüllen von Formularen, usw. Der Benutzer erstellt den Agenten interaktiv, indem er im eingebauten Browser die einzelnen Elemente einer Webseite anklickt und die gewünschte Handlung auswählt (siehe Abbildung 5). Die extrahierten Daten können anschließend in einer Reihe von Dateiformaten gespeichert werden.

Zu den Vorteilen von Mozenda zählen vor allem die intuitive Benutzeroberfläche sowie die Möglichkeit der interaktiven Erstellung eines Agenten. Einmal erstellt, kann der Agent zu gewünschten Tageszeiten automatisch gestartet werden, um die Datenextraktion auszuführen.

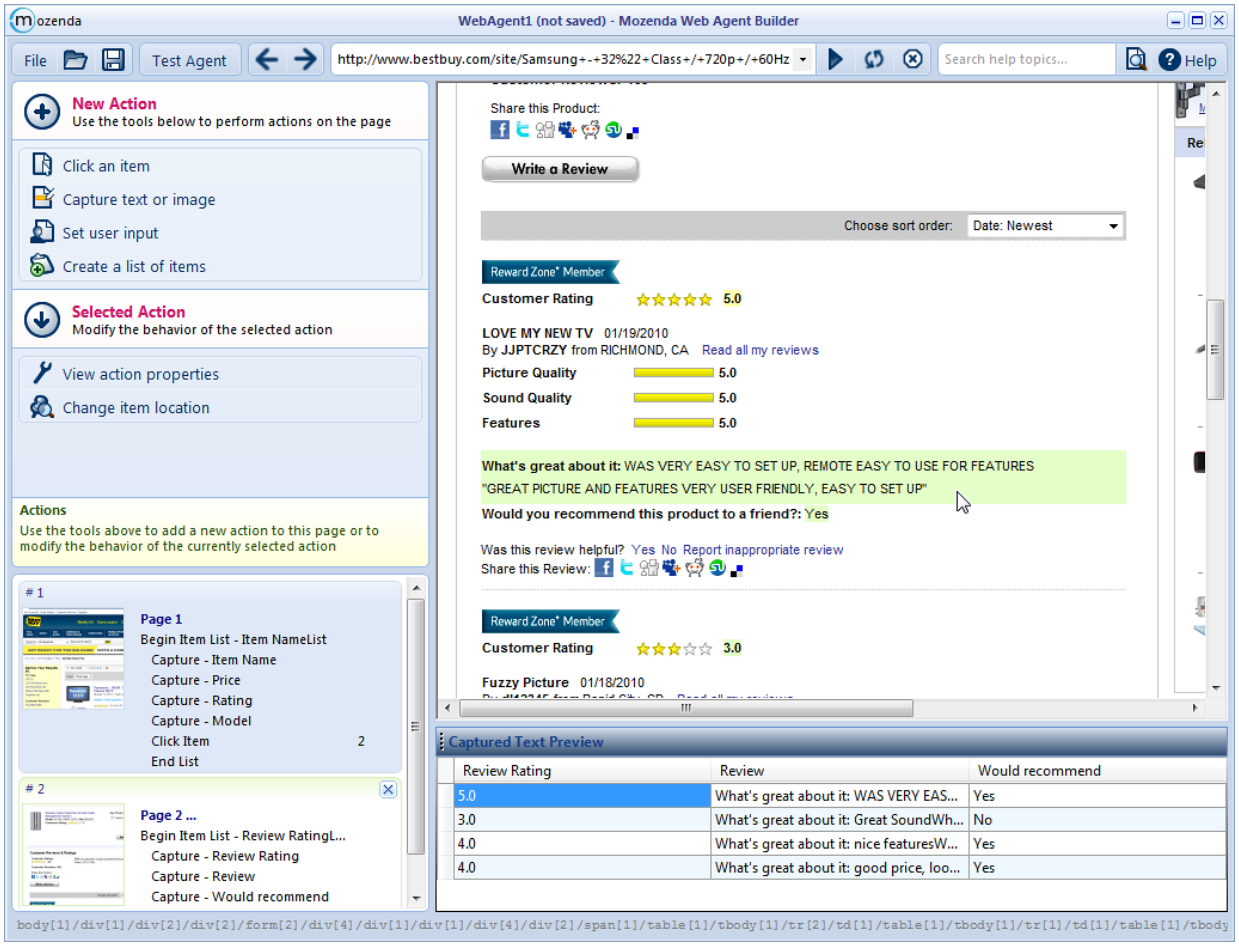


Abbildung 5. Extraktion der Kundenbewertungen eines Produkts mit Mozenda[9]

Nachteilig wirken die eingeschränkten Möglichkeiten der Navigation über die Webseiten. Es ist z.B. schwierig, eine Liste von Eingaben für ein Formular zu definieren.

Die Datenklassifikation beschränkt sich auf das Benennen einzelner Spalten der Tabelle mit den extrahierten Daten.

Auch der hohe Preis für die Nutzung des Produkts (es wird nach der Anzahl der abgerufenen Webseiten abgerechnet) ist von Nachteil.

[9] http://www.mozenda.com/Tour02-Web-Site-Data-Mining-User-Friendly-Interface

2.6.3 WebSundew

Das Tool WebSundew[10] bietet die Extraktion von Daten und Bildern mithilfe eines integrierten Browsers. Dem Benutzer stehen Wizards für häufige Extraktionsszenarien zur Verfügung, die z.B. die Iteration über mehrere Webseiten unterstützen.

Im integrierten Browser hat der Benutzer die Möglichkeit, die Datenbereiche anhand von XPath-Ausdrücken zu definieren und sofort zu sehen, welche Informationen extrahiert werden (siehe Abbildung 6).

Diese Software ist kostenpflichtig.

Von Vorteil sind die interaktive Bedienung mit den Möglichkeiten, die Navigation über die Webseite in grafischer Form zu definieren und die Datenextraktion mit den Wizards vorzubereiten.

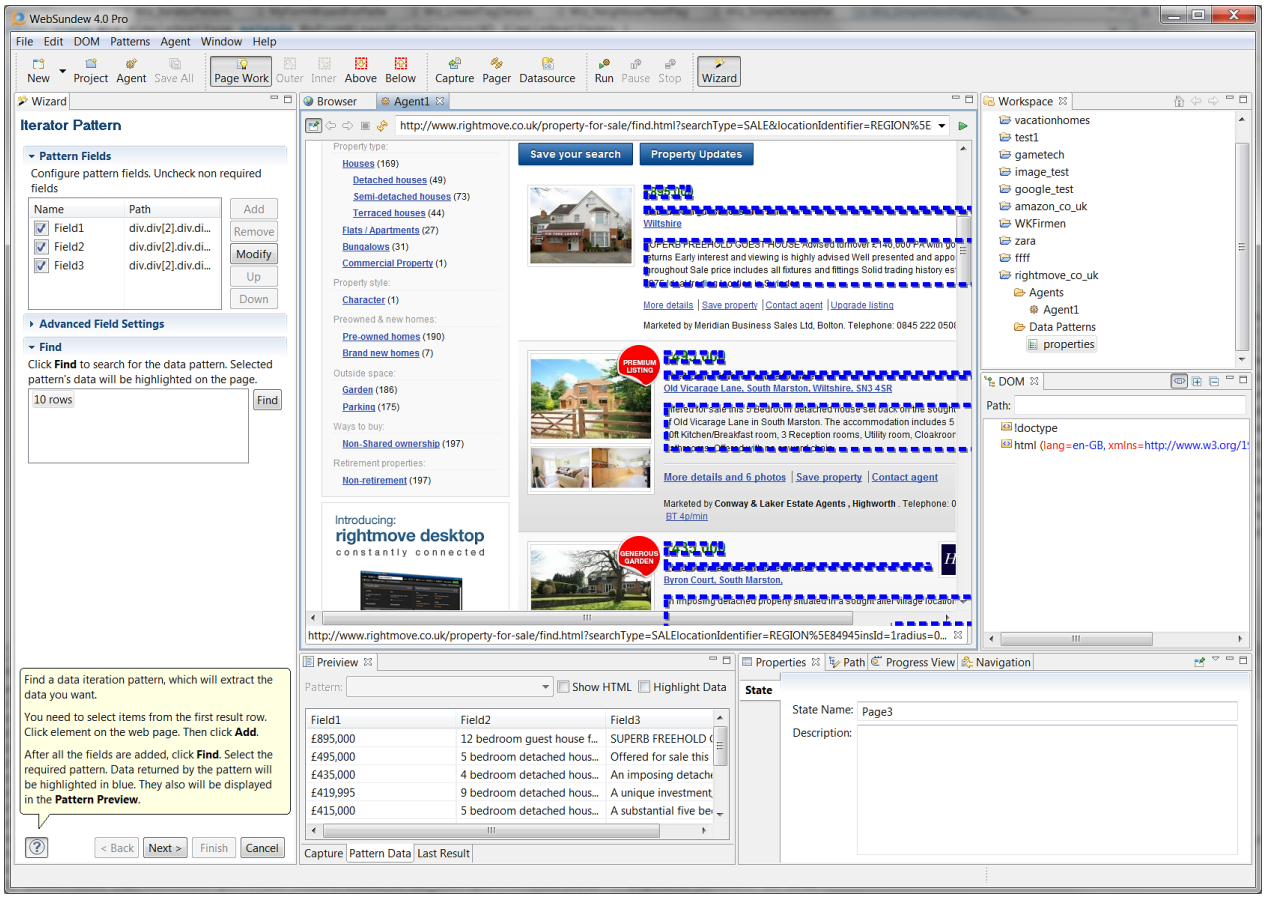


Abbildung 6. WebSundew zeigt anhand von Markierungen die mit XPath ausgewählten Textblöcke[11]

Viele interessante Möglichkeiten für die Datenextraktion sind allerdings nur in den teuren Professional und Enterprise Editionen verfügbar.

Ähnlich wie Mozenda bietet WebSundew keine Datenklassifikation an.

[10] http://www.websundew.com/
[11] http://www.websundew.com/screenshots/

2.6.4 Screen-Scraper

Das Produkt Screen-Scraper[12] ist plattformunabhängig und ermöglicht die Integration mit anderer Software über eine Anzahl der unterstützten Programmiersprachen.

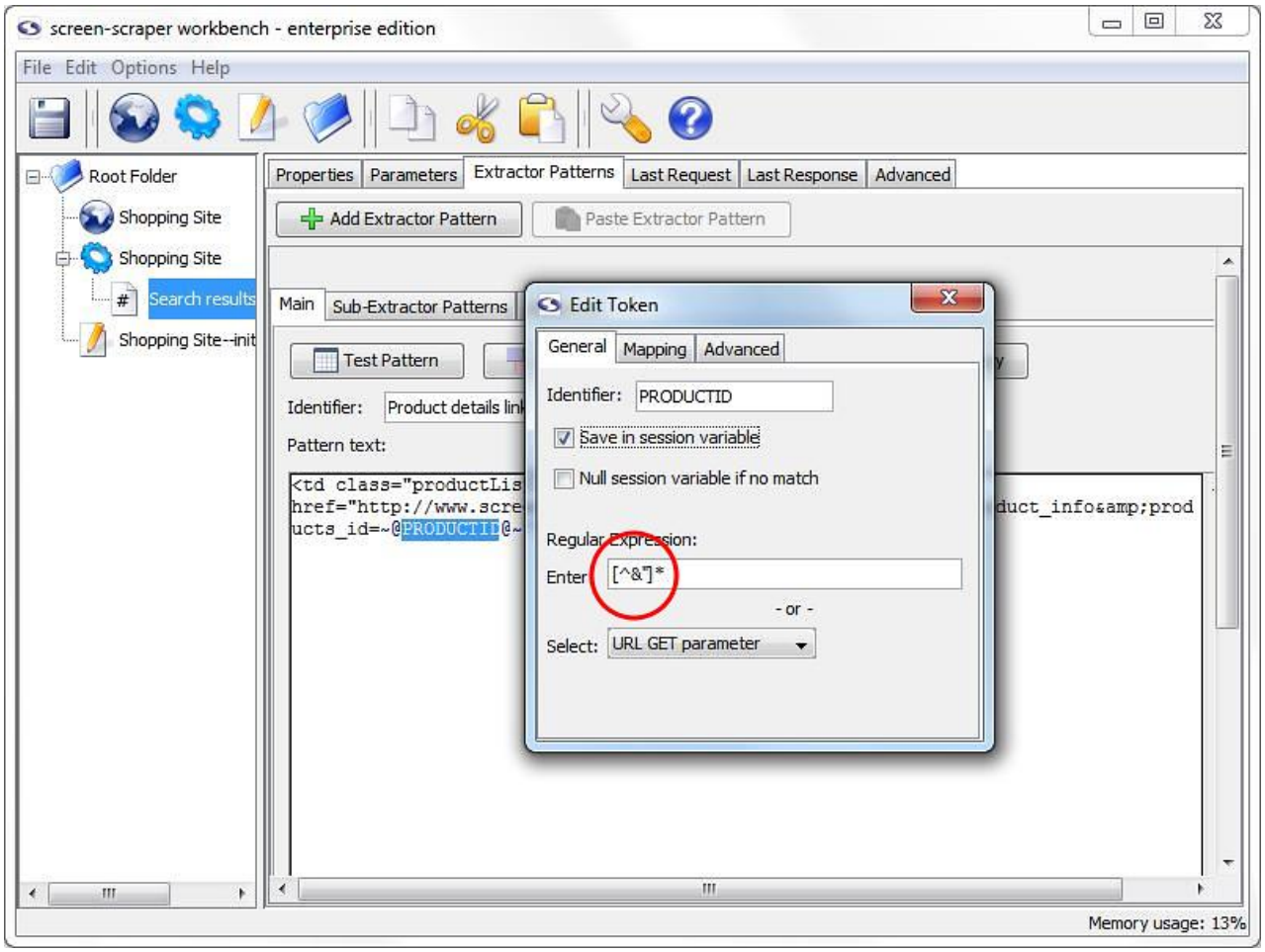


Abbildung 7. Einsatz der regulären Ausdrücke in Screen-Scraper[13]

Die Datenextraktion mit Screen-Scraper erfolgt mit Skripten und Patterns. Die Skripte dienen dazu, um zu den gewünschten Webseiten zu gelangen; mit Patterns können die Informationen gezielt extrahiert werden.

Dieses Tool ist kostenpflichtig, es gibt allerdings eine kostenlose Basisversion mit einem eingeschränkten Funktionsumfang.

Auch Screen-Scraper bietet keine nennenswerte Datenklassifikation an.

2.6.5 Vergleich der Tools

Um den Vergleich der Tools aus den Abschnitten 2.6.1 bis 2.6.4 zu ermöglichen, werden diese Softwareprodukte in Tabelle 2 noch einmal aufgeführt. Diese Tabelle enthält die Bezeichnung der Tools, die verwendete Lizenz, die Verfügbarkeit einer kostenlosen Version, sowie die Vorteile und die Nachteile.

[12] http://www.screen-scraper.com/
[13] http://community.screen-scraper.com/Tutorial_2_page_5

Produkt	Lizenz	Verfügbarkeit	Vorteile	Nachteile
Web-Harvest	BSD-Lizenz	kostenlos	erweiterbar, gute Konfigurationsmöglichkeiten der Datenextraktion mit XPath und XQuery	keine Interaktionsmöglichkeiten mit dem Browser, kein Ausfüllen von Formularen
Mozenda	proprietär	kostenpflichtig	einfaches Bedienen durch die Interaktion mit dem Browser, Export in viele Dateiformate, Zeitplan für die Ausführung der Datenextraktion	die Navigation auf der Website ist nur eingeschränkt möglich, die iterative Extraktion von Daten ist problematisch
WebSundew	proprietär	kostenpflichtig	erleichterte Bedienung mit Wizards, interaktive Benutzung von XPath	eingeschränkte Interaktionsmöglichkeiten mit dem Browser
Screen-Scraper	proprietär	Basisversion kostenlos, sonst kostenpflichtig	gute Konfigurationsmöglichkeiten der Datenextraktion mit XPath und Patterns, Zeitplan für die Ausführung der Datenextraktion	keine Interaktionsmöglichkeiten mit dem Browser

Tabelle 2. Eigenschaften der aktuell verfügbaren Tools für die Datenextraktion aus Webseiten

Die untersuchten Tools haben unterschiedliche Möglichkeiten zur Datenextraktion aus den Webseiten. Kein einziges Tool bietet jedoch eine automatisierte Klassifikation der extrahierten Daten an.

3 Die neue Methodik „EH“

Dieses Kapitel beschreibt die neue Methodik für die automatisierte Extraktion und Klassifikation von Daten aus Webseiten.

3.1 Motivation

Die neue Methodik EH (EH steht für „Extraction Heuristic“) soll eine automatisierte Extraktion und Klassifikation von Daten aus Webseiten ermöglichen. In erster Linie werden mit dieser Methodik Produkt- und Adressdaten aus den Webseiten extrahiert, nichtsdestotrotz soll sie auch auf weitere Domänen übertragbar sein.

Die Extraktion von Daten erfolgt aus dem HTML-Dokument der Zielwebseite. Dabei spielen die strukturellen Merkmale des Dokuments, vor allem die ähnlichen und sich wiederholenden Strukturen, die größte Rolle.

Für die Datenklassifikation kommen Ontologien zum Einsatz. Eine Ontologie beschreibt, welche Daten extrahiert werden und welche Eigenschaften sie besitzen. Die Verwendung von Ontologien ermöglicht das automatische Erkennen von Inhalten wie z.B. Produktpreis, Anschrift, Telefonnummer oder E-Mail.

Für die Anwendung der Methodik EH müssen folgende Voraussetzungen erfüllt sein:

1. Die Webseite und die Inhalte sollen in HTML vorliegen. Auf andere Inhalte wie z.B. Adobe Flash oder Web-Anwendungen wie Microsoft Silverlight kann kein Zugriff erfolgen, weil keine semistrukturierten Daten vorliegen (siehe Abschnitt 2.5.4). Prinzipiell ist es möglich, die Datenextraktion aus XML-Dateien durchzuführen. Dies ist jedoch nicht das primäre Ziel dieser Methodik, außerdem kann der Bereich „Navigation“ auf XML-Daten nicht angewendet werden.
2. Die Webseite muss über eine valide Struktur verfügen. Eine „valide Struktur“ bedeutet in diesem Zusammenhang, dass die Webseite problemlos von einem Browser geöffnet und dargestellt werden kann und alle Inhalte der Webseite über das DOM verfügbar sind.
3. Die zu extrahierenden Informationen müssen voneinander durch Tags abgegrenzt sein. Die Verwendung von Zeilenumbrüchen oder von speziellen Zeichen wie z.B. Tabulator allein reicht nicht aus. Diese Einschränkung kann jedoch bei einer Weiterentwicklung der Methodik aufgehoben werden.
4. Für die automatische Erkennung von Listen muss die Webseite eine iterative Struktur aus HTML-Knoten enthalten (eine Tabelle oder eine Auflistung).
5. Die gesamte Interaktion mit der Website muss innerhalb eines Browserfensters stattfinden: Websites, die mehrere Fenster öffnen (z.B. Popups) werden nicht unterstützt. Ein interessanter Aspekt für die Weiterentwicklung der Methodik ist die Unterstützung der Interaktion mit mehreren Webseiten einer Website in

verschiedenen Browserfenstern. Durch parallele Verarbeitung mehrerer Webseiten kann der Prozess der Extraktion und der Klassifikation von Daten beschleunigt werden. Diese Fragestellung liegt jedoch außerhalb der vorliegenden Arbeit.

Da Webseiten für Menschen erstellt werden, ist ein bestimmtes Vorgehen, um zu den gesuchten Informationen zu gelangen, bereits vorgegeben. So bieten die meisten Websites eine Einstiegsseite an. Auf dieser Seite haben Besucher die Möglichkeit, über Links oder durch das Ausfüllen eines Formulars zu weiteren Seiten zu gelangen. Auf der Zielseite befindet sich schließlich die Information, die von den Benutzern benötigt werden.

Oft ist die Information in Form einer Liste vorhanden (vgl. die Suchergebnisse einer Suchmaschine), dabei wird auf der ersten Seite nur eine begrenzte Zahl der Einträge aus der Liste angezeigt. Um zu weiteren Einträgen zu gelangen, muss der Benutzer über die entsprechenden Links navigieren.

Damit die automatisierte Datenextraktion funktioniert, muss die neue Methodik die Navigation auf Websites berücksichtigen. Ohne Navigation kann die neue Methodik lediglich auf das Surface Web angewendet werden, was ihre Einsatzmöglichkeiten deutlich verringert (vgl. Abschnitt 2.5.5). Außerdem können die Informationen in Listen, die über mehrere Webseiten verteilt sind, nicht ohne Navigation erreicht werden.

Webseitennavigation ist jedoch ein sehr breites Fachgebiet mit vielen Themen wie z.B. dem Bearbeiten von Formularen [34]. Die Anwendung der Webseitennavigation in dieser Methodik beschränkt sich deshalb auf die folgenden Aspekte:

- Öffnen einer vorgegebenen Webseite,
- Navigation über Listen, die mittels Paginierung[14] über mehrere Webseiten verteilt sind,
- Ausfüllen von Formularen, um die Kriterien der Datenausgabe zu definieren,
- Anklicken der Links und Buttons, um zu der Zielseite zu gelangen.

Diese Aspekte sind ausreichend, um auf den meisten Websites zu den zu extrahierenden Inhalten zu gelangen. Eine Weiterentwicklung der Methodik kann diese Aspekte um weitere Punkte wie beispielsweise die Verwendung von Cookies oder die Nutzung der Browser-Historie ergänzen. Eine andere Möglichkeit für die Weiterentwicklung besteht in der Realisierung von Konzepten von algorithmischen Programmiersprachen, wie beispielsweise bedingte Anweisungen, Aufruf von Funktionen oder parametrisierte Schleifen.

[14] Paginierung ist ein Verfahren, um Listen und Tabellen mit besonders vielen Einträgen über mehrere Seiten zu verteilen. Dabei zeigt jede einzelne Seite lediglich einen Teil aller Einträge. Ein Benutzer hat die Möglichkeit, in der Liste mit den Einträgen zu navigieren, z.B. zur nächsten Seite oder zur letzten Seite zu wechseln.

Im einfachsten Fall besteht die Navigation aus dem Öffnen der Webseite, die die zu extrahierenden Daten enthält. Bei komplexeren Szenarien muss z.B. ein Formular ausgefüllt und ein Button angeklickt werden, ehe der Benutzer an die zu extrahierenden Informationen herankommt.

Die vorhandenen Lösungen zur Datenextraktion von Webseiten setzen auf die aktive Rolle des Benutzers. In einigen vorhandenen Anwendungen muss ein Benutzer z.B. die zu extrahierenden Textblöcke im Browser anklicken. Bei einer Tabelle mit mehreren Spalten müsste der Benutzer somit jede Spalte einzeln markieren, um sie zur ausgewählten Datenmenge hinzuzufügen. Die neue Methodik soll im Gegensatz dazu erst die Struktur mit ähnlichen Einträgen erkennen, um alle Spalten einer Tabelle als solche zu kennzeichnen.

Im nächsten Schritt können die Daten in einzelnen Spalten mit den Begriffen der Ontologie verglichen werden, um das Mapping zwischen den Spalten der Tabelle und den Begriffen der Ontologie herzustellen. Stimmen die Merkmale eines Begriffs mit den Merkmalen der Inhalte einer Spalte überein, so wird diese Spalte dem jeweiligen Begriff zugeordnet. Wenn eine automatische Zuordnung nicht möglich ist, kann der Benutzer die Zuordnung manuell durchführen. Sobald eine Spalte allen notwendigen Begriffen aus einer Ontologie zugeordnet wurde, ist das Mapping vollständig.

Die einzelnen Schritte des ganzen Vorgangs werden in ein Scrapingszenario eingefügt. Das Scrapingszenario enthält alle Schritte, die für die Navigation, die Datenextraktion und die Datenklassifikation erforderlich sind (vom Öffnen einer Webseite bis zum Speichern der Ergebnisse).

Ein Scrapingszenario besteht aus Aktionen. Eine Aktion kann die Eingaben eines Benutzers simulieren und die Daten extrahieren, verarbeiten und speichern.

Der Benutzer kann für jede Website, die die für ihn interessanten Informationen enthält, ein Scrapingszenario erstellen. Nach einmaliger Konfiguration einzelner Aktionen kann dieses Szenario immer wieder ausgeführt werden, um die aktuellen Daten aus einer Website zu extrahieren.

3.2 Beschreibung der Methodik EH

Die Methodik EH beinhaltet alle Schritte, die für die Extraktion und Klassifikation von Daten aus Webseiten notwendig sind. Sie bietet einen Prozess an, der als Vorlage bei der Implementierung dieser Methodik dienen kann, und berücksichtigt die Handlungen des Benutzers um diesen Prozess zu steuern. Der Prozess besteht aus vier Stufen:

1. Navigation
2. Datenextraktion
3. Datenklassifikation
4. Ausgabe

Die Reihenfolge der Stufen, die Handlungen des Benutzers sowie die Ein- und Ausgabedaten sind in der Abbildung 8 angezeigt.

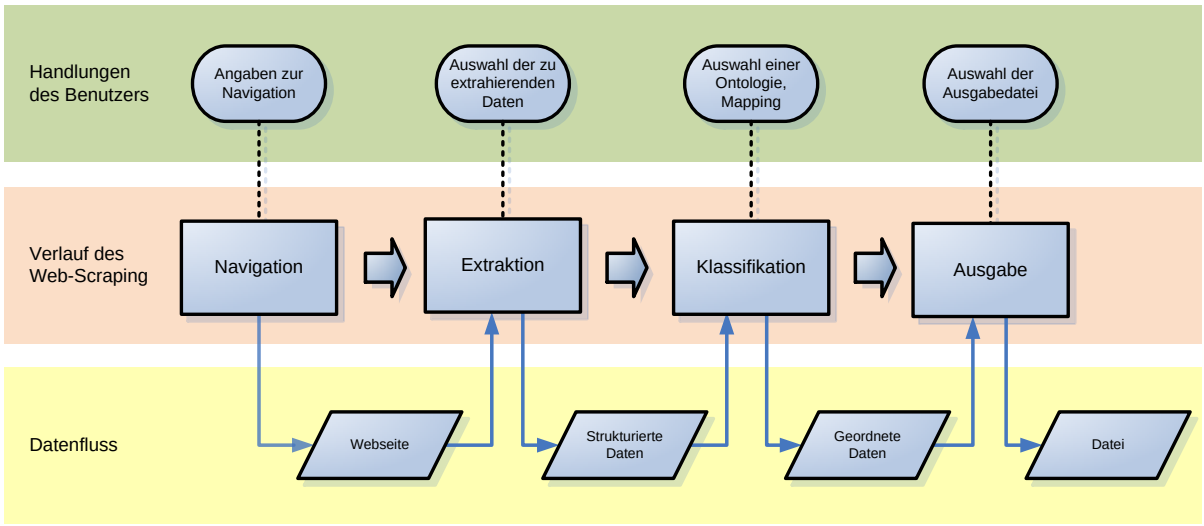


Abbildung 8. Konzeptionelle Zusammenfassung der Methodik EH

Zur Extraktion von Daten durch die Methodik EH sind folgende Angaben des Benutzers notwendig:

1. Die Adresse der Einstiegsseite sowie die Angaben zur Navigation (z.B. welche Links sollen verfolgt werden oder welche Werte sollen in Textfelder eines Formulars eingetragen werden),
2. Eine Ontologie mit der Beschreibung der Informationen, die von den Webseiten gewonnen werden sollen,
3. Dateiname für die Ausgabe.

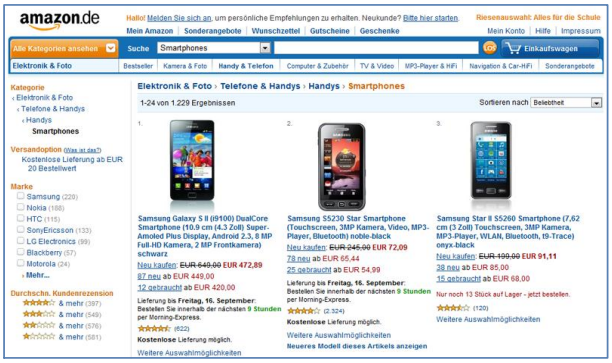
Durch diese Angaben werden der Methodik die erforderlichen Informationen geliefert, um die einzelnen Aktionen der Reihe nach auszuführen. Abbildung 9 zeigt die einzelnen Schritte der Methodik, angewendet auf die Website eines Online-Shops. Nach der Navigation zur Webseite mit der Artikelliste findet zunächst die Analyse der Liste statt. Im nächsten Schritt werden die Inhalte der erkannten Datensätze analysiert und anhand ihrer Merkmale auf die Elemente einer Ontologie abgebildet. Danach erfolgt die Datenausgabe in Form einer Tabelle.

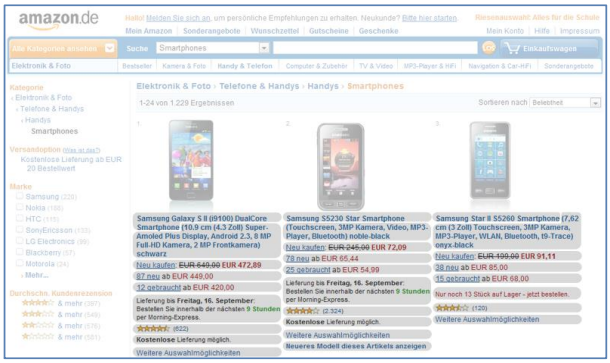
In den folgenden Abschnitten werden die einzelnen Prozessstufen der Methodik erklärt.

3.2.1 Navigation

Zusammengefasst besteht die Aufgabe dieser Stufe darin, von der Einstiegsseite zu den Zielseiten mit strukturierten Daten zu gelangen und den Quelltext dieser Zielseiten weiterzugeben.

1. Navigation von der Startseite zur Webseite mit den Daten





2. Analyse der Webseite und Erkennen von ähnlichen Inhalten

Ontologie

3. Zuordnung der Inhalte zu den Elementen einer Ontologie

4. Ausgabe

Product	Description	Price
Samsung Galaxy S II (i9100) DualCore Smartphone	(10.9 cm (4.3 Zoll) Super-Amoled Plus Display, Android 2.3, 8 MP Full-HD Kamera, 2 MP Frontkamera) schwarz	EUR 472,89
Samsung S5230 Star Smartphone	(Touchscreen, 3MP Kamera, Video, MP3-Player, Bluetooth) noble-black	EUR 72,09
Samsung Star II S5260 Smartphone	(7,62 cm (3 Zoll) Touchscreen, 3MP Kamera, MP3-Player, WLAN, Bluetooth, t9-Trace) onyx-black	EUR 90,59

Abbildung 9. Beispiel der Anwendung der Methodik EH auf eine Website[15]

[15] www.amazon.de

Die Aktionen aus diesem Bereich simulieren die Eingaben eines Benutzers. Der Benutzer kann auf Links einer Webseite klicken, die Textfelder ausfüllen und Einträge in Comboboxen auswählen.

Der Bereich der Navigation sollte folgende Aktionen anbieten:

- Die Aktion „Webseite öffnen“ lädt die Webseite mit der angegebenen URL.
- Die Aktion „Klicken“ ermöglicht das Anklicken eines Elements auf einer Webseite. Das Element kann dabei beispielsweise durch einen XPath-Ausdruck oder anhand seiner ID definiert werden.
- Die Aktion „Text eingeben bzw. Eintrag auswählen“ kann auf Textfelder, Comboboxen und Auswahllisten angewendet werden. Sie sollte ein Formular anhand der zuvor vom Benutzer spezifizierten Angaben automatisiert befüllen, ComboBoxen auswählen und ein Element einer Auswahliste selektieren können.
- Die Aktion "Weitere Webseite" sollte den Benutzer automatisch über *n* Seiten führen können bis die letzte Seite erreicht ist; beispielsweise durch Anklicken eines Buttons.
- Die Aktion „Warten“ sollte dem Benutzer ermöglichen, eine Pause mit einer durch ihn definierten Länge in das Scrapingszenario einzufügen. Diese Aktion wird z.B. zur Verzögerung beim Datenaustausch durch Ajax notwendig.

Diese Aktionen erlauben das Erstellen von komplexen Navigationsszenarien. Es ist z.B. möglich, auf der Website einer Fluggesellschaft den Abflug- sowie den Zielflughafen einzugeben, die Daten des Flugs zu setzen, und so zu der Liste mit den Flugpreisen zu gelangen.

3.2.2 Datenextraktion

Als Eingabe für diese Stufe dient der Quelltext einer Webseite. Dieser Quelltext enthält eine Beschreibung verschiedener Merkmale von einem oder von mehreren Objekten in tabellarischer Form (z.B. eine Liste der Bücher mit Angaben wie Autor, Erscheinungsjahr, Preis usw.). Die Stufe "Datenextraktion" kann beispielsweise durch den Algorithmus MDR (siehe Abschnitt 3.3.2) erfolgen. Dieser Algorithmus dient zur Identifikation der strukturierten Inhalte, zur Abgrenzung der verschiedenen Objekte, zur Extraktion der Datensätze und der Erhaltung der Relationen sowie Merkmale zwischen den Objekten.

Nach der Datenextraktion sollten die Listendaten in eine für die weitere Verarbeitung geeignete Form (z. B. Tabelle) gebracht werden. Die extrahierten Einzelwerte (Variablen) werden als Zeichenfolgen gespeichert.

Für die Datenextraktion bieten sich zwei verschiedene Aktionen an:

- Die Aktion „Extrahiere Variable“ ermöglicht die Extraktion eines Textblocks, der nur einmal auf einer Webseite vorkommt.
- Die Aktion „Extrahiere Liste“ bietet die Extraktion von strukturieren Daten in Listenform oder als Tabelle.

3.2.3 Datenklassifikation

Die Aufgabe der Datenklassifikation besteht darin, die Spalten der extrahierten Listendaten und die Variablen auf die Elemente einer Ontologie abzubilden. Für die automatische Abbildung (Mapping) von Werten auf Elemente einer vorhandenen Ontologie können beispielsweise reguläre Ausdrücke verwendet werden. Hierbei würde ein regulärer Ausdruck das Muster beschreiben, das in den extrahierten Daten vorhanden sein soll, um diese auf ein Element der Ontologie abzubilden. Einem Benutzer sollte jedoch die Möglichkeit gegeben werden, den automatisch erkannten Datentyp anzupassen. Falls die automatische Erkennung fehlgeschlagen ist, kann er so den Datentyp manuell angeben.

Um die extrahierten Daten einer vom Benutzer angegebenen Ontologie zuzuordnen, sollte eine Aktion "Klassifizieren" angeboten werden.

3.2.4 Ausgabe

Die letzte Stufe eines Web-Scraping-Prozesses sollte die Speicherung der extrahierten und klassifizierten Daten darstellen. Diese Stufe benötigt die folgenden zwei Aktionen:

- Die Aktion „Ausgabe auf den Bildschirm“, die dazu dient, die Ergebnisse der Datenextraktion und Klassifikation in Form einer Tabelle auf dem Bildschirm darzustellen.
- Die Aktion „Ausgabe in eine Datei“, die das Speichern der extrahierten Daten in eine Datei ermöglicht.

3.3 Verfügbare Technologien und Verfahren

Die vorgeschlagene Methodik setzt die Anwendung mehrerer Technologien und Verfahren voraus.

Für die Navigation sowie für den Zugriff auf die Inhalte einer Webseite kommt das Document Object Model (DOM) zum Einsatz (siehe Abschnitt 2.2.3). Das DOM bietet ein einfaches aber gleichzeitig mächtiges Mittel, um mit Webseiten zu interagieren. So können die Benutzerhandlungen wie das Anklicken eines Links oder das Ausfüllen eines Formulars simuliert werden.

Für das Auswählen von HTML-Elementen kann die Abfragesprache XPath verwendet werden.

3.3.1 XPath

Die „XML Path Language“ (XPath) ist eine Abfragesprache, mit der einzelne Elemente oder Teile von XML-Dokumenten adressiert werden können [6]. Ein XML-Dokument verfügt über eine baumähnliche Struktur; XPath benutzt Relationen zwischen einzelnen Knoten für die Adressierung.

Wie bereits in Abschnitt 2.5.6 erklärt, muss ein HTML-Dokument nicht zwingend alle Voraussetzungen eines XML-Dokuments erfüllen. Das DOM unterstützt jedoch die XPath-Abfragen und ermöglicht somit die Anwendung von XPath über HTML.

Ein XPath-Ausdruck besteht aus Achsen, Knotentests und Prädikaten. Eine Achse ermöglicht das Adressieren von Knoten aus der oberen, der gleichen oder der unteren Ebene im Baum bezüglich eines bestimmten Knotens. So referenziert z.B. die Achse „/html/body/table“ alle Tabellen innerhalb des Body-Elements, das sich innerhalb von Tags „<html>…</html>“ befindet.

Mit Knotentests können die Elemente einer Achse zusätzlich eingeschränkt werden. Der XPath-Ausdruck „/descendant-or-self::a/child::*“ wählt beispielsweise alle Elemente, die innerhalb von Links platziert sind.

Prädikate ermöglichen die Auswahl von Elementen, die bestimmte Bedingungen erfüllen. So kann z.B. sichergestellt werden, dass nur das Element mit einem festgelegtem Wert des Attributs gewählt wird, wie im folgenden Ausdruck: „//a[@id=‘nextPageLink‘]“. Hier wird lediglich das Anker-Element mit dem ID „nextPageLink“ ausgewählt.

XPath bietet mehrere Möglichkeiten, um dasselbe Element zu adressieren, z.B. nur über seine Stelle in der Struktur des Dokuments oder über zusätzliche Merkmale eines Attributs oder sogar über die Eigenschaften seines Inhalts. Da ein Element jedoch immer nur einen Pfad zur Wurzel des HTML-Dokuments besitzt, kann dieser Pfad in Form eines XPath-Ausdrucks als die eindeutige „Adresse“ des Elements benutzt werden.

Mit XPath ist es möglich, korrekte Verweise auf HTML-Elemente beizubehalten, selbst nach der Aktualisierung einer Webseite, die die Struktur dieser Webseite verändert. Hierfür können HTML-Elemente nicht anhand ihrer Position im DOM-Baum, sondern anhand ihrer Attribute adressiert werden (sofern vorhanden). Viele Attribute, wie beispielsweise die ID eines HTML-Elements, bleiben über eine lange Zeit konstant, obwohl sich die Struktur einer Webseite verändert. Die Nutzung solcher Attribute in einer XPath-Abfrage erhöht die Wahrscheinlichkeit, dass ein HTML-Element auch nach mehreren strukturellen Veränderungen mit dem ursprünglichen XPath-Ausdruck adressiert werden kann.

3.3.2 MDR-Algorithmus

Der „Mining Data Records“ Algorithmus (MDR) wurde von Liu et al. [35] entwickelt, um die strukturellen Muster auf Webseiten völlig automatisch zu erkennen und dadurch die Datensätze zu identifizieren. Dieser Algorithmus analysiert eine Webseite und ermittelt alle Datenbereiche und Datensätze (im Original: „data regions“ und „data records“). Ein Datensatz beschreibt ein Objekt und seine Eigenschaften. Ein Datenbereich ist eine Gruppe von Datensätzen, die ähnliche Objekte beschreiben und sich in der Hierarchie eines HTML-Dokuments auf der gleichen Ebene befinden. In Abbildung 10 bilden drei Laptops einen Datenbereich. Jeder Laptop kann als Datensatz mit den folgenden Informationen angesehen werden: Beschreibung, Neupreis, Gebrauchtpreis, Kundenbewertung und Möglichkeit einer kostenlosen Lieferung.

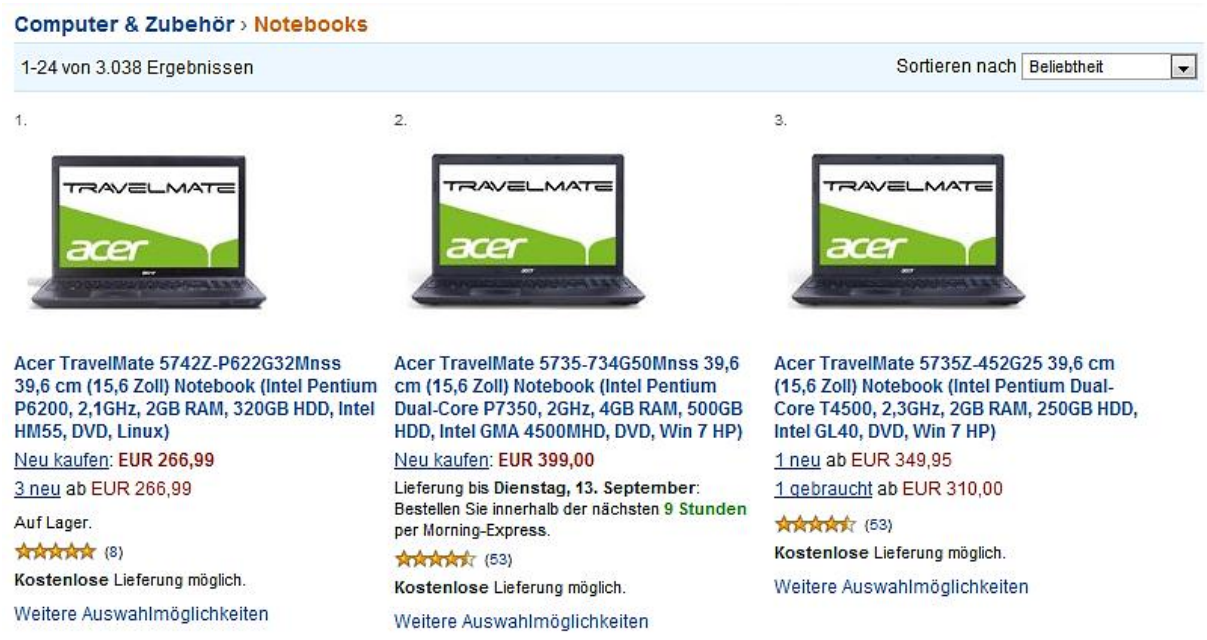


Abbildung 10. Ein Datenbereich mit drei Datensätzen auf der Website eines Online-Shops[16]

Die Analyse des HTML-Dokuments hinter dieser Webseite zeigt, dass die HTML-Abschnitte, die die Beschreibung von Laptops enthalten, sehr ähnlich aufgebaut sind (siehe Abbildung 11). Jeder Datensatz besitzt zunächst das Wurzelelement „div“. Dieses Element hat drei untergeordnete Tags (die „Kinderelemente“). Das erste untergeordnete Element enthält die Nummer des Laptops in der Liste, das zweite Element enthält die grafische Abbildung des Produkts und das dritte Element enthält die textuelle Beschreibung. Sowohl das zweite als auch das dritte Element haben ihre eigenen untergeordneten Knoten, wobei deren Struktur übereinstimmt.

Solche strukturelle Ähnlichkeiten können von einem Algorithmus erkannt werden, und das ist genau das Ziel des MDR-Algorithmus. Neben den strukturellen Ähnlichkeiten nutzt dieser Algorithmus weitere Erkenntnisse über Webseiten. Die meisten Datensätze sind z.B. zusammenhängend. Es ist also sehr unwahrscheinlich, dass im Beispiel der

[16] www.amazon.de

Abbildung 10 die Beschreibung zum ersten Laptop unter dem Knoten mit der ID „result_2“ vorkommt. Des Weiteren ist es üblich, dass alle Datensätze einen gemeinsamen Wurzelknoten haben. Für das betrachtete Beispiel bedeutet dies, dass der Knoten mit der ID „result_2“ mit sehr hoher Wahrscheinlichkeit auf der gleichen Ebene, wie die Knoten mit den IDs „result_0“ und „result_1“ vorkommt.

```
<div id="atfResults" class="grid results cols3">
  <div id="result_0" class="result firstRow product" name="B0054466IC">
      <div id="srNum_0" class="number">1.</div>
    <div class="image">
    <div class="data">
  </div>
  <div id="result_1" class="result firstRow product" name="B005EW9OTC">
      <div id="srNum_1" class="number">2.</div>
    <div class="image">
    <div class="data">
  </div>
  <div id="result_2" class="result firstRow product" name="B004RJZAAA">
      <div id="srNum_2" class="number">3.</div>
    <div class="image">
    <div class="data">
  </div>
  <br class="unfloat">
</div>
```

Abbildung 11. Ausschnitt des HTML-Dokuments, dargestellt mit Firebug

Der MDR-Algorithmus besteht aus drei Schritten:

1. Das Erstellen des Tag-Baums für ein HTML-Dokument.
2. Die Analyse und das Erkennen der Datenbereiche auf der Webseite. Dabei wird der Tag-Baum aus dem ersten Schritt verwendet.
3. Das Identifizieren von Datensätzen innerhalb jeden Datenbereichs.

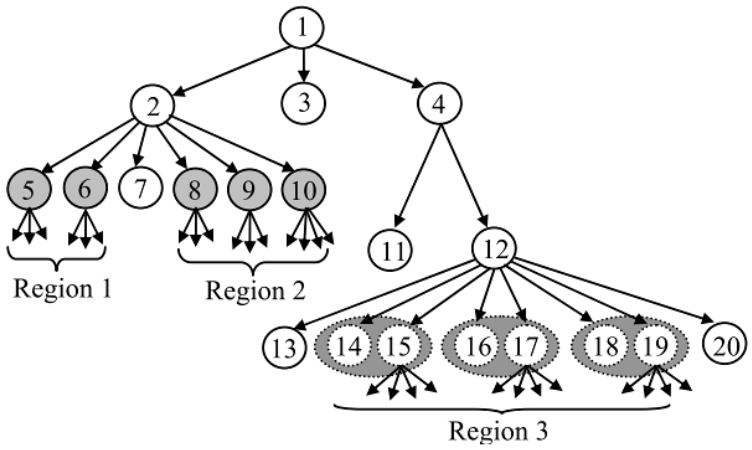


Abbildung 12. Generalisierte Knoten und Datenbereiche im Tag-Baum

Beim Erstellen des Tag-Baums findet die Bereinigung des HTML-Dokuments statt. Für manche Tags wie „
“ oder „<hr>“ fehlen oft die abschließenden Tags „
“ und „</hr>“. Damit beim Erstellen des Baums eine eindeutige Zuordnung von untergeord-

neten Elementen möglich ist, werden diese fehlenden Tags hinzugefügt. Außerdem werden die für die Analyse unbedeutenden Knoten mit Kommentaren oder mit Skripten entfernt.

Im zweiten Schritt ermittelt der Algorithmus die „generalisierten Knoten“ (die originale Bezeichnung: „generalised nodes“). Generalisierte Knoten befinden sich auf der gleichen Ebene des Tag-Baums und haben einen gemeinsamen Wurzelknoten. Abbildung 12 zeigt ein Beispiel des Tag-Baums mit mehreren Datenbereichen („Regions“), bestehend aus den generalisierten Knoten.

Anschließend wird jeder Knoten, angefangen mit dem Wurzelknoten, mit anderen Knoten auf der gleichen Ebene verglichen. Dabei wird die normalisierte Levenstein-Distanz verwendet (siehe Abschnitt 3.3.3).

Um alle generalisierten Knoten zu ermitteln, folgt der Knotenvergleich einem bestimmten Muster. Es wird angenommen, dass ein generalisierter Knoten aus mehreren Tags bestehen kann. In der Praxis liegt die Zahl der Tags, die zu einem generalisierten Knoten gehören können, meistens unter 10. Die maximale Zahl der Knoten, die der Algorithmus auf eine mögliche Zugehörigkeit zu einem generalisieren Knoten überprüft, wird auf K gesetzt. Der Vergleich für die untergeordnete Knoten des Knotens „2“ von der Abbildung 12 läuft demnach wie folgt:

1. (5, 6), (6, 7), (7, 8), (8, 9), (9, 10)
2. (5-6, 7-8), (7-8, 9-10)
3. (5-6-7, 8-9-10)

Zunächst werden also die einzelnen Knoten paarweise verglichen. Danach werden die angrenzenden Knoten zusammengelegt und mit weiteren zusammengelegten Knoten verglichen. Das geht so lang, bis die Anzahl von Knoten innerhalb eines zusammengesetzten Knotens die Zahl K erreicht oder bis mit der Anzahl von Knoten keine weiteren zusammengesetzten Knoten erstellt werden können.

Im letzten Schritt werden die Datensätze innerhalb von Datenbereichen identifiziert. Hierfür werden die generalisierten Knoten in jedem Datenbereich analysiert. Ein generalisierter Knoten kann mehrere Objekte enthalten. In Abbildung 13 werden z.B. zwei generalisierte Knoten als Zeile dargestellt. Jede Zeile enthält jedoch zwei Spalten, d.h. jede Zeile enthält zwei Datensätze. Die Erkennung von einzelnen Datensätzen basiert auf der Annahme, dass einzelne Datensätze über eine ähnliche Struktur verfügen.

row 1	Object 1	Object 2
row 2	Object 3	Object 4

Abbildung 13. Generalisierte Knoten mit jeweils zwei Objekten

Am Ende gibt der Algorithmus alle Datenbereiche und alle Datensätze, die auf einer Webseite erkannt wurden, aus.

Die Zeitkomplexität dieses Algorithmus wird auf $O(nK)$ geschätzt, wo n die Anzahl von Knoten im Tag-Baum darstellt.

3.3.3 Normalisierte Levenstein-Distanz

Um ähnliche Teile der Struktur in einem HTML-Dokument zu erkennen, vergleicht der MDR-Algorithmus Zeichenfolgen miteinander. Ein oft angewendetes Verfahren für den Vergleich von Zeichenfolgen ist die Levenstein-Distanz [5]. Diese Distanz gibt die Anzahl der Veränderungen an, die nötig sind, um die Zeichenfolge s_1 in die Zeichenfolge s_2 umzuwandeln. Als Veränderungen dienen drei atomare Handlungen: das Ersetzen, das Einfügen und das Entfernen eines einzelnen Zeichens.

Die Levenstein-Distanz hängt von der Länge beider Zeichenfolgen ab. Für den MDR-Algorithmus spielt die Länge der zu vergleichenden Zeichenfolgen jedoch keine Rolle, vielmehr ist es wichtig zu ermitteln, in wie weit sich die zwei Zeichenfolgen inhaltlich unterscheiden. Aus diesem Grund verwendet der MDR-Algorithmus die normalisierte Levenstein-Distanz, die wie folgt definiert ist:

$$ND(s_1, s_2) = \frac{d(s_1, s_2)}{(|s_1| + |s_2|)/2}$$

Wie aus der Formel folgt, wird die Levenstein-Distanz durch die halbierte Länge beider zu vergleichender Zeichenfolgen geteilt. In Folge dessen ist die normalisierte Levenstein-Distanz desto geringer, je weniger sich die zwei Zeichenfolgen unterscheiden (bei zwei gleichen Zeichenfolgen ergibt sie den Wert „0“).

Der Algorithmus zur Berechnung der Levenstein-Distanz aus [5] liegt in $O(|s_1||s_2|)$.

In der Praxis gelten zwei Zeichenfolgen als „ähnlich“, sobald die normalisierte Levenstein-Distanz unter einem bestimmten Schwellenwert liegt, meistens unterhalb 0,3 bis 0,5. Falls eine der zu vergleichenden Zeichenfolgen mindestens doppelt so lang ist wie die andere, muss keine genauere Berechnung der normalisierten Levenstein-Distanz erfolgen, da die beiden Zeichenfolgen offensichtlich zu unterschiedlich sind.

3.3.4 Reguläre Ausdrücke

Reguläre Ausdrücke ermöglichen das Beschreiben der Mengen von Zeichenketten mithilfe syntaktischer Regeln in einer speziellen Notation. Ein solcher Ausdruck kann z.B. angeben, welche Bestandteile eine Zeichenfolge besitzen soll, wie oft und in welcher Reihenfolge diese Bestandteile vorkommen sollen.

Für die regulären Ausdrücke existieren vordefinierte Klassen von Inhalten, wie z.B. Buchstaben, Ziffern oder Trennzeichen. So könnte man beispielsweise festlegen, dass

eine Zeichenfolge, die nur aus den Ziffern und aus dem Punktzeichen besteht ein Geldbetrag ist.

Das Erstellen von allgemeingültigen regulären Ausdrücken ist problematisch, weil für Geldbeträge in verschiedenen Ländern entweder der Punkt oder das Kommazeichen verwendet wird. Das gleiche gilt für das Datum: In Deutschland wird das Format „31.12.2011“ verwendet, in den USA dagegen „2011-12-31“.

Für dieses Problem bietet die Methodik EH zwei Lösungen:

1. Falls sich die Extraktion von Daten auf die Webseiten eines einzigen Landes begrenzt, können die landesüblichen Darstellungen für das Datum, für die Währung usw. in die Ontologie unmittelbar übernommen werden.
2. Bei der Extraktion von Daten aus Webseiten aus verschiedenen Ländern können Datentypen für verschiedene Darstellungen innerhalb der Ontologie definiert werden, z.B. „Datum (TT-MM-JJJJ)“ und „Datum (JJJJ-MM-TT)“.

Bei der späteren Verarbeitung der extrahierten Daten können die Daten vereinheitlicht werden, diese Aufgabe ist jedoch nicht Teil der Methodik.

4 Prototypische Implementierung

Dieses Kapitel beschreibt die Anwendung xScraper, die die Methodik EH prototypisch implementiert. Mit dieser Anwendung kann ein Benutzer Web-Scraping-Szenarien erstellen, anpassen und ausführen. Zudem bietet xScraper eine einfache Möglichkeit, um Ontologien für die Klassifikation von Daten zu bearbeiten.

Die Benutzeroberfläche der Anwendung xScraper wird in englischer Sprache entwickelt, um die Möglichkeit zur Nutzung der Anwendung nicht auf den Kreis der deutschsprachigen Benutzer zu beschränken.

4.1 Auswahl geeigneter Technologien

Bei der Auswahl der Technologien (Plattform, Programmiersprache, usw.) für die Realisierung der Anwendung spielten die folgenden Voraussetzungen eine große Rolle:

- Es muss eine Möglichkeit geben, Websites im Browser zu öffnen und mit diesen Websites zu interagieren,
- Das Document Object Model (DOM) muss verfügbar sein,
- Die Anwendung muss über eine einfache und intuitiv klare Benutzeroberfläche verfügen.

Grundsätzlich besteht die Möglichkeit, eine eigenständige Desktop-Anwendung oder ein Browser-Plugin zu entwickeln. Der Vorteil beim Entwickeln eines Browser-Plugings ist die vorhandene Schnittstelle, um mit den Inhalten einer Webseite zu interagieren und auf das DOM zuzugreifen. Bei dieser Lösung wäre die Anwendung jedoch sehr eingeschränkt, weil Browser-Plugins nicht auf alle Funktionen Zugriff haben, die einer Desktop-Anwendung zur Verfügung stehen. Deshalb wurde xScraper als Desktop-Anwendung entwickelt.

Im nächsten Schritt der Auswahl sollte eine passende Lösung für den Zugriff auf einen Browser gefunden werden. Es gibt mehrere Technologien, die genau das ermöglichen:

- „HtmlUnit“ ist ein Java-basierter Webbrowser ohne graphische Darstellung, d.h. er kann eine Webseite öffnen und bietet verschiedene Interaktionsmöglichkeiten, dabei wird die Webseite jedoch nicht angezeigt. Der Vorteil ist in diesem Fall der sparsame Umgang mit den Ressourcen, weil z.B. keine Bilder angezeigt werden müssen. In den meisten Fällen ist es jedoch problematisch, mit einer Webseite „blind“ zu interagieren. Die Navigation kann in so einem Fall nur dann erfolgen, wenn ein weiterer Browser zum Anzeigen der Webseite verwendet wird. Aus diesem Grund konnte HtmlUnit nicht weiter betrachtet werden.

- „Selenium“ ist ein Framework zum Testen von Webseiten und Webanwendungen. Dieses Framework bietet Schnittstellen für verschiedene Programmiersprachen (Java, C#, PHP, usw.) und lässt sich mit vielen Browsern integrieren. Selenium bietet sehr gute Möglichkeiten für den Zugriff auf die Inhalte von Webseiten und unterstützt XPath, kann jedoch keine Inhalte verändern. Eine weitere Einschränkung von Selenium: Falls ein HTML-Tag in einem CSS Stylesheet als versteckt („hidden“) beschrieben ist, wird er im Browser nicht angezeigt und ist für Selenium nicht erreichbar. Eine solche Einschränkung ist beim Testen von Webseiten durchaus sinnvoll, für die Datenextraktion stellt sie jedoch ein Hindernis dar. Deshalb konnte Selenium nicht eingesetzt werden.
- „MSHTML“ oder „Trident“ ist eine Rendering-Engine des Browsers Internet Explorer von Microsoft. Sie kann in einer .NET-Anwendung eingesetzt werden und bietet Zugriff auf das DOM an. Außerdem erlaubt MSHTML eine dynamische Veränderung von Inhalten der Webseiten. Aus der Sicht eines Benutzers wirkt MSHTML wie ein gewöhnlicher Browser, der in einer Anwendung eingebettet ist.

MSHTML ist die einzige Technologie, die den Zugriff auf alle Elemente des HTML-Dokuments erlaubt, unabhängig davon, ob diese Elemente im Browser sichtbar sind. Außerdem ermöglicht MSHTML eine dynamische Veränderung der HTML-Elemente. Diese Gründe waren entscheidend, um MSHTML für die Entwicklung der Anwendung xScraper auszuwählen.

Die Auswahl des MSHTML hat dazu geführt, dass xScraper zur .NET-Anwendung für Windows wurde, weil MSHTML keine weiteren Schnittstellen, z.B. für Java, anbietet. Als Basis für die graphische Benutzeroberfläche wurde Windows Presentation Foundation (WPF) gewählt[17].

Die Entwicklung erfolgte somit mit Microsoft Visual Studio 2010. Für die Versionsverwaltung wurde Subversion eingesetzt.

4.2 Architektur der Anwendung

Die Implementierung der Anwendung xScraper folgt dem Architekturmuster Model-View-Controller (MVC) [18]. Der Einsatz dieses Architekturmusters ermöglicht eine Trennung der Logik von der Benutzeroberfläche einer Anwendung. Das *Model* wird im xScraper durch ein Scrapingszenario dargestellt, die *View* ist die Hauptansicht und der *Controller* ist eine Singleton-Klasse, die die Anwendungslogik enthält.

Die zentralen Klassen von xScraper sind in Abbildung 14 dargestellt. Die Klasse *Controller* wird beim Starten der Anwendung initialisiert. Sie öffnet die Hauptansicht, die in der Klasse *MainWindow* realisiert wurde. Der Controller kann neue Scrapingszenarien

[17] http://msdn.microsoft.com/de-de/library/ms754130.aspx
[18] http://www.phpwact.org/pattern/model_view_controller

erstellen und die bereits existierenden Szenarien aus XML-Dateien laden. Die Implementierung der Scrapingszenarien erfolgt in der Klasse *Scenario*.

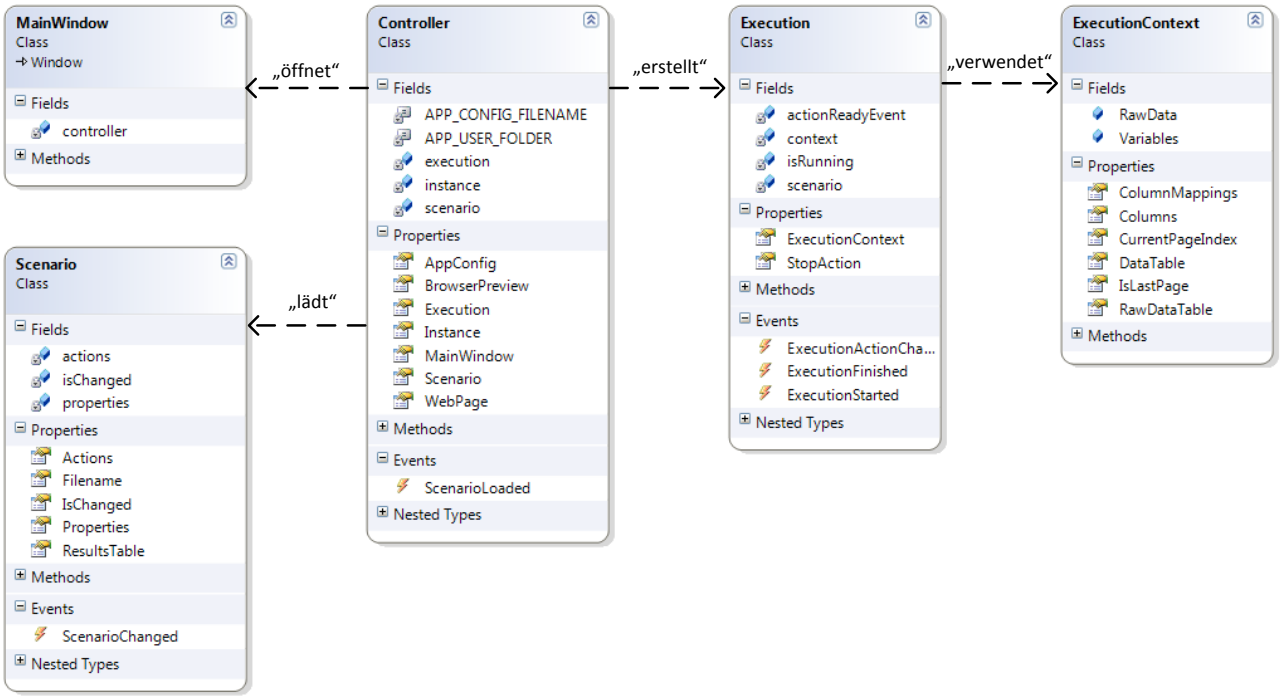


Abbildung 14. Diagramm der wichtigsten Klassen von xScraper

Wenn ein Szenario ausgeführt wird, initialisiert der Controller die Klasse *Execution*. Diese Klasse ermöglicht das Ausführen eines Szenarios in einem neuen Thread, damit die Benutzeroberfläche während der Ausführung nicht blockiert wird. Die Klasse *Execution* erstellt eine Instanz der Klasse *ExecutionContext*, die den Ausführungskontext realisiert. Das Scrapingszenario und der Ausführungskontext werden in weiteren Abschnitten näher beschrieben.

Eine weitere Komponente der Architektur ist der integrierte Webbrowser. Er wurde als Adapter in der Klasse *BrowserPreview* implementiert. Diese Klasse ermöglicht die Navigation auf Websites und den Zugriff auf das DOM.

4.3 Benutzeroberfläche

Die Benutzeroberfläche von xScraper ist mit WPF realisiert. Die Hauptansicht besteht aus den folgenden Komponenten (siehe Abbildung 15):

- Die Titelleiste enthält den Namen der Anwendung und den Namen des Szenarios.
- Im Hauptmenü befinden sich alle Befehle, die zum Erstellen, Öffnen und Speichern von Szenarien und Ontologien notwendig sind.

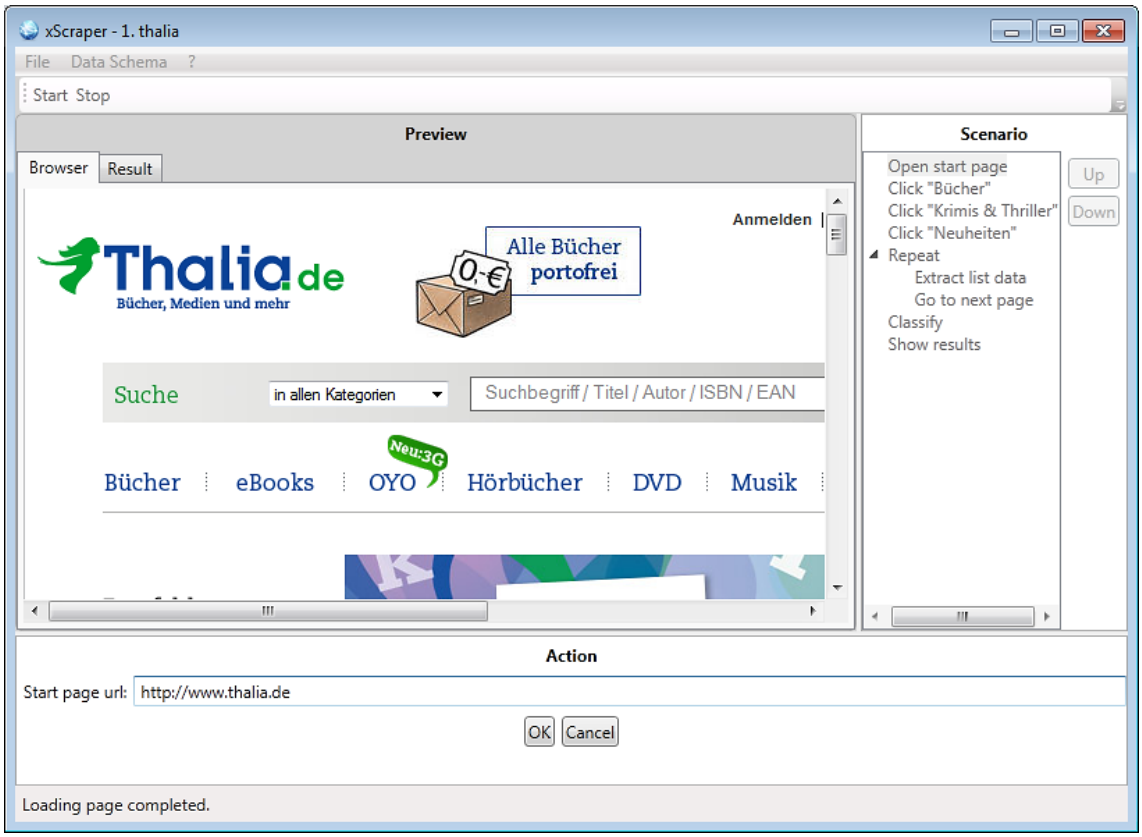


Abbildung 15. Die Hauptansicht von xScraper

- Die Symbolleiste enthält die Tasten, um ein Scrapingszenario ausführen und um die Ausführung abbrechen zu können.
- Der Hauptbereich enthält drei Ansichten: „Preview“, „Scenario“ und „Action“.
    - Die Ansicht „Preview“ besteht aus dem integrierten Browser und aus der Tabelle mit den extrahierten Daten. Der Wechsel zwischen dem Browser und der Tabelle erfolgt über die entsprechenden Lesezeichen.
    - Die Ansicht „Scenario“ enthält alle Aktionen eines Szenarios. Die Aktionen sind in Form eines Baums dargestellt, um die inneren Aktionen einer iterativen Schleife anzeigen zu können.
    - In der Ansicht „Action“ werden die Dialogfenster zum Bearbeiten von Aktionen eingeblendet. Diese Lösung bietet einen wichtigen Vorteil für den Benutzer, weil durch das Einblenden von Dialogfenstern *im* unteren Bereich der Hauptansicht statt *über* der Hauptansicht der Benutzer eine Möglichkeit bekommt, jederzeit mit dem integrierten Browser zu interagieren.
- In der Statusleiste wird der Status des Browsers angezeigt. Dies ist hilfreich, um zu erkennen, wann das Laden einer Webseite abgeschlossen ist.

Neben der Hauptansicht bietet xScraper weitere Dialogfenster. Diese Dialogfenster werden in den folgenden Abschnitten dieses Kapitels beschrieben.

4.4 Scrapingszenario

Ein Scrapingszenario besteht aus Aktionen, die verschiedene Aufgaben im Bereich der Navigation, der Extraktion und der Klassifikation von Daten, sowie bei der Ausgabe erledigen. Beim Erstellen eines neuen Szenarios werden die grundlegenden Aktionen automatisch hinzugefügt. Der Benutzer kann eigene Aktionen einfügen, um die Navigation zu den Webseiten mit den zu extrahierenden Inhalten zu ermöglichen. Abbildung 16 zeigt ein Beispiel eines Scrapingszenarios zum Extrahieren der Adressdaten von Pizzalieferanten für die Postleitzahl 70173.

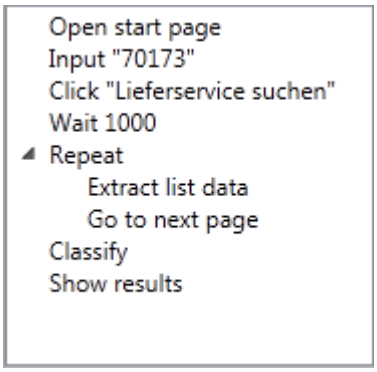


Abbildung 16. Beispiel eines Scrapingszenarios

Das Scrapingszenario ist in der Klasse *Scenario* implementiert. Die Aktionen werden in der Liste *Actions* dieser Klasse gespeichert.

Ein Scrapingszenario kann in eine XML-Datei gespeichert werden. Der Benutzer kann beliebig viele Szenarien erstellen, konfigurieren und speichern.

Neben den Aktionen verfügt ein Szenario über Einstellungen. Diese Einstellungen stehen in der Eigenschaft *Properties* zur Verfügung.

4.5 Ausführungskontext

Der Benutzer kann ein Szenario ausführen. Dabei erstellt der *Controller* einen Ausführungskontext, der in der Klasse *ExecutionContext* implementiert wurde. Der Ausführungskontext enthält alle Daten, die zur Laufzeit der Ausführung eines Szenarios aus einer oder mehreren Webseiten extrahiert werden.

Die extrahierten Daten werden dabei im Wörterbuch *RawData* zwischengespeichert. Dieses Wörterbuch legt eine Tabelle für jede Webseite an, die als Quelle für die Datenextraktion verwendet wurde. Die Spalten dieser Tabelle enthalten die Attribute, die ein Algorithmus erkannt hat. In den Zeilen befinden sich die extrahierten Daten (Objekte).

Des Weiteren enthält der Ausführungskontext die Zuordnung von Datentypen einer Ontologie zu den Attributen von extrahierten Objekten. Außerdem sind im Ausführungskontext interne Daten zwischengespeichert, die z.B. für den Abbruch einer Schleife benötigt werden.

Während der Ausführung eines Szenarios erhält jede Aktion der Reihe nach den Ausführungskontext. Sie kann somit auf die Eigenschaften des Ausführungskontexts zugreifen und die Daten verändern.

4.6 Aktionen

Die Aktionen sind der wichtigste Bestandteil eines Scrapingszenarios. Jede Aktion stellt eine bestimmte Handlung dar. Eine solche Handlung kann z.B. eine Benutzereingabe in ein Textfeld auf einer Webseite oder die Ausgabe der extrahierten Daten sein.

Ein Scrapingszenario enthält eine geordnete Liste von Aktionen. Beim Ausführen eines Szenarios fängt xScraper bei der ersten Aktion an und arbeitet alle folgenden Aktionen der Reihe nach ab, bis alle Aktionen erfolgreich ausgeführt wurden oder bis ein Fehler auftritt.

Die Anwendung xScraper implementiert die Aktionen aus der Methodik EH in eigenen Klassen. Der Zusammenhang zwischen den Aktionen aus der Methodik und den Klassen von xScraper ist in Tabelle 3 abgebildet.

Nr.	Aktion in EH	Implementierung in xScraper	Handlung
1	Webseite öffnen	StartPageAction	Startseite öffnen
2	Weitere Webseite	NextPageAction	Zur nächsten Webseite wechseln (bei der Paginierung)
3	Klicken	ClickAction	Anklicken eines HTML-Elements durch den Benutzer simulieren
4	Text eingeben / Eintrag auswählen	InputAction	Benutzereingabe in einem Textfeld oder in einer Auswahlliste simulieren
5	Warten	WaitAction	Die Ausführung eines Szenarios anhalten
6	Liste extrahieren	ExtractListDataAction	Eine Liste von Daten extrahieren
7	Klassifizieren	ClassifyAction	Die extrahierte Daten anhand einer Ontologie klassifizieren
8	Ausgabe auf den Bildschirm	ShowDataAction	Die extrahierten Daten anzeigen
9	Wiederholen	RepeatAction	Die innere Sequenz von Aktionen in einer Schleife ausführen

Tabelle 3. Die in xScraper implementierten Aktionen

Bei der Implementierung wurde die Basisklasse XSAction erstellt, alle anderen Aktionen erben von dieser Klasse. In den folgenden Abschnitten wird die Implementierung jeder Aktion einzeln erläutert.

4.6.1 StartPageAction

Die Aktion *StartPageAction* ermöglicht das Öffnen der Startseite einer Website. Der einzige Parameter dieser Aktion ist die Adresse einer Website. Bei der Ausführung der *StartPageAction* versucht xScraper innerhalb eines bestimmten Zeitraum die Webseite zu öffnen. Falls dies gelingt, setzt xScraper die Ausführung eines Szenarios fort. Sonst wird die Ausführung eines Szenarios abgebrochen und eine Fehlermeldung wird angezeigt.

4.6.2 NextPageAction

Mit der Aktion *NextPageAction* kann xScraper bei Paginierung von der ersten Webseite zu den nächsten Webseiten navigieren. Diese Aktion, eingefügt in die innere Schleife einer Aktion *RepeatAction*, dient als Abbruchbedingung dieser inneren Schleife von Aktionen. Außerdem hilft sie dabei, die von mehreren Webseiten extrahierten Daten zu verwalten.

Der Seitenwechsel erfolgt durch das Anklicken eines Links. Solche Links können normalerweise auf einer Webseite dadurch erkannt werden, dass sie den Text „weiter“ bzw. „next“ oder das Bild eines Pfeiles enthalten. Manche Websites bieten lediglich die Möglichkeit zu einer bestimmten Webseite zu wechseln, indem der Benutzer die Nummer dieser Seite anklicken muss, z.B. „2“, „3“, usw. Um alle diese Fälle zu berücksichtigen, bietet xScraper zwei Möglichkeiten für die Angabe eines Links an: XPath-Ausdruck oder Dateiname des Bildes auf der Taste.

Falls der Benutzer die Navigation mittels XPath-Ausdrucks auswählt, wird dieser Ausdruck als *NextLinkXPath* gespeichert. Während der Ausführung versucht xScraper, das durch den XPath-Ausdruck definierte HTML-Element zu finden und anzuklicken. Bei der Angabe eines Bildes sucht xScraper nach einem Bild mit dem entsprechenden Dateinamen innerhalb eines Links und klickt diesen Link an. Wenn das gesuchte Element nicht gefunden werden kann, wartet die Anwendung 30 Sekunden und wiederholt die Suche nach dem Element in regelmäßigen Abständen. Diese Wartezeit hilft dabei, die dynamischen Änderungen auf einer Webseite zu berücksichtigen, weil in manchen Fällen eine Verzögerung zwischen dem vollständigen Laden einer Webseite und der Initialisierung aller Elemente dieser Webseite entsteht. Wenn kein Element während der Wartezeit gefunden werden konnte, wird das Flag *IsLastPage* beim Ausführungskontext gesetzt. Dies gilt für die *RepeatAction* als Zeichen dafür, die Ausführung der inneren Schleife mit den Aktionen zu beenden. Bei einem erfolgreichen Seitenwechsel wird eine neue interne Struktur für das Speichern der zu extrahierenden Daten angelegt.

Zusätzlich kann ein Benutzer angeben, ob alle verfügbaren Webseiten oder nur die ersten ***n*** Webseiten geöffnet werden sollen. Die maximale Anzahl der Webseiten wird durch den Parameter *MaxPagesCount* bestimmt. Das Flag *IsLimitedPagesCount* legt fest, ob die maximale Zahl der Webseiten berücksichtigt werden muss.

4.6.3 ClickAction

Die Aktion *ClickAction* ermöglicht das Anklicken eines HTML-Elements auf einer Webseite. Diese Aktion kann an einer beliebigen Stelle im Szenario nach dem Öffnen der Startseite und vor der Ausgabe von Daten eingefügt werden. Die Anzahl der verwendbaren Aktionen dieses Typs ist nicht begrenzt.

Der einzige Parameter der Aktion *ClickAction* ist *XPath*, der den XPath-Ausdruck eines HTML-Elements enthält. Bei der Ausführung startet xScraper eine Abfrage mit dem

Ziel, das durch den XPath-Ausdruck beschriebene HTML-Element zu finden. Falls die Abfrage mehrere HTML-Elemente liefert, wird nur das erste Element berücksichtigt. Wenn die Abfrage kein Element liefert, wird diese Abfrage innerhalt von 30 Sekunden erneut regelmäßig ausgeführt. Diese Verzögerung erlaubt die Ausführung von Skripten auf der Webseite nach der vorherigen Aktion abzuwarten. Es kommt oft vor, dass nach dem Anklicken einer Taste durch eine zuvor ausgeführte Aktion die komplette Webseite neu geladen wird, oder ein Teil der Webseite dynamisch verändert wird. Währenddessen fehlt noch möglicherweise das Element, das die aktuelle Aktion anklicken soll.

Falls kein Element innerhalb der Wartezeit gefunden werden konnte, wird eine Fehlermeldung ausgegeben und die Ausführung des Szenarios wird abgebrochen.

4.6.4 InputAction

Das Ziel der *InputAction* ist die Eingabe von Text in die Textfelder und die Auswahl von Einträgen in den Auswahllisten. Ähnlich wie die *ClickAction*, kann die *InputAction* an einer beliebigen Stelle innerhalb eines Scrapingszenarios zwischen der ersten und der letzten Aktion vorkommen. Die Anzahl der im Szenario genutzten Aktionen von diesem Typ ist nicht beschränkt.

Eine *InputAction* enthält eine Liste von Schlüssel-Wert-Paaren, bei denen der Schlüssel ein XPath-Ausdruck eines Elements und der Wert der Text für die Eingabe ist. Somit kann innerhalb einer *InputAction* die Texteingabe für mehrere Textfelder erfolgen.

Für Auswahllisten wird anstelle des Eingabetexts die Bezeichnung eines Eintrags aus einer Liste gespeichert.

Während der Ausführung wird die Liste von Schlüssel-Wert-Paaren verarbeitet. Für jeden Eintrag wird eine Abfrage des enthaltenen XPath-Ausdrucks ausgeführt. Je nachdem, ob das gefundene HTML-Element ein Textfeld oder eine Auswahlliste ist, wird ein Text eingegeben oder ein Eintrag ausgewählt. Ähnlich wie bei der *ClickAction,* erlaubt die Verzögerung die Ausführung von Skripten auf einer Webseite abzuwarten.

4.6.5 WaitAction

Manche Vorgänge während der Interaktion mit einer Webseite nehmen Zeit in Anspruch, z.B. beim Suchen nach einer Flugkarte muss der Benutzer warten, während die Flugdaten aus einem Buchungssystem abgefragt werden. Bei den Websites, die für das Nachladen von Daten Ajax verwenden, ist das Erkennen solcher Zwangspausen schwierig. In diesem Fall hilft die Aktion *WaitAction*, indem sie die Ausführung eines Szenarios vorübergehend anhält. Der Benutzer bestimmt die Länge der Pause durch die Eingabe der Länge der Pause in Millisekunden als Parameter *Delay*.

4.6.6 ExtractListDataAction

Die *ExtractListDataAction* ermöglicht die Extraktion von Daten aus Tabellen und Listen auf einer Webseite. Bei der Ausführung eines Scrapingszenarios extrahiert diese Aktion die mittels eines XPath-Ausdrucks angegebenen HTML-Elemente und spaltet die Inhal-

te dieser Elemente auf, so dass am Ende eine zweidimensionale Tabelle mit den extrahierten Daten entsteht.

Die zu extrahierenden Daten („Objekte“) können beispielsweise Zeilen einer Tabelle auf einer Webseite sein. Bei der Initialisierung führt die Aktion *ExtractListDataAction* den MDR-Algorithmus aus dem Abschnitt 3.3.2 aus. Dieser Algorithmus analysiert ein HTML-Dokument und liefert alle HTML-Elemente mit ähnlicher Struktur. Er markiert jedoch in der Regel zu viele Elemente als ähnlich, beispielsweise die HTML-Elemente für die Navigation auf einer Webseite. Deshalb werden anschließend die durch MDR gelieferten Elemente mit einem heuristischen Algorithmus gefiltert, um lediglich HTML-Elemente in der Auswahl zu lassen, die für den Benutzer interessant sind. Der Benutzer kann die zu extrahierenden Objekte mittels eines einzigen XPath-Ausdrucks oder mittels mehrerer XPath-Ausdrücke angeben. Die Liste dieser XPath-Ausdrücke wird im Parameter *RecordsXPath* gespeichert. Während der Ausführung eines Szenarios nutzt xScraper im ersten Schritt diese XPath-Ausdrücke, um Abfragen über die Inhalte einer Webseite durchzuführen. Alle durch diese Abfragen gelieferten HTML-Elemente werden zunächst in eine Liste zwischengespeichert.

Im nächsten Schritt spaltet die Aktion *ExtractListDataAction* die Inhalte von HTML-Elementen auf, um die Attribute der extrahierten Objekte voneinander zu trennen. Dabei wird ein Algorithmus eingesetzt, der alle Knoten eines HTML-Baums rekursiv durchläuft und für alle Knoten, die lediglich Text und bestimmte HTML-Tags enthalten, ein Attribut generiert. Dieser Algorithmus „Single Tree Alignment“ basiert auf dem Algorithmus „Partial Tree Alignment“ (PTA) [53]. Der Unterschied zum PTA besteht darin, dass XPath-Ausdrücke eingesetzt werden, um die einzelnen Attribute zu unterscheiden. Listing 1 enthält den Algorithmus „Single Tree Alignment“ als Pseudocode.

Eingabe:
Ausschnitt eines HTML-Dokuments mit dem Wurzelelement $Root$
Anfangselement E
Liste von HTML-Tags T

Ausgabe:
Liste von Tupeln A

Ablauf:
Funktion STA($Root$, E, T, A)
für jedes Kinderelement K in A:
falls das Element K nur Text und HTML-Tags aus T enthält,
füge in die Liste A den XPath-Ausdruck von K bezüglich $Root$ und
die textuellen Inhalte des Knotens K
sonst
STA($Root$, K, T, A)
falls das Element E Text innerhalt des eigenen Tags enthält,
füge in die Liste A den XPath-Ausdruck von E bezüglich $Root$ und
die textuellen Inhalte des Knotens E
Ende

Listing 1. Algorithmus „Simple Tree Alignment“ als Pseudocode

Nach der Ausführung dieses Algorithmus wird für jedes Element aus dem ersten Schritt eine Liste von Tupeln mit Attributen erstellt. Jedes Tupel besteht aus einem XPath-Ausdruck, der einen Knoten im HTML-Baum beschreibt und dem textuellen Inhalt dieses Knotens.

Die mit dem Algorithmus ermittelten Attribute aller extrahierten Objekte aus einer Webseite werden in das Wörterbuch *RawData* des Ausführungskontexts gespeichert. Dieses Wörterbuch enthält die Attribute der extrahierten Objekte für jede Webseite, die mit der Navigation erreicht werden konnte.

4.6.7 ClassifyAction

Die Aktion *ClassifyAction* ermöglicht eine Zuordnung von Attributen der aus einer Webseite extrahierten Objekte zu den Datentypen einer Ontologie. Gleichzeitig dient diese Aktion als Filter, um die extrahierten Inhalte, die keine Objekte sind, zu entfernen. Die Unterscheidung erfolgt anhand der notwendigen Attribute. Wenn mindestens ein notwendiges Attribut fehlt, wird das gesamte Objekt aussortiert. Es gibt jedoch Ausnahmen, die weiter in diesem Abschnitt erläutert werden.

Damit diese Zuordnung möglich ist, muss der Benutzer eine Datei mit einer Ontologie angeben. Die Ontologie wird über die Eigenschaft *DataSchema* geladen und bleibt ein Bestandteil des Szenarios.

Für die Zuordnung von extrahieren Daten zu den Elementen einer Ontologie verwendet xScraper eine Liste von Spalten, die der Tabelle mit extrahierten Daten entnommen werden. Diese Spalten enthalten die Attribute von extrahierten Objekten und können mittels eines XPath-Ausdrucks eindeutig identifiziert werden. xScraper speichert die Liste dieser XPath-Ausrücke ins Szenario.

xScraper erstellt für jeden Datentyp innerhalb einer Ontologie ein Objekt *ColumnMapping*. Dieses Objekt besitzt die Eigenschaft *Target*, die auf einen Datentyp in einer Ontologie verweist. Des Weiteren hat das Objekt *ColumnMapping* die Eigenschaft *Source*, die einen XPath-Ausdruck aus einer Spalte der Tabelle mit extrahierten Daten enthält. Diese Eigenschaften ermöglichen eine Zuordnung zwischen den Attributen eines extrahierten Objekts und den Datentypen einer Ontologie.

Das Objekt *ColumnMapping* bietet zusätzliche Eigenschaften, um die extrahierten Daten zu bereinigen. Mit dem *ExtractionPattern* kann der Benutzer einen regulären Ausdruck angeben, mit dem ein Teil des ursprünglichen Attributs extrahiert werden kann. Die Verwendung dieser Möglichkeit ist sinnvoll, um beispielsweise aus dem Attribut „Neupreis: 19,95 €“ den Betrag „19,95“ zu extrahieren.

Die nächste Eigenschaft, *DefaultValue*, ermöglicht dem Benutzer die Angabe eines Standardwerts für den Fall, wenn ein Attribut eines Objekts fehlt. Die Verwendung

dieser Eigenschaft hilft zu verhindern, dass die Aktion *ClassifyAction* Objekte verwirft, bei denen die notwendigen Attribute fehlen.

Bei der ersten Ausführung benutzt die *ClassifyAction* einen Algorithmus, um eine Zuordnung von Attributen zu den Datentypen einer Ontologie zu ermitteln. Dieser Algorithmus basiert auf einem Naiven Bayes-Klassifikator. Zunächst wird eine Tabelle mit den Testdaten vorbereitet. Hierfür werden aus der internen Tabelle mit den extrahierten Daten n Zeilen ausgesucht. Die Zahl n ist als Konstante MAX_ROWS_TO_ANALYZE in der Klasse *ClassifyAction* gespeichert, bei der Evaluation war n=40.

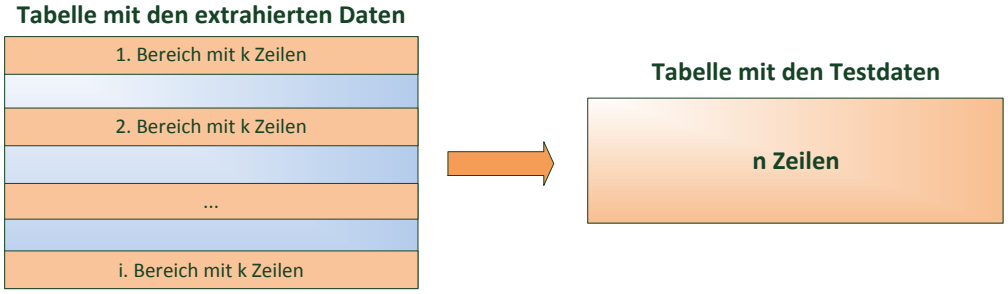


Abbildung 17. Erstellen einer Testtabelle für die Datenklassifikation

Eine weitere Konstante, ADJACENT_ROWS_TO_ANALYZE, legt die Anzahl der angrenzenden Zeilen, die zusammen in die Testtabelle kopiert werden. Während der Evaluation hatte dieser Konstante den Wert 4.

Eingabe:
Zweidimensionale Tabelle T[n, m] mit n Zeilen und m Spalten
Array regex[k] mit den regulären Ausdrücken (Erkennungsmuster) der Datentypen einer Ontologie

Ausgabe:
Matrix M[k, m] mit den folgenden Werten:
1, falls die Spalte m dem Datentyp k zugeordnet wird,
0 sonst

Ablauf:
```
für jeden regulären Ausdruck x aus regex
    initialisiere einen Array p[m] mit dem Wert 0
    für jede Spalte j aus T
        für jede Zeile i aus T
            teste den Inhalt T[i,j] mit dem regex[x]
            falls T[i,j] eine Zeichenfolge enthält, die mit dem regex[x]
            übereinstimmt, erhöhe p[j] um 1
    finde den Index h von p mit dem höchsten Wert
    setze M[x, h] = 1
Ende
```

Listing 2. Algorithmus für die Datenklassifikation als Pseudocode

Die Blöcke von angrenzenden Zeilen werden aus der ursprünglichen Tabelle mit den extrahierten Daten in regelmäßigen Abständen in die Tabelle mit den Testdaten kopiert, damit die Stichproben gleichmäßig die ganze ursprüngliche Tabelle abdecken.

Abbildung 17 verdeutlicht diesen Vorgang. Wenn ein kontinuierlicher Bereich von Zeilen aus k Zeilen besteht, werden $n/k = i$ Bereiche benötigt.

Sobald die Testtabelle mit n Zeilen fertig ist, kann die Klassifikation beginnen. Sie verläuft nach dem Algorithmus, der in Listing 2 abgebildet ist. Dieser Algorithmus verwendet die Erkennungsmuster von Datentypen aus einer Ontologie, um die Testtabelle zu analysieren und eine wahrscheinliche Zuordnung von Attributen zu den Datentypen zu ermitteln.

Falls die Zuordnung inkorrekt ist, kann der Benutzer sie manuell anpassen. Die Aktion *ClassifyAction* kann nur einmal im Szenario enthalten sein.

4.6.8 ShowDataAction

Diese Aktion dient dazu, die extrahierten Daten in Form einer Tabelle anzuzeigen. Sie ist die letzte Aktion eines Szenarios.

4.6.9 RepeatAction

Die Aktion *RepeatAction* erlaubt eine iterative Ausführung von Aktionen. Sie enthält eine innere Schleife mit Aktionen, die so lange ausgeführt werden, bis eine Abbruchbedingung erfüllt wird. Diese Bedingung wird durch das Flag *IsLastPage* des Ausführungskontexts dargestellt. Sobald dieser Wert wahr ist, wird die Ausführung der Aktionen aus der inneren Schleife beendet und die nächste Aktion nach *RepeatAction* wird ausgeführt.

Die Anwendung xScraper erlaubt nur eine Aktion *RepeatAction* im Scrapingszenario.

4.7 Ontologien

Ontologien sind notwendig, damit xScraper die extrahierten Daten klassifizieren kann. Diese Klassifikation besteht darin, die Spalten einer Tabelle mit den Attributen von extrahierten Objekten auf die Elemente einer Ontologie abzubilden. In der prototypischen Entwicklung werden deshalb Ontologien mit einer einfachen Struktur eingesetzt. Solche Ontologien sind ausreichend für die Datenklassifikation; die Unterstützung von Ontologien mit einer komplexeren Struktur ist im Rahmen dieser Arbeit, aufgrund des dafür benötigten zusätzlichen Aufwands, nicht möglich.

Eine Ontologie in xScraper verfügt über einen Namen und über eine Bezeichnung für die Objekte, die sie beschreibt. Des Weiteren enthält eine Ontologie eine Liste mit beliebig viel Attributen. Attribute haben die folgende Struktur: Name, Erkennungsmuster und Kardinalität. Der Name dient einer eindeutigen Identifizierung eines Attributs. Ein Erkennungsmuster in Form eines regulären Ausdrucks ermöglicht die Beschreibung von typischen Inhalten eines Attributs. Die Kardinalität legt fest, welche Attribute ein Objekt mindestens haben muss, damit dieses Objekt als valides gilt. Die Anwendung xScraper unterstützt zwei Kardinalitäten: „0 bis 1“ für die optionale Attribute und „genau 1“ für die notwendigen Attribute.

Im xScraper erfolgt die Implementierung von Ontologien durch die Klasse *DataSchema*. Sie enthält die Eigenschaften *Name*, *ObjectName* und die Liste *ObjectAttributes*. Jedes Attribut hat die Eigenschaften *Name*, *ExtractionPattern* und *IsObligatory*. Die Eigenschaft *IsObligatory* ist ein Flag, das angibt, ob ein Attribut notwendig ist.

Benutzer können Ontologien direkt in xScraper erstellen und anpassen, hierfür gibt es einen Dialog *Data Schema Editor*. Abbildung 18 zeigt die Ontologie „Book“ für die Beschreibung von Büchern, die in der Evaluation eingesetzt wurde.

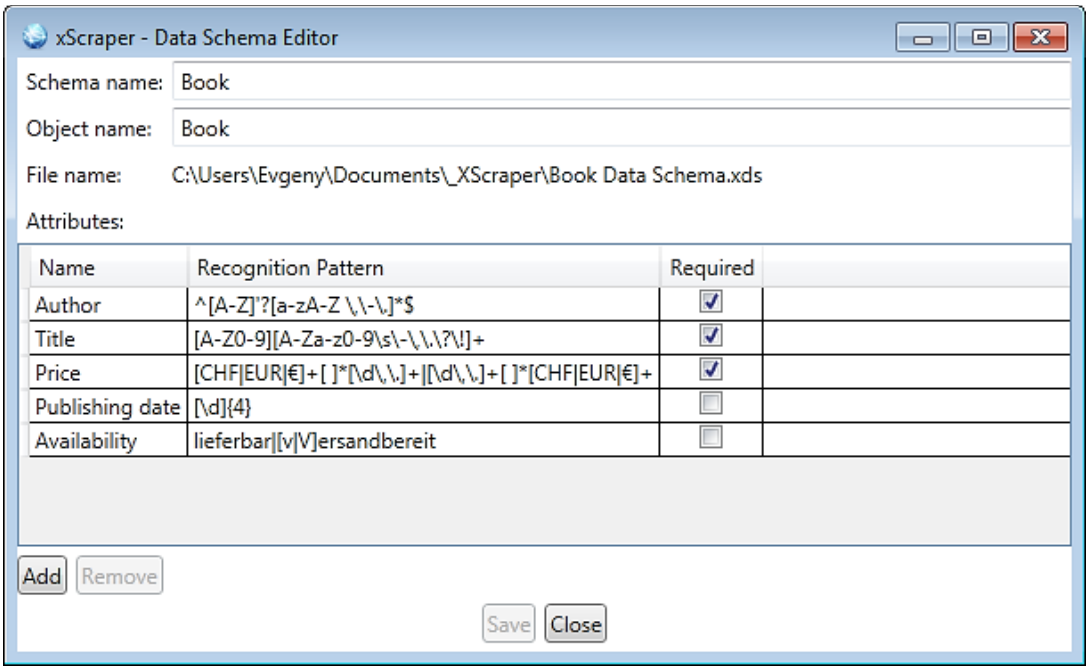


Abbildung 18. Ontologie für das Objekt "Buch"

Eine Ontologie wird im XML-Format gespeichert. Abschnitt 9.1 des Anhangs zeigt die innere Struktur der Ontologie „Book“.

4.8 Benutzung der Anwendung

Das Tool xScraper ist eine .NET Anwendung für Microsoft Windows. Ihre Ausführung erfolgt durch das Anklicken der Datei xscraper.exe.

Beim ersten Start legt xScraper die Konfigurationsdatei xscraper.cfg an. Diese Datei enthält die Benutzereinstellungen sowie den Dateinamen des zuletzt verwendeten Scrapingszenarios.

Sobald xScraper das erste Mal gestartet wird, öffnet sich die Hauptansicht des Programms (siehe Abbildung 19). Es wird kein Szenario automatisch erstellt: Der Benutzer kann ein neues Szenario erstellen oder ein existierendes Szenario laden.

4.8.1 Ein neues Scrapingszenarios erstellen

Für das Erstellen eines neuen Scrapingszenarios wählt der Benutzer den Befehl „New“ aus dem Hauptmenü „File“. Die Anwendung xScraper erstellt ein neues Szenario und

füllt es mit den grundlegenden Aktionen (siehe Abbildung 20). Jetzt hat der Benutzer die Möglichkeit, einzelne Aktionen zu konfigurieren und neue Aktionen hinzuzufügen.

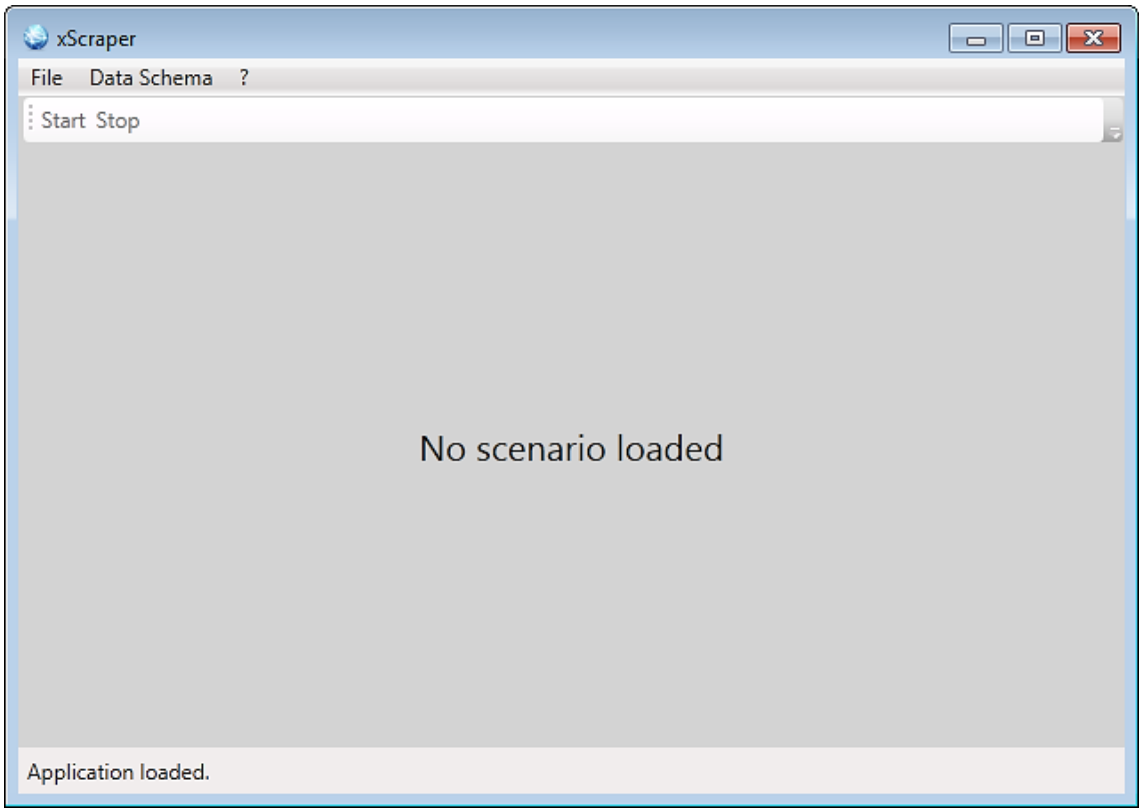


Abbildung 19. Hauptansicht von xScraper ohne Szenario

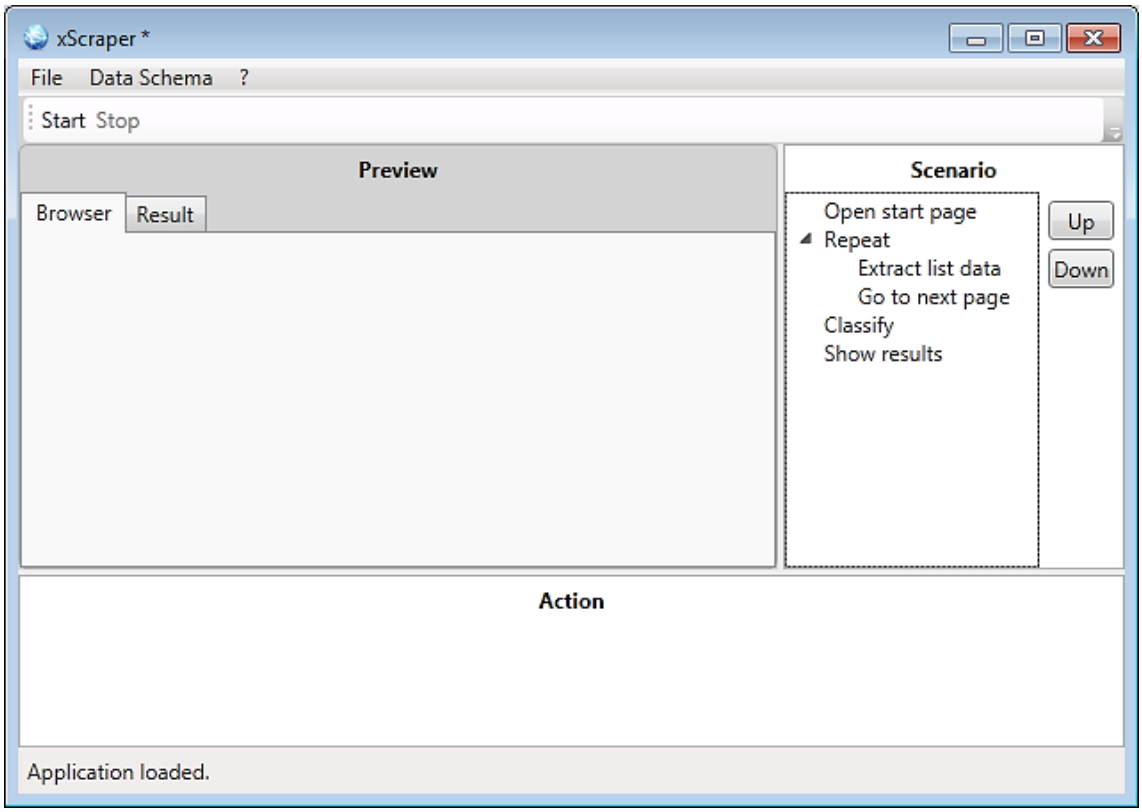


Abbildung 20. Ein neues Scrapingszenario

4.8.2 Aktionen einfügen und löschen

Das Einfügen einer neuen Aktion erfolgt über das Kontextmenü. Der Benutzer klickt eine der Aktionen aus der Liste mit der rechten Maustaste an und wählt eine Aktion aus dem Kontextmenü aus. Die neue Aktion wird direkt nach der angeklickten Aktion eingefügt. Jetzt kann sie konfiguriert werden.

Es ist zu beachten, dass nur die Aktionen für die Navigation eingefügt werden können. Diese Aktionen sind: „Click“, „Input text“ und „Wait“.

Eine weitere Einschränkung der prototypischen Implementierung ist die fehlende Überprüfung der Korrektheit eines Szenarios. Der Benutzer kann durch das Einfügen und durch das Löschen von Aktionen ein Szenario derart verändern, dass seine Ausführung nicht mehr möglich ist.

Der Benutzer kann wählen, ob xScraper die extrahierten Daten von der Klassifikation („raw extracted data“) oder die Daten nach der Klassifikation („classified data“) anzeigt. Die gewählten Daten erscheinen in einer Tabelle (siehe Abbildung 22).

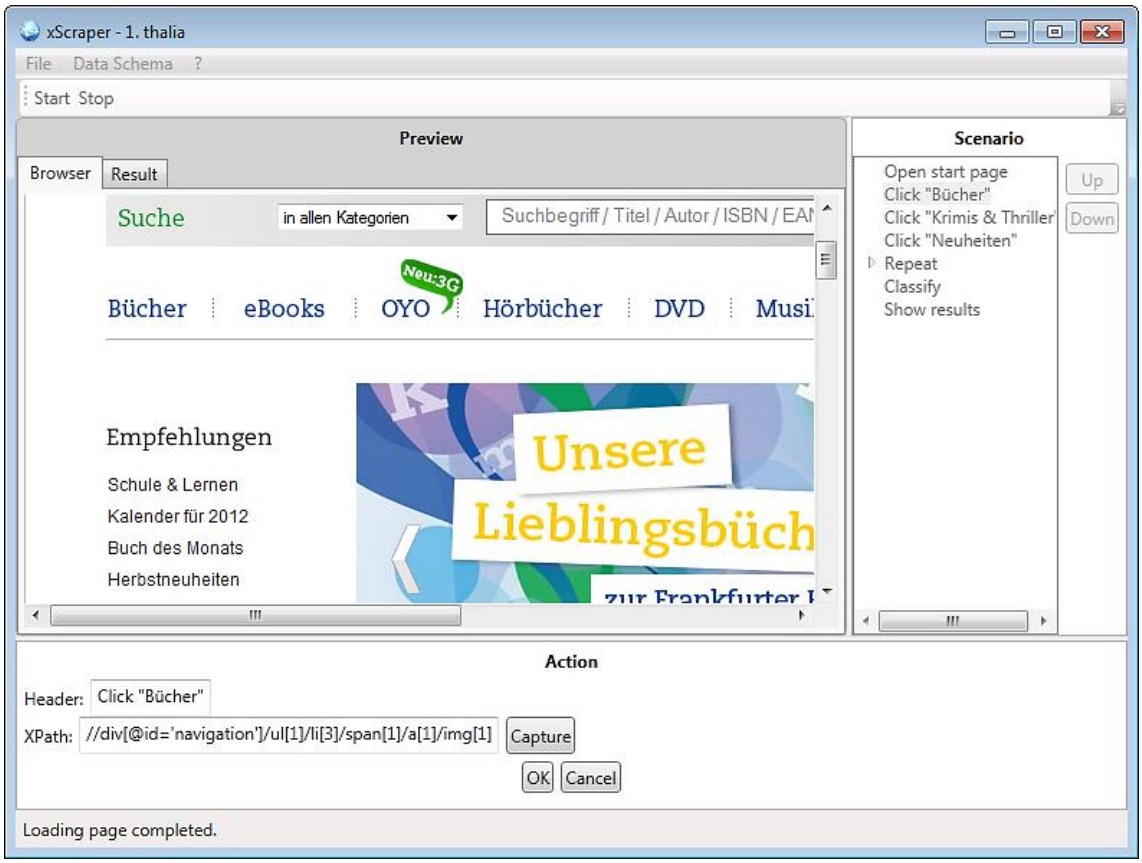


Abbildung 21. Bearbeiten einer Aktion

Das Löschen einer Aktion aus dem Szenario erfolgt auch über das Kontextmenü. Hierfür klickt der Benutzer die zu entfernende Aktion mit der rechten Maustaste an und wählt den Befehl „Delete“ aus dem Kontextmenü aus. Der Benutzer kann nur die Akti-

onen entfernen, die er selbst zuvor eingefügt hat. Die grundlegenden Funktionen können nicht aus einem Szenario entfernt werden.

4.8.3 Aktionen konfigurieren

Nach dem Erstellen eines neuen Szenarios sind die Aktionen noch nicht konfiguriert. Um eine Aktion zu konfigurieren, muss der Benutzer diese Aktion mit einem Doppelklick öffnen. Dann lädt xScraper das Dialogpanel für die ausgewählte Aktion in den unteren Bereich der Hauptansicht (siehe Abbildung 21). Der Inhalt dieses Dialogpanels hängt davon ab, welche Aktion ausgewählt wurde. Die Tasten „OK“ und „Cancel“ sind auf allen Dialogpanels vorhanden.

4.8.4 Szenario ausführen

Sobald ein Scrapingszenario konfiguriert wurde, kann der Benutzer dieses Szenario ausführen. Hierfür klickt er die Taste „Start“ in der Symbolleiste. xScraper erstellt einen neuen Ausführungskontext und führt die Aktionen der Reihe nach aus, bis alle Aktionen ausgeführt werden oder bis ein Fehler auftritt. Die Aktion, die gerade ausgeführt wird, wird dabei in der Liste der Aktionen grün markiert (siehe Abbildung 22). So kann der Benutzer jederzeit wissen, wie der Fortschritt der Ausführung ist.

Der Benutzer hat eine Möglichkeit, die Ausführung eines Szenarios abzubrechen. Dafür muss er während der Ausführung die Taste „Stop“ in der Symbolleiste anklicken.

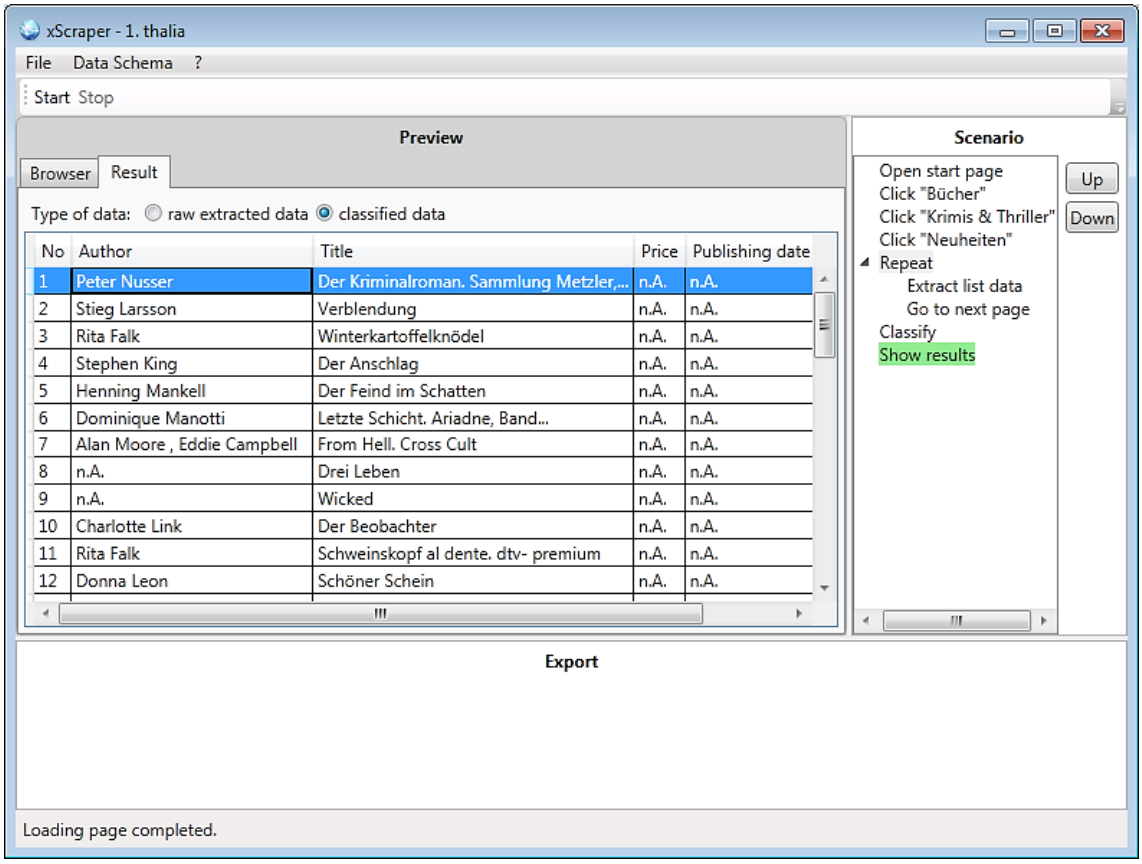


Abbildung 22. Anzeige der extrahierten Daten

4.8.5 Extrahierte Daten ansehen und speichern

Nach der Ausführung eines Szenarios enthält der Ausführungskontext die extrahierten Daten. Diese Daten stehen unter dem Lesezeichen „Result“ zur Verfügung.

Im Kontextmenü zu der Tabelle stehen dem Benutzer zwei Befehle zur Verfügung: „Copy“ und „Export“. Der Befehl „Copy“ erstellt eine Kopie der ausgewählten Zeilen der Tabelle und stellt diese Kopie über die Zwischenablage den anderen Anwendungen zur Verfügung. Der Befehl Export ermöglicht die Ausgabe der gesamten Tabelle in eine Textdatei im CSV-Format.

4.8.6 Ein Szenario speichern bzw. laden

Der Benutzer kann jederzeit die Änderungen in einem Szenario speichern. Dafür wählt er den Befehl „Save“ oder „Save As…“ aus dem Hauptmenü „File“ aus. Dabei wird zunächst die alte Kopie des Szenarios auf der Festplatte umbenannt, falls vorhanden. Dies ermöglicht das Wiederherstellen der alten Version eines Szenarios falls die Änderungen nicht zum gewünschten Ergebnis geführt haben.

Das Laden eines Szenarios erfolgt mit dem Befehl „Open“ aus dem Hauptmenü „File“.

4.8.7 Eine Ontologie erstellen und bearbeiten

Eine neue Ontologie für Datenklassifikation kann mit dem Befehl „New“ im Hauptmenü „Data Schema“ erstellt werden. Um eine bereits existierende Ontologie zu laden, wählt der Benutzer den Befehl „Open“ aus dem gleichen Hauptmenü.

Das Bearbeiten einer Ontologie erfolgt im *Data Schema Editor* (siehe Abschnitt 4.7). Hier kann der Benutzer den Namen der Ontologie und die Bezeichnung der zu beschreibenden Objekte eingeben, und die Liste der Attribute vervollständigen.

Nach der Bearbeitung kann eine Ontologie gespeichert werden, hierfür klickt der Benutzer die Taste „Save“ an.

4.8.8 Einstellungen eines Szenarios bearbeiten

Jedes Szenario kann Einstellungen enthalten, die nur für dieses Szenario gelten. In der prototypischen Anwendung xScraper haben diese Einstellungen den einzigen Parameter: Die Verzögerung zwischen der Eingabe von Text in Textfelder (siehe Abbildung 23).

Bei den meisten Websites funktioniert die Interaktion mit dem Startwert von 500 Millisekunden. Einige Websites brauchen jedoch mehr Zeit, z.B. um mit Ajax einen Datenaustausch mit dem Server durchzuführen. Für solche Websites kann der Wert der Verzögerung erhöht werden, damit alle Eingaben von der Website korrekt interpretiert werden können.

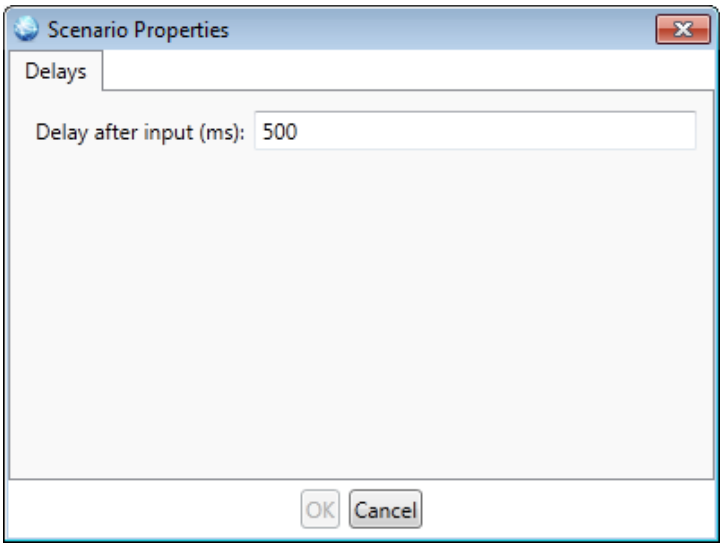


Abbildung 23. Einstellungen eines Szenarios

Um die Einstellungen eines Szenarios zu bearbeiten, wählt der Benutzer den Befehl „Properties“ aus dem Hauptmenü „File“ aus.

4.9 Fehlerbehandlung

Fehlerbehandlung ist ein wichtiger Aspekt jeder Anwendung. Da es sich bei xScraper jedoch um eine prototypische Implementierung handelt, wurde die Fehlerbehandlung auf das Wesentliche begrenzt. Die Berücksichtigung aller möglichen Angaben eines Benutzers, um eine defensive Strategie für die Fehlerbehandlung zu konzipieren, war in Rahmen dieser Arbeit nicht möglich.

Falls ein Fehler während der Ausführung eines Scrapingszenarios auftritt, wird die Ausführung abgebrochen. Ein solcher Fehler kann verschiedene Gründe haben, die Webseite kann beispielsweise nicht erreicht werden oder der eingegebene XPath-Ausdruck für die Extraktion von Daten kann fehlerhaft sein. Wenn ein Fehler passiert, zeigt xScraper eine Mitteilung mit möglichst genauen Informationen über den Grund des Fehlers an.

5 Evaluation der Methodik

Dieses Kapitel beschreibt die Evaluation der Methodik und stellt ihre Ergebnisse vor. Nach der Beschreibung des Evaluationsverfahrens folgen die Kriterien der Evaluation. Danach werden die Websites für die Evaluation der Methodik mit Produkt- und Adressdaten ausgewählt. Im Anschluss folgen die Ergebnisse der Evaluation in tabellarischer Form mit Kommentaren.

Das Ziel der Evaluation ist zu prüfen, in wie weit sich die Methodik EH für die Extraktion von Produkt- und Adressinformationen eignet. Insgesamt werden fünfzig repräsentative Websites aus einer Menge von Adress- und Produktwebsites ausgewählt, um die Extraktion und die Klassifikation von Inhalten aus Webseiten zu testen. Das Evaluationsverfahren basiert auf den Elementen von experimentellen Untersuchungen aus [33] und [35].

Die Evaluation von Websites mit Produktdaten erfolgt für drei ausgewählte Produkte: Handys, Krimis und Flugkarten. Eine Flugkarte ist zwar kein Produkt, sondern eine Dienstleistung, im Wesentlichen verfügt sie jedoch über alle Eigenschaften eines Produkts, die für die Extraktion und Klassifikation relevant sind. Die Auswahl dieser Produkte erfolgte zufällig mit dem Ziel, möglichst viele unterschiedliche Websites abzudecken.

Im Bereich der Adressdaten werden für die Evaluation die Adressen von Pizzalieferanten und die Adressen der Filialen von großen Handelsunternehmen verwendet. Die Auswahl genau dieser Informationen liegt an der großen Anzahl von Websites, die in diesem Bereich zur Verfügung stehen. Die nachfolgende Tabelle 4 zeigt zusammengefasst die Art der ausgewählten Websites, die gesuchten Inhalte und den abgedeckten Zielbereich.

Nr.	Gruppe	Zielbereich	Anzahl von Websites	Art von Websites	Gesuchte Inhalte
1	Bücher	Produktdaten	10	Online-Buchhandlungen	Krimipreise
2	Mobiltelefone	Produktdaten	10	Online-Shops	Handypreise
3	Flüge	Produktdaten	10	Webauftritte von Fluggesellschaften	Flugkartenpreise
4	Pizzalieferanten	Adressdaten	10	Online-Telefonbücher	Adressen von Pizzalieferanten
5	Filialen	Adressdaten	10	Webauftritte von Handelsunternehmen	Adressen von Filialen

Tabelle 4. Die Auswahl der Websites für die Evaluation

In jeder Gruppe werden zehn Websites getestet. Die Auswahl von Websites innerhalb jeder Gruppe enthält die wichtigsten Websites der Branche (je nach Gruppe die Websites von Unternehmen mit dem größten Umsatz oder die Websites mit der höchsten Trefferquote in Suchmaschinen). Die Einzelheiten hierfür werden im Abschnitt 5.2 erklärt.

Für die Klassifikation von Inhalten benötigt die Methodik EH Ontologien. Deshalb wird für jede Gruppe eine eigene Ontologie bereitgestellt. Die verwendeten Ontologien werden im Abschnitt 5.3 vorgestellt.

Der Vorgang beim Testen der Methodik EH ist für alle Websites gleich. Die einzelnen Schritte dieses Vorgangs sind:

1. Das Erstellen eines Scrapingszenarios
2. Die erste Ausführung des Szenarios
3. Das Anpassen des Szenarios
4. Die Ausführung des angepassten Szenarios
5. Die Auswertung der extrahierten Inhalte

Zunächst wird für jede Website ein Scrapingszenario erstellt. Am Anfang enthält das Szenario lediglich die Eingangsseite der Website. Dann wird das Szenario um die Navigationsschritte ergänzt, um zu der Zielseite mit den zu extrahierenden Inhalten zu gelangen. Falls die Zielseite eine Paginierung enthält, wird die Navigation angepasst, um insgesamt die Inhalte von höchstens fünf Webseiten einzulesen. Anschließend wird eine Ontologie für die Klassifikation angegeben.

Nachdem die Konfiguration des Szenarios abgeschlossen ist, erfolgt seine erste Ausführung. Dabei versucht ein Algorithmus die zu extrahierenden Inhalte automatisch zu erkennen und dann zu klassifizieren. Falls beides erfolgreich ist, können die extrahierten Inhalte ausgewertet werden. In der Regel wird jedoch eine Anpassung des Szenarios notwendig, z.B. die Anpassung der zu extrahierenden Textblöcke aus einer Webseite oder die Korrektur der Zuordnung von extrahierten Inhalten zu den Elementen einer Ontologie.

Das Anpassen des Szenarios und die darauffolgende Ausführung werden so oft wiederholt bis die zu entnehmenden Inhalte erfolgreich erkannt, extrahiert und klassifiziert sind oder bis es klar wird, dass dies mit den vorhandenen Mitteln nicht möglich ist.

Am Ende erfolgt die Auswertung der extrahierten Inhalte. Dabei werden verschiedene Kriterien, die im Abschnitt 5.1 erklärt werden, eingesetzt.

5.1 Kriterien der Evaluation

Die Evaluation soll verschiedene Aspekte der Methodik EH bewerten. Diese Aspekte lassen sich von den einzelnen Stufen des in der Methodik verwendeten Prozesses ableiten: die Navigation, die Datenextraktion und die Datenklassifikation (siehe Abschnitt 3.2).

Bei der Navigation wird geprüft, ob alle Zielwebseiten erreicht werden können. Das Kriterium „Navigation“ kann folgende Werte annehmen:

- „+“ falls alle Zielseiten erfolgreich erreicht werden,
- „+/-“ falls nur ein Teil der Zielseiten erreicht wird,
- „-“ falls keine Zielseite erreicht werden kann.

Im Bereich der Datenextraktion beschreibt das Kriterium „Erkennung“, wie die Inhalte aus einer Webseite extrahiert wurden. Der Wertebereich dieses Kriteriums besteht aus:

- „A“ falls die automatische Erkennung von Inhalten ohne Korrekturen erfolgt,
- „K“ falls die automatische Erkennung von Inhalten mit Korrekturen erfolgt,
- „M“ falls die automatische Erkennung keine ausreichenden Ergebnisse liefern kann und die manuelle Analyse der Webseite notwendig ist.

Des Weiteren bietet der Bereich der Datenextraktion zwei qualitative Kriterien: „Precision“ und „Recall“ [38]. Das Kriterium Precision (Genauigkeit) zeigt, wie viele der extrahierten Daten relevant sind. Das Kriterium Recall (Trefferquote) zeigt dagegen, wie viele der relevanten Daten extrahiert sind. Der Wertebereich beider Kriterien liegt zwischen 0% und 100%.

Für die Datenklassifikation gibt das Kriterium „Zuordnung“ an, wie viele Elemente einer Ontologie den extrahierten Inhalten korrekt zugeordnet werden können.

5.2 Websites für die Evaluation

Die Auswahl von Websites für die Evaluation der Methodik ist äußerst wichtig, um zuverlässige Ergebnisse zu erhalten. Dabei muss die Liste von Websites einerseits zufällige Websites enthalten, andererseits müssen diese Websites repräsentativ für ihre Gruppe sein. Um diese Anforderungen zu erfüllen, basiert die Auswahl der Websites meist auf den frei erhältlichen Ranglisten. Aus diesen Ranglisten werden die ersten zehn Einträge ausgewählt, die die Voraussetzungen für die Evaluation erfüllen. In manchen Fällen stehen auf solchen Listen nicht die Websites, sondern die Unternehmen im Vordergrund. In dieser Situation wird mit einer Suchmaschine nach der Website des Unternehmens gesucht.

Für die Gruppe „Bücher“ werden die Websites von Unternehmen aus der Liste der 50 größten Buchhandlungen im deutschsprachigen Raum aus dem Buchreport-Magazin

verwendet[19]. Die Liste der ausgewählten Websites besteht aus den zehn ersten Online-Shops, die die Bücher in der Kategorie „Krimis“ anbieten. Die vollständige Liste der Websites dieser Gruppe ist in Tabelle 5 angezeigt.

Nr.	Website	Beschreibung
1	www.thalia.de	Online-Shop der Buchhandlung Thalia
2	www.hugendubel.de	Online-Shop der Buchhandlung Hugendubel
3	www.mayersche.de	Online-Shop der Mayersche Buchhandlung
4	www.buecher.ch	Online-Shop der Orell Füssli Buchhandlung
5	www.libro.at	Online-Shop der Libro Handelsgesellschaft
6	www.lehmanns.de	Online-Shop der Lehmanns Media
7	www.galeria-kaufhof.de	Online-Shop von Galeria Kaufhof
8	www.osiander.de	Online-Shop der Osiandersche Buchhandlung
9	www.morawa-styria.at	Online-Shop der Morawa Buch und Medien-Gruppe
10	www.buchhaus.ch	Online-Shop von Lüthy + Stocker

Tabelle 5. Websites der Gruppe "Bücher" für die Evaluation

Der Gruppe „Mobiltelefone“ liegt die Liste der größten Online-Shops Deutschlands nach Besucherzahlen zugrunde[20]. Da nicht alle Online-Shops Mobiltelefone anbieten, musste die Liste um zwei Einträge manuell ergänzt werden. Die letzten zwei Einträge wurden den Suchergebnissen für den Ausdruck „Handy ohne Vertrag“ bei Google[21] entnommen. Die gesamte Liste ist in Tabelle 6 zu finden.

Nr.	Website	Beschreibung
1	www.ebay.de	Internetauktionshaus eBay
2	www.amazon.de	Versandhaus Amazon
3	www.otto.de	Versandhaus Otto
4	www.neckermann.de	Versandhaus Neckermann
5	www.tchibo.de	Einzelhandelsunternehmen Tchibo
6	www.weltbild.de	Online-Shop der Verlagsgruppe Weltbild
7	www.conrad.de	Online-Shop von Conrad Electronic SE
8	www.lidl.de	Online-Shop der Discount-Kette Lidl
9	www.handyshop.de	Online-Shop der HTM GmbH
10	www.getmobile.de	Online-Shop der getmobile GmbH

Tabelle 6. Websites der Gruppe "Mobiltelefone" für die Evaluation

Die Website-Gruppe „Flüge“ entstand aus der Liste der Fluggesellschaften mit dem größten Fluggastaufkommen im internationalen Flugverkehr[22] (siehe Tabelle 7).

Für die Gruppe „Pizzalieferanten“ konnte keine passende Rangliste gefunden werden. Deshalb setzt sich diese Gruppe aus den Websites zusammen, die die besten Trefferquoten in der Suche nach den Schlüsselworten „Telefonbuch Pizza“ und „Branchenverzeichnis Pizza“ bei Google erzielten.

[19] http://www.buchreport.de/analysen/50_groesste_buchhandlungen.htm?no_cache=1
[20] http://de.statista.com/statistik/daten/studie/158229/umfrage/online-shops-in-deutschland-nach-besucherzahlen/
[21] www.google.de
[22] http://www.iata.org/ps/publications/Pages/wats-passenger-carried.aspx

Nr.	Website	Beschreibung
1	www.ryanair.com	Website der Billigfluggesellschaft Ryanair
2	www.lufthansa.com	Website der Fluggesellschaft Deutsche Lufthansa
3	www.easyjet.com	Website der Billigfluggesellschaft easyJet
4	www.airfrance.de	Website der Fluggesellschaft Air France
5	www.emirates.com	Website der Fluggesellschaft Emirates
6	www.britishairways.com	Website der Fluggesellschaft British Airways
7	www.klm.com	Website der Fluggesellschaft KLM
8	www.delta.com	Website der Fluggesellschaft Delta
9	www.americanairlines.de	Website der Fluggesellschaft American Airlines
10	www.cathaypacific.com	Website der Fluggesellschaft Cathay Pacific

Tabelle 7. Websites der Gruppe "Flüge" für die Evaluation

Nr.	Website	Beschreibung
1	www.dastelefonbuch.de	Telefonbuch der Deutsche Telekom Medien GmbH
2	www.dasoertliche.de	Telefonbuch der Deutsche Telekom Medien GmbH mit weiteren Partnern
3	www.klicktel.de	Telefon- und Branchenbuch der telegate MEDIA AG
4	www.goyellow.de	Telefon- und Branchenbuch der GoYellow GmbH
5	www.ixquick.com	Suchmaschine von Surfboard Holding B.V.
6	www.gelbeseiten.de	Telefonbuch der Deutsche Telekom Medien GmbH und Partnerfachverlage
7	www.yellowmap.de	Telefonbuch der YellowMap AG
8	www.gewusst-wo.de	Telefonbuch der Verlagsgruppe Beleke
9	www.branchen-info.net	Branchenbuch der fastline GmbH & Co. KG
10	www.pizza.de	Bestellvermittlung für Lieferservices pizza.de GmbH

Tabelle 8. Websites der Gruppe "Pizzalieferanten" für die Evaluation

Für die Gruppe „Filialen“ wurde die Liste der größten Lebensmittelhandelsunternehmen in Deutschland verwendet[23]. Die Websites aus dieser Gruppe sind in Tabelle 9 dargestellt.

Nr.	Website	Beschreibung
1	www.rewe.de	Website von REWE Markt GmbH
2	www.kaufland.de	Website von Kaufland Warenhandel GmbH & Co. KG
3	www.mediamarkt.de	Website von Media Markt TV-HiFi-Elektro GmbH
4	www.real.de	Website von real,- SB-Warenhaus GmbH
5	www.kik-textilien.com	Website von KiK Textilien und Non-Food GmbH
6	www.kaisers.de	Website von Kaiser’s Tengelmann GmbH
7	www.norma-online.de	Website von NORMA Lebensmittelfilialbetrieb GmbH & Co. KG
8	www.dm-drogeriemarkt.de	Website von dm-drogerie markt GmbH + Co. KG
9	www.netto-online.de	Website von Netto Marken-Discount AG & Co. KG
10	www.hit.de	Website von HIT Handelsgruppe GmbH & Co. KG

Tabelle 9. Websites der Gruppe "Filialen" für die Evaluation

[23] http://www.lebensmittelzeitung.net/business/handel/rankings/pages/Top-30-LEH-Deutschland-2011_165.html#rankingTable

5.3 Die zu extrahierenden Inhalte und Ontologien

Die Websites für die Evaluation lassen sich in fünf Gruppen einordnen (siehe Tabelle 4). Alle Websites innerhalb einer Gruppe bieten vergleichbare Informationen, z.B. die Buchpreise, wobei der Umfang dieser Informationen auf jeder Website unterschiedlich ist. Für die Evaluation ist es jedoch notwendig, dass ein Mindestmaß an Daten festgelegt wird, die auf jeder Website vorhanden sind. Dieser Abschnitt definiert den Umfang von Informationen, die für jede Gruppe von Websites extrahiert werden. Mit diesen Informationen wird anschließend eine Ontologie erstellt, die für die gesamte Gruppe von Websites verwendet wird.

Die für die Datenklassifikation benutzten regulären Ausdrücke wurden so gewählt, um möglichst viele der Zielinhalte zu erkennen. Da beispielsweise Buchpreise in verschiedenen Währungen angegeben werden, wurden diese Währungen in den regulären Ausdruck übernommen.

Die Gruppe „Bücher“ besteht aus den Websites von Online-Buchhandlungen. Die zu extrahierenden Informationen sind die Buchpreise. Für die Extraktion und die Klassifikation von Daten werden die folgenden Informationen verwendet: Autor oder Autoren, Titel des Buchs, Preis, Ausgabejahr und Lieferstatus.

Der Preis, das Ausgabejahr und der Lieferstatus sind spezifische Daten, sie lassen sich mit einem regulären Ausdruck sehr genau beschreiben. Bei den Autoren und beim Buchtitel handelt es sich dagegen um Freitext mit wenigen Merkmalen, dadurch ist die Erkennung dieser Informationen problematisch. Die für diese Gruppe benutzte Ontologie ist in Tabelle 10 angezeigt.

Nr.	Information	Bezeichnung in der Ontologie	Regulärer Ausdruck	Notwendig
1	Autor oder Autoren	Author	^[A-Z]'?[a-zA-Z \,\-\.]*$	ja
2	Buchtitel	Title	[A-Z0-9][A-Za-z0-9\s\-\,\.\?\!]+	ja
3	Preis	Price	[CHF|EUR|€]+[ ]*[\d\,\.]+|[\d\,\.]+[ ]*[CHF|EUR|€]+	ja
4	Ausgabejahr	Publishing date	[\d]{4}	nein
5	Lieferstatus	Availability	lieferbar|[v|V]ersandbereit	nein

Tabelle 10. Die Ontologie für die Klassifikation von Büchern

Für die Gruppe „Mobiltelefone“ werden die folgenden Informationen extrahiert: Bezeichnung des Modells, Hersteller, Preis und Lieferstatus. Tabelle 11 enthält die für die Klassifikation von Mobiltelefonen verwendete Ontologie.

Nr.	Information	Bezeichnung in der Ontologie	Regulärer Ausdruck	Notwendig
1	Modell	Model	[A-Za-z0-9\,\.\-]+	ja
2	Hersteller	Manufacturer	[A-Za-z]+	nein
3	Preis	Price	[CHF|EUR|€]+[ ]*[\d\,\.]+|[\d\,\.]+[ ]*[CHF|EUR|€]+	ja
4	Lieferstatus	Shipment status	lieferbar|Kostenlose Lieferung	nein

Tabelle 11. Die Ontologie für die Klassifikation von Mobiltelefonen

Die Gruppe „Flüge“ setzt sich aus den Webauftritten von Fluggesellschaften mit der Möglichkeit einer Online-Buchung zusammen. Bei der Extraktion von Flugpreisen werden die folgenden Informationen erhoben: Abflugort, Ankunftsort, Abflugdatum, Abflugzeit, Fluggesellschaft und Flugpreis. Die Ontologie für die Klassifikation von Flugpreisen ist in Tabelle 12 angezeigt.

Nr.	Information	Bezeichnung in der Ontologie	Regulärer Ausdruck	Notwendig
1	Abflugort	Departure	^[A-Z]+[\D]{2,}	ja
2	Ankunftsort	Destination	^[A-Z]+[\D]{2,}	ja
3	Abflugdatum	Date	\d{2}\.\d{2}\.\d{4}	nein
4	Abflugzeit	Time	\d{2}:\d{2}	nein
5	Fluggesellschaft	Airline	^[A-Z]+[\D]{2,}	ja
6	Preis	Price	[EUR|€|$\|£]+[ ]*[\d\,\.]+\|[\d\,\.]+[ ]*[EUR\|€\|$|£]+	ja

Tabelle 12. Die Ontologie für die Klassifikation von Flügen

Die Gruppe „Pizzalieferanten“ besteht aus den Online-Telefonbüchern und den Branchenverzeichnissen. Beim Suchen nach einem Pizzalieferanten sind die folgenden Informationen relevant: Firmenname, Anschrift, Telefonnummer und E-Mail-Adresse. Tabelle 13 enthält die Ontologie für die Klassifikation von Pizzalieferanten.

Nr.	Information	Bezeichnung in der Ontologie	Regulärer Ausdruck	Notwendig
1	Firmenname	Name	[\D]+[[\D]+]*	ja
2	Anschrift	Address	[[\w\s\d\.]*\,?\s?[\d]{5}\s[\w]+	ja
3	Telefonnummer	Phone	^\(?(\+?[\d|\s]+)?\)?(\s|-)?(\(\d+\))?(\d|\s|-){5,}	nein
4	E-Mail	E-mali	[a-z0-9!#$%&'*+/=?^_`{\|}~-]+(?:\.[a-z0-9!#$%&'*+/=?^_`{|}~-]+)*@(?:[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\.)+[a-z0-9](?:[a-z0-9-]*[a-z0-9])?	nein

Tabelle 13. Die Ontologie für die Klassifikation von Pizzalieferanten

Die Gruppe „Filialen“ enthält Websites von Handelsunternehmen, die über ein Filialnetz verfügen. Aus den Websites dieser Gruppe werden die folgenden Informationen extrahiert: Anschrift, Telefonnummer und Öffnungszeiten. Die für diese Gruppe benutzte Ontologie ist in Tabelle 14 abgebildet.

Nr.	Information	Bezeichnung in der Ontologie	Regulärer Ausdruck	Notwendig
1	Anschrift	Address	[[\w\s\d\.\-ßäöü]*\,?\s?[\d]{5}\s[\w]+	ja
2	Telefonnummer	Phone	\(?[\+?[\d|\s]+]?\)?[\s|-|//]?[\(\d{3,}\)]?[\d|\s]{5,}	ja
3	Öffnungszeiten	Opening hours	.*Uhr.*	nein

Tabelle 14. Die Ontologie für die Klassifikation von Filialen

5.4 Ergebnisse der Evaluation

Die Ergebnisse der Evaluation für jede Website-Gruppe werden in Tabellen zusammengefasst. Jede Tabelle enthält die Kriterien aus dem Abschnitt 5.1 und besteht aus den folgenden Spalten:

- „Nr.“: Laufende Nummer
- „Website“: Website-Adresse
- Kriterium „Navigation“
- „Zielseiten“ zeigt an, wie viele Webseiten für die Extraktion von Daten benutzt werden (in der Regel eine Webseite, bei Paginierung maximal fünf Webseiten). Der erste Wert ist die Zahl der erfolgreich besuchten Webseiten, der zweite Wert ist der die Zahl der vorhandenen Webseiten.
- Kriterium „Erkennung“
- „Anzahl der Objekte“ enthält die Zahl der vorhandenen Objekte auf allen zu extrahierenden Webseiten
- „Extrahierte Objekte“ zeigt, wie viele Objekte während der Extraktion erkannt wurden
- „Relevante Objekte“ gibt die Anzahl der Objekte an, die für den Benutzer interessant (relevant) sind
- „Falsche Objekte“ enthält die Zahl der falsch erkannten Inhalte, die keine relevanten Objekte sind
- Kriterium „Recall“
- Kriterium „Precision“
- Kriterium „Zuordnung“

Neben den ermittelten Werten für jede Website enthält jede Tabelle eine zusammenfassende Zeile „Gesamt“, die die Summen bzw. die Mittelwerte einzelner Kriterien bietet.

Die für die Gruppe „Bücher“ ermittelten Werte sind in Tabelle 15 angezeigt. Mit der Ausnahme von einer Website hat die Navigation bei allen Websites problemlos funktioniert. Bei „www.buecher.ch“ konnte der Übergang zwischen den Webseiten einer Paginierung nicht erkannt werden. Der Webseitenwechsel erfolgt zwar, wird aber durch den Browser nicht erkannt. Nach einiger Zeit kommt es dann zu einem Timeout, und die Navigation wird abgebrochen.

In zwei Fällen hat der Algorithmus von xScraper die strukturellen Ähnlichkeiten automatisch und korrekt erkannt, sodass keine Anpassungen notwendig waren. Interessanterweise war in diesen Fällen auch die automatische Zuordnung von Inhalten zu den Elementen der Ontologie erfolgreich. Die korrekten Daten wurden somit bereits nach der Angabe einer Ontologie und nach der Anpassung der Navigation erhalten.

In den meisten Fällen war jedoch eine Anpassung der automatisch ermittelten Inhalte notwendig. Die Inhalte konnten zwar automatisch identifiziert werden, der generierte XPath-Ausdruck referenzierte aber nur einen Teil dieser Inhalte, z.B. nur eine von mehreren Spalten in einer Tabelle.

Wie aus Tabelle 15 folgt, hat xScraper die meisten zu extrahierenden Objekte erkannt. Die fehlenden Objekte bei „www.hugendubel.de“ gehen auf die unterschiedlichen Vorlagen für die Erstellung von Listen zurück. Die meisten Inhalte hatten die gleiche Struktur, bei den Sonderangeboten wurden jedoch zusätzliche Elemente verwendet. Diese zusätzlichen Elemente haben die Struktur der Inhalte derart verändert, dass die für die Klassifikation entscheidenden Inhalte wie etwa der Preis an einer anderen Stelle waren. Dies hat zur Folge, dass solche Objekte bei der Klassifizierung aussortiert werden, weil sie nicht die Voraussetzungen der verwendeten Ontologie erfüllen.

Nr.	Website	Navigation	Zielseiten	Erkennung	Anzahl der Objekte	Extrahierte Objekte	Relevante Objekte	Falsche Objekte	Recall	Precision	Zuordnung
1	www.thalia.de	+	5/5	K	50	50	50	0	100%	100%	2/2
2	www.hugendubel.de	+	5/5	K	41	35	35	0	85%	100%	3/4
3	www.mayersche.de	+	5/5	K	75	75	75	0	100%	100%	3/5
4	www.buecher.ch	+/-	1/5	K	10	10	10	0	100%	100%	5/5
5	www.libro.at	+	5/5	A	60	60	60	0	100%	100%	2/2
6	www.lehmanns.de	+	5/5	K	100	100	100	0	100%	100%	2/3
7	www.galeria-kaufhof.de	+	1/1	K	8	8	8	0	100%	100%	0/3
8	www.osiander.de	+	1/1	A	19	19	19	0	100%	100%	4/4
9	www.morawa-styria.at	+	1/1	K	10	10	10	0	100%	100%	2/4
10	www.buchhaus.ch	+	5/5	K	50	50	50	0	100%	100%	3/5
Gesamt		+: 9 +/-: 1 -: 0		A: 2 K: 8 M: 0	423	417	417	0	98,6%	100,0%	70,3%

Tabelle 15. Ergebnisse der Evaluation für die Website-Gruppe "Bücher"

Bei der Klassifikation von Inhalten konnte über die Hälfte der in den extrahierten Objekten vorhandenen Elemente der Ontologie korrekt erkannt werden. Die meisten korrekt erkannten Elemente sind der Preis und das Ausgabejahr. Besonders problematisch ist die Erkennung in den Fällen, wenn sich zwei Elemente einer Ontologie zusammengefasst innerhalb eines HTML-Tags befinden, wie im folgenden Beispiel aus eBay: „Die dunkle Seite von Frank Schätzing“. In einer solchen Situation wird der ganze Inhalt als Buchtitel oder als Autor erkannt und es ist eine manuelle Anpassung notwendig.

In der Gruppe „Mobiltelefone“ gab es bei zwei Websites Probleme mit der Navigation (siehe Tabelle 16). Genau wie in der vorherigen Gruppe handelt es sich um den Webseitenwechsel.

Bei einer Website konnten die Inhalte vollkommen automatisch identifiziert werden. Die automatische Erkennung schlug hier ebenfalls in einem Fall fehl. Im Rest der Fälle erfolgten Anpassungen des automatisch generierten XPath-Ausdrucks.

Die Website „www.otto.de“ verwendet verschiedene Vorlagen für Normal- und Sonderangebote, deshalb konnten für diese Website nicht alle Inhalte extrahiert werden.

Bei der automatischen Zuordnung von Spalten wurden der Preis und der Lieferstatus in den meisten Fällen korrekt erkannt, die Spalten für den Hersteller und für die Modellbezeichnung wurden oft falsch interpretiert. Ähnlich wie bei Büchern, befinden sich auch bei Mobiltelefonen der Hersteller und das Modell in einem Satz. Eine automatische Erkennung ist in einem solchen Fall nur schwer möglich.

Nr.	Website	Navigation	Zielseiten	Erkennung	Anzahl der Objekte	Extrahierte Objekte	Relevante Objekte	Falsche Objekte	Recall	Precision	Zuordnung
1	www.ebay.de	+	5/5	K	252	252	252	0	100%	100%	1/3
2	www.amazon.de	+	5/5	K	120	120	120	0	100%	100%	3/4
3	www.otto.de	+	5/5	K	75	60	60	0	80%	100%	3/3
4	www.neckermann.de	+/-	1/5	K	20	20	20	0	100%	100%	4/4
5	www.tchibo.de	+	1/1	M	6	6	6	0	100%	100%	2/3
6	www.weltbild.de	+	1/1	K	6	6	6	0	100%	100%	4/4
7	www.conrad.de	+	5/5	K	100	100	100	0	100%	100%	2/4
8	www.lidl.de	+	1/1	A	36	36	36	0	100%	100%	0/3
9	www.handyshop.de	+/-	1/3	K	25	25	25	0	100%	100%	2/4
10	www.getmobile.de	+	1/1	K	59	59	59	0	100%	100%	2/3
Gesamt		+: 8 +/-: 2 -: 0		A: 1 K: 8 M: 1	699	684	684	0	97,9%	100,0%	65,7%

Tabelle 16. Ergebnisse der Evaluation für die Website-Gruppe "Mobiltelefone"

Die Website-Gruppe „Flüge“ bot im Bereich der Navigation die größte Herausforderung. Fast jede Website dieser Gruppe verarbeitet die Eingaben des Benutzers auf ihre eigene Weise. Bei der Angabe des Abflugorts wurden z.B. die folgenden Möglichkeiten beobachtet:

- Der Benutzer kann den kompletten Namen des Flughafens angeben, z.B. „Paris (CDG)“,
- Der Benutzer kann einen Teil des Abflugorts eintippen, danach erscheint eine Auswahl von Orten und der Benutzer kann einen davon auswählen,
- Der Benutzer kann den Abflugort nur aus einer Liste auswählen,
- Der Benutzer kann den Abflugort eintippen (z.B. „Berlin“), im nächsten Schritt muss der Benutzer einen Flughafen wählen (z.B. „Berlin-Tegel“ oder „Berlin-Schönefeld“).

Mit der Flexibilität von Web-Scraping-Szenarien in xScraper konnte die Navigation in allen Fällen erfolgreich konfiguriert werden (siehe Tabelle 17).

In zwei Fällen hat die automatische Erkennung von strukturellen Ähnlichkeiten sofort die korrekten Inhalte geliefert. Eine Fluggesellschaft verwendet eine komplexe Tabelle, um die Ergebnisse der Flugkartensuche darzustellen, deshalb musste in diesem Fall eine manuelle Eingabe des XPath-Ausdrucks für die Extraktion von Daten erfolgen. Für die weiteren sieben Websites wurden lediglich einige wenige Korrekturen notwendig.

Bei den Flügen, die aus mehreren einzelnen Strecken bestehen, verwendet „www.airfrance.de“ abweichende Vorlagen. Ähnlich wie in den anderen Website-Gruppen führt es dazu, dass die betroffenen Objekte aussortiert werden.

Die automatische Zuordnung von Spalten war in dieser Gruppe problematisch. Für die spezifischen Informationen wie der Preis und die Uhrzeit konnten gute Ergebnisse erzielt werden. Die Angabe des Datums war jedoch bei jeder Fluggesellschaft anders, z.B. „1 Nov.“, „2011-11-01“ oder „Nov 01“. Besonders selten hat die automatische Zuordnung von Ab- und Ankunftsorten funktioniert.

Nr.	Website	Navigation	Zielseiten	Erkennung	Anzahl der Objekte	Extrahierte Objekte	Relevante Objekte	Falsche Objekte	Recall	Precision	Zuordnung
1	www.ryanair.com	+	1/1	A	7	7	7	0	100%	100%	1/2
2	www.lufthansa.com	+	1/1	K	44	44	44	0	100%	100%	4/5
3	www.easyjet.com	+	1/1	K	3	3	3	0	100%	100%	2/3
4	www.airfrance.de	+	1/1	M	16	14	14	0	88%	100%	4/5
5	www.emirates.com	+	1/1	K	5	5	5	0	100%	100%	2/5
6	www.britishairways.com	+	1/1	K	3	3	3	0	100%	100%	1/4
7	www.klm.com	+	1/1	K	11	11	11	0	100%	100%	2/3
8	www.delta.com	+	1/1	K	19	19	19	0	100%	100%	2/5
9	www.americanairlines.de	+	1/1	A	6	6	6	0	100%	100%	3/4
10	www.cathaypacific.com	+	1/1	K	40	40	40	0	100%	100%	1/4
Gesamt		+: 10 +/-: 0 -: 0		A: 2 K: 7 M: 1	154	152	152	0	98,7%	100,0%	55,0%

Tabelle 17. Ergebnisse der Evaluation für die Website-Gruppe "Flüge"

Die Gruppe „Pizzalieferanten“ zeigt gute Ergebnisse in der Navigation, die Erkennung von Inhalten hat jedoch die meisten Probleme vorbereitet. Wie Tabelle 18 zeigt, musste für die Extraktion von Daten bei drei Websites eine manuelle Eingabe der Zielinhalte erfolgen. Immerhin hat in zwei Fällen die automatische Erkennung von Inhalten zum Erfolg geführt. In der Hälfte aller Fälle waren Korrekturen erforderlich.

Drei Websites aus der Gruppe „Pizzalieferanten“ hatten Inhalte mit unterschiedlichen Vorlagen. Telefonbücher bieten die sogenannten „Premiumeinträge“ an, damit die betroffenen Unternehmen zusätzliche Information zu den Standarddaten wie die Adresse und die Telefonnummer hinzufügen können. Diese zusätzlichen Informationen verändern jedoch die Struktur eines Objekts, wodurch es bei der Klassifikation aussortiert wird.

Die automatische Zuordnung der E-Mail-Adresse (sofern vorhanden), der Telefonnummer und der Anschrift verlief sehr gut, bei Erkennung der Bezeichnung des Lieferanten gab es, wie erwartet, Probleme. Diese Bezeichnungen können derart unterschiedliche Formen annehmen, dass es schwierig ist, gemeinsame Regeln in Form eines regulären Ausdrucks zu definieren.

Nr.	Website	Navigation	Zielseiten	Erkennung	Anzahl der Objekte	Extrahier-te Objekte	Relevante Objekte	Falsche Objekte	Recall	Precision	Zuordnung
1	www.dastelefonbuch.de	+	5/5	A	97	97	97	0	100%	100%	3/3
2	www.dasoertliche.de	+	5/5	K	100	100	100	0	100%	100%	3/4
3	www.klicktel.de	+	5/5	K	100	95	95	0	95%	100%	3/4
4	www.goyellow.de	+	4/4	M	73	72	72	0	99%	100%	2/3
5	www.ixquick.com	+	3/3	A	28	28	28	0	100%	100%	2/3
6	www.gelbeseiten.de	+/-	1/5	K	15	12	12	0	80%	100%	2/3
7	www.yellowmap.de	+	5/5	K	75	75	75	0	100%	100%	1/3
8	www.gewusst-wo.de	+	5/5	K	50	50	50	0	100%	100%	2/3
9	www.branchen-info.net	+	1/1	M	12	12	12	0	100%	100%	2/3
10	www.pizza.de	+	1/1	M	42	42	42	0	100%	100%	2/2
Gesamt		+: 9 +/-: 1 -: 0		A: 2 K: 5 M: 3	592	583	583	0	98,5%	100,0%	71,0%

Tabelle 18. Ergebnisse der Evaluation für die Website-Gruppe "Pizzalieferanten"

In der letzten Gruppe „Filialen“ war die Navigation in den meisten Fällen erfolgreich, trotz der verschiedensten Arten der Interaktion. Manche Handelsunternehmen versuchen es, dem Benutzer eine konkrete Adresse des nahesten Geschäfts anzubieten, ohne die komplette Liste aller Filialen in der Nähe anzuzeigen. Andere Handelsunternehmen zeigen ihre Geschäfte auf einem Stadtplan an. Mit xScraper konnten alle diese Navigationsszenarien erfolgreich ausgeführt werden. Die Probleme mit der Navigation bei zwei Websites in dieser Website-Gruppe sind ähnlich, wie bei den bereits zuvor beschriebenen Schwierigkeiten.

Nr.	Website	Navigation	Zielseiten	Erkennung	Anzahl der Objekte	Extrahier-te Objekte	Relevante Objekte	Falsche Objekte	Recall	Precision	Zuordnung
1	www.rewe.de	+	1/1	M	12	12	12	0	100%	100%	2/2
2	www.kaufland.de	+	1/1	M	7	7	7	0	100%	100%	2/3
3	www.mediamarkt.de	+	1/1	K	3	3	3	0	100%	100%	2/3
4	www.real.de	+/-	1/1	M	316	316	316	0	100%	100%	1/1
5	www.kik-textilien.com	+	1/1	A	28	28	28	0	100%	100%	2/2
6	www.kaisers.de	+/-	1/5	A	5	5	5	0	100%	100%	1/1
7	www.norma-online.de	+	1/1	A	8	8	8	0	100%	100%	0/2
8	www.dm-drogeriemarkt.de	+	1/1	K	21	21	21	0	100%	100%	1/2
9	www.netto-online.de	+	1/1	A	9	9	9	0	100%	100%	1/1
10	www.hit.de	+	1/1	A	5	5	5	0	100%	100%	2/2
Gesamt		+: 8 +/-: 2 -: 0		A: 2 K: 5 M: 3	414	414	414	0	100,0%	100,0%	73,7%

Tabelle 19. Ergebnisse der Evaluation für die Website-Gruppe "Filialen"

Wie aus Tabelle 19 folgt, wurden absolut alle zu extrahierende Inhalte auf den Websites der Gruppe „Filialen“ extrahiert. Die automatische Erkennung von Inhalten war in

zwei Fällen erfolgreich, bei drei Anbietern musste die Extraktion jedoch manuell konfiguriert werden.

Bei der automatischen Zuordnung hat die Tatsache, dass die Anschrift, die Telefonnummer und die Öffnungszeiten gut strukturierte Daten sind, die entscheidende Rolle gespielt. Die Klassifikation war somit in den meisten Fällen korrekt.

Die Zusammenfassung der Ergebnisse der Evaluation für die einzelnen Website-Gruppen und der daraus berechnete Durchschnitt sind in Tabelle 20 abgebildet.

Gruppe	Navigation			Extraktion			Klassifikation		
	+	+/-	-	A	K	M	Recall	Precision	Zuordnung
Bücher	90%	10%	0%	20%	80%	0%	98,6%	100,0%	70,3%
Mobiltelefone	80%	20%	0%	10%	80%	10%	97,9%	100,0%	65,7%
Flüge	100%	0%	0%	20%	70%	10%	98,7%	100,0%	55,0%
Pizzalieferanten	90%	10%	0%	20%	50%	30%	98,5%	100,0%	71,0%
Filialen	80%	20%	0%	50%	20%	30%	100,0%	100,0%	73,7%
Durschnitt	**88%**	**12%**	**0%**	**24%**	**60%**	**16%**	**98,7%**	**100,0%**	**67,1%**

Tabelle 20. Zusammenfassung der Ergebnisse der Evaluation

Im Bereich der Navigation zeigen die Ergebnisse, dass die meisten Navigationsszenarien mit xScraper erfolgreich waren. Bei keiner einzigen der fünfzig getesteten Websites war die Navigation nicht möglich. Bei sechs Websites gab es ein Problem beim Webseitenwechsel mit Paginierung. Dieses Problem wurde nicht näher untersucht, jedoch besteht ein Verdacht, dass der Einsatz von Ajax auf Websites dieses Problem verursachen kann. Mit den restlichen Websites hat die Navigation ohne nennenswerte Probleme funktioniert.

Die automatische Erkennung von Inhalten in Form von Listen und Tabellen war in über 80% aller Fälle möglich. Bei 24% der Websites war das Ergebnis so gut, dass keine weiteren Anpassungen für die Extraktion von Daten notwendig waren. Lediglich in 16% der Fälle war die Erkennung nicht erfolgreich. Dies hat verschiedene Gründe, z.B. können die Listen und Tabellen eine heterogene interne Struktur haben, oder eine Webseite kann durch die Vielzahl von Listeninhalten (inklusive Werbung) so viele unterschiedliche iterative Strukturen enthalten, dass der Algorithmus eine falsche Liste oder Tabelle erkennt.

Es besteht also ein großes Potenzial, die automatische Erkennung von strukturell ähnlichen Tabellen zu verbessern. Die Anwendung xScraper nutzt die eindeutigen Attribute von HTML-Tags, um XPath-Ausdrucke der identifizierten ähnlichen Inhalte zu generieren, z.B. das Attribut „ID“. Ähnliche Inhalte werden jedoch oft für eine einheitliche Darstellung in Verbindung mit CSS Styles verwendet. Dies ermöglicht die Auswahl von

Elementen anhand nicht eindeutiger HTML-Attribute wie „Class“. Diese Erkenntnis könnte in der Weiterentwicklung der Methodik EH berücksichtigt werden.

Die Ergebnisse der automatischen Klassifikation spiegeln die Schwierigkeiten wieder, die mit der Beschreibung von Inhalten mittels regulärer Ausdrücke zusammenhängen. Bei manchen Inhalten führt die Verwendung von regulären Ausdrücken zu einer sehr guten Erkennungsquote, z.B. bei Telefonnummern, E-Mail-Adressen oder Anschriften. Bei anderen Inhalten, wie beispielsweise Ortsnamen oder Personennamen, ist der Einsatz von regulären Ausdrücken nicht ausreichend. Hier könnte beispielsweise durch den Einsatz von Lexika eine effizientere Erkennung erreicht werden. Es ist z.B. möglich, eine Liste aller Orte oder aller Autoren zu erstellen, und die zu identifizierenden Inhalte mit Einträgen aus diesen Listen zu vergleichen.

Nichtsdestotrotz ist die durchschnittliche Erkennungsquote von über 67% als Erfolg zu bewerten.

Die hohen Werte von Recall und Precision sind dadurch zu erklären, dass in den meisten Fällen eine Korrektur der erkannten Inhalte stattfand. Dadurch konnte erreicht werden, dass die meisten identifizierten und klassifizierten Inhalte am Ende in der Ergebnisliste waren.

Durch den Einsatz von mehreren Vorlagen für die Darstellung von zu erkennenden Inhalten kam es jedoch bei mehreren Websites dazu, dass ein Teil der identifizierten Inhalte während der Klassifikation aussortiert wurde.

Insgesamt bestätigen die Ergebnisse der Evaluation die Wirksamkeit der Methodik EH und geben Hinweise auf weitere Verbesserungsmöglichkeiten.

6 Zusammenfassung und Ausblick

Dieses Kapitel bietet einen Überblick über die Ergebnisse der Arbeit. Zunächst werden die einzelnen Abschnitte der Arbeit vorgestellt, im Anschluss folgt der Ausblick mit den Möglichkeiten der Weiterentwicklung der neuen Methodik.

6.1 Zusammenfassung

Im Hauptteil dieser Arbeit wurde eine neue Methodik für die Extraktion und Klassifikation von Daten aus Webseiten vorgestellt und erläutert. Dieses Thema umfasst mehrere Wissensbereiche, deshalb wurden zunächst die einzelnen Aspekte aus dem Bereich der Extraktion und Klassifikation von Daten erläutert. Bei der Analyse der vorhandenen Verfahren und Werkzeugen wurde festgestellt, dass keine einzige zugängliche Lösung alle Anforderungen der gestellten Aufgabe erfüllt. Die getesteten Werkzeuge decken lediglich einen Teilbereich der Anforderungen ab, außerdem zeigen sie Defizite bei den Interaktionsmöglichkeiten, was die Möglichkeiten zur Extraktion von Daten einschränkt. Die Klassifikation von Daten wird von keinem Werkzeug angeboten, weil alle Werkzeuge eine aktive Rolle des Benutzers bei der Datenextraktion voraussetzen und keine eigene Analyse von Webseiten durchführen.

Bei der Entwicklung der Methodik EH („Extraction Heuristics“) haben die erkannten Probleme und Defizite eine wichtige Rolle gespielt. Nach der Analyse der einzelnen Teilaufgaben beim Web-Scraping wurden vier Bereiche der Methodik definiert: die Navigation, die Extraktion, die Klassifikation und die Ausgabe. Die Methodik bietet einen Prozess, der diese vier Bereiche integriert und alle notwendigen Tätigkeiten vom Öffnen einer Startseite bis zum Speichern der extrahierten Informationen umschließt.

Für die Verifikation der Methodik erfolgte die Entwicklung einer prototypischen Anwendung xScraper, die die Methodik EH implementiert. Diese Anwendung realisiert den Prozess aus der Methodik EH mit Web-Scraping-Szenarien. Bei der Extraktion und der Klassifikation von Daten wurden spezielle heuristische Algorithmen eingesetzt, um in akzeptabler Zeit valide Ergebnisse zu liefern.

Nach der Fertigstellung der Anwendung xScraper fand die Evaluation statt. Dabei wurden Produkt- und Adressdaten aus einer Auswahl von Websites extrahiert und anhand von Ontologien klassifiziert. Die Ergebnisse der Evaluation haben die Wirksamkeit der Methodik bestätigt.

6.2 Ausblick

Die neue Methodik EH bietet eine Grundlage für weitere Entwicklungen. Jeder der vier Bereiche der Methodik (Navigation, Extraktion, Klassifikation und Ausgabe) lässt sich erweitern und anpassen, um neue Aufgabestellungen erfolgreich zu bewältigen.

Im Bereich der Navigation können neue Aktionen realisiert werden, um erweiterte Navigationsstrategien zu unterstützen. Bei der Evaluation wurden u. a. die Flugpreise bei mehreren Fluggesellschaften erhoben, wobei der Abflugort, der Ankunftsort und das Datum fest angegeben wurden. Hier sind weitere Anwendungsfälle denkbar, z.B. die Eingabe von Flughäfen anhand einer vorher definierten Liste oder spezielle Regeln für die Eingabe des Datums.

Ein anderes Beispiel für die Weiterentwicklung im Bereich der Navigation ist die Unterstützung eines weit verbreiteten Paradigmas: Ein Benutzer klickt auf einen Artikel in der Liste und gelangt zu der Artikelseite, die zusätzliche Informationen über den ursprünglichen Artikel anbietet. Die Möglichkeit, zusätzliche Informationen zum Artikel auf diese Weise zu gewinnen, würde den Informationsgehalt der extrahierten Daten deutlich erhöhen.

Weitere Möglichkeiten für die Ergänzung der neuen Methodik aus Sicht der Navigation beinhalten eine bessere Interaktion mit Skripten, die Unterstützung der Navigation in mehreren Browserfenstern und eine engere Bindung von Datenextraktion an die Navigation, um auf die dynamischen Inhalte zugreifen zu können.

Bei der Extraktion von Daten aus Webseiten gibt es ebenfalls einen großen Spielraum für die Weiterentwicklung. Bisher lag der Fokus der Datenextraktion auf textuellen Daten. Eine Webseite enthält aber viel mehr Informationen, z.B. Abbildungen und Links. Die Einbindung dieser Inhalte in den Prozess der Datenextraktion kann neue Anwendungsfälle ermöglichen.

Ein zentraler Erfolgsfaktor in der Extraktion von Daten ist die korrekte Abgrenzung der einzelnen Informationen eines Objekts. Die HTML bietet viele Möglichkeiten für die Gestaltung der Informationsdarstellung, die weit über die Grenzen von zweidimensionalen Tabellen, die in dieser Methodik betrachtet wurden, hinausgehen. Die Behandlung der verschiedenen Darstellungsmöglichkeiten könnte der Methodik mehr Flexibilität verleihen.

Nicht zuletzt könnten weitere Techniken wie Natural Language Processing (NLP) oder Maschinelles Lernen dazu beitragen, die Identifikation interessanter Informationen auf Webseiten zu verbessern.

Die Klassifikation von Daten aus Webseiten ist ein Bereich mit besonders vielen Herausforderungen und Möglichkeiten für die Weiterentwicklung. Die Methodik EH behandelt die Klassifikation von Daten getrennt von der Extraktion. Es ist jedoch möglich, die beiden Bereiche zu verbinden, um bessere Ergebnisse zu erzielen: Die Klassifikation der Inhalte einer Webseite könnte zur genaueren Identifikation der zu extrahierenden Informationen führen.

Der Einsatz von Ontologien für die Datenklassifikation ermöglicht eine präzise Auswahl und eine strukturierte Ausgabe der extrahierten Inhalte. Ontologien können jedoch mehr: Sie können semantische Merkmale und Beziehungen zwischen den extrahierten Objekten speichern. Außerdem können die in Ontologien enthaltenen Erkenntnisse eine bessere Klassifizierung der extrahierten Daten ermöglichen.

Auch im Bereich der Ausgabe gibt es mehrere Möglichkeiten für die Weiterentwicklung der Methodik. Die Ausgabe kann z.B. in eine relationale Datenbank erfolgen, was den Vergleich von Daten aus verschiedenen Websites erleichtern würde.

Das breite Spektrum und die verschiedenen Richtungen für eine mögliche Weiterentwicklung der Methodik EH bilden eine erfolgsversprechende und anwendungsnahe Grundlage für weitere wissenschaftliche Arbeiten.

7 Literaturliste

1. Adelberg, B. NoDoSE---a tool for semi-automatically extracting structured and semistructured data from text documents. *ACM SIGMOD Record 27*, 2 (1998), 283-294.

2. Aly, M. Survey on Multiclass Classification Methods Extensible algorithms. *Neural Networks*, 11 (2005), 1-9.

3. Arocena, G.O. and Mendelzon, A.O. WebOQL: Restructuring documents, databases and webs. *Data Engineering 1998 Proceedings 14th International Conference on*, IEEE (2002), 24–33.

4. Aulbach, A., Schmid, E., Winstead, J., et al. PHP Manual. *Environment*, (2001), 12–04.

5. Baeza-Yates, R.A. Algorithms for string searching. *ACM SIGIR Forum 23*, 3-4 (1989), 34-58.

6. Berglund, A., Boag, S., Chamberlin, D., et al. XML Path Language (XPath) 2.0. *http://www.w3.org/TR/xpath20/*. Abgerufen am: 27.06.2011.

7. Bergman, M.K. The Deep Web: Surfacing Hidden Value. *The Journal of Electronic Publishing 7*, 1 (2001).

8. Berners-Lee, T., Hendler, J., and Lassila, O. The Semantic Web. *Scientific American 284*, 5 (2001), 34-43.

9. Brickley, D. and Guha, R.V. RDF Vocabulary Description Language 1.0: RDF Schema. *http://www.w3.org/TR/rdf-schema/*. Abgerufen am: 10.08.2011.

10. Cafarella, M.J., Halevy, A., Wang, D.Z., Wu, E., and Zhang, Y. WebTables: exploring the power of tables on the web. *Proceedings of the VLDB Endowment 1*, 1 (2008), 538-549.

11. Califf, M.E. and Mooney, R.J. Relational learning of pattern-match rules for information extraction. *Proceedings Of The National Conference On Artificial Intelligence*, JOHN WILEY & SONS LTD (1999), 9-15.

12. Chang, C.-H. and Kayed, M. A Survey of Web Information Extraction Systems. *IEEE Transactions on Knowledge and Data Engineering 18 no. 19*, (2006), 1411-1428.

13. Crescenzi, V. and Mecca, G. Grammars have exceptions. *Information Systems Journal 23*, 8 (1998), 539-565.

14. Crescenzi, V., Mecca, G., and Merialdo, P. RoadRunner: Towards Automatic Data Extraction from Large Web Sites. *Very Large Data Bases*, (2001), 109-118.

15. Cunningham, H. Information Extraction, Automatic. *Science 5*, November (2005), 1-22.

16. Embley, D.W., Campbell, D.M., Jiang, Y.S., et al. Conceptual-model-based data extraction from multiple-record Web pages. *Data & Knowledge Engineering 31*, 3 (1999), 227-251.

17. Freitag, D. Machine Learning for Information Extraction in Informal Domains. *Machine Learning 39*, 2 (2000), 169-202.

18. Garrett, J.J. Ajax: A New Approach to Web Applications. *http://www.adaptivepath.com/publications/essays/archives/000385.php*. Abgerufen am: 29.11.2011.

19. Gatterbauer, W. Web Harvesting. *Encyclopedia of Database Systems*, (2009), 3472-3473.

20. Giunchiglia, F., Marchese, M., and Zaihrayeu, I. Towards a theory of formal classification. Proceedings of the AAAI-05 Workshop on Contexts and Ontologies: Theory, Practice and Applications (2005), 1-8.

21. Gruber, T.R. A Translation Approach to Portable Ontology Specifications by A Translation Approach to Portable Ontology Specifications. *Knowledge Creation Diffusion Utilization 5*, April (1993), 199-220.

22. Horrocks, I. DAML+OIL: A Reason-able Web Ontology Language. *Lecture Notes in Computer Science 2512*, (2002), 2-13.

23. Hsu, C. and Dung, M. Generating finite-state transducers for semi-structured data extraction from the Web. *Information Systems Journal 23*, 8 (1998), 521-538.

24. Huck, G., Fankhauser, P., Aberer, K., and Neuhold, E.J. Jedi: Extracting and Synthesizing Information from the Web. *COOPIS 98 Proceedings of the 3rd IFCIS International Conference on Cooperative Information Systems*, IEEE Computer Society Press (1998), 32-43.

25. Le Hégaret, P. Document Object Model (DOM). *http://www.w3.org/DOM/*. Abgerufen am: 20.08.2011.

26. James, S. Screen scraping and web harvesting: the legal issues. *e-Commerce Law and Policy*, June 2011, 13-15.

27. Kobayashi, M. and Takeda, K. Information retrieval on the web. *ACM Computing Surveys 32*, 2 (2000), 144-173.

28. Koster, M. A Standard for Robot Exclusion. *http://www.robotstxt.org/orig.html*. Abgerufen am: 02.09.2011.

29. Kushmerick, N. Wrapper Induction for Information Extraction. IJCAI (1997), Volume: telligence, Publisher: Citeseer, Pages: 729-737 (1997).

30. Kushmerick, N. Wrapper induction: Efficiency and expressiveness. *Artificial Intelligence 118*, 1-2 (2000), 15-68.

31. Laender, A., Ribeiro-Neto, B., and Da Silva, A.S. DEByE-data extraction by example. *Data & Knowledge Engineering 40*, 2 (2002), 121–154.

32. Laender, A.H.F. and Ribeiro-neto, B.A. A Brief Survey of Web Data Extraction Tools. *ACM SIGMOD Record 31*, 2 (2002).

33. Li, Z., Ng, W.K., and Sun, A. Web data extraction based on structural similarity. *Knowledge and Information Systems 8*, 4 (2005), 438-461.

34. Liddle, S., Embley, D., Scott, D., and Yau, S.H. Extracting Data Behind Web Forms. *Lecture Notes in Computer Science*, 2784 (2003), 402-413.

35. Liu, B., Grossman, R., and Zhai, Y. Mining data records in Web pages. *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, (2003), 601–606.

36. Liu, L., Pu, C., and Han, W. XWRAP: An XML-enabled Wrapper Construction System. *Proceedings 16th International Conference on*, (2000).

37. Lucas-Nülle, T. *Product Information Management in Deutschland*. pro literatur Verlag, 2005.

38. Makhoul, J., Kubala, F., Schwartz, R., and Weischedel, R. Performance measures for information extraction. *Broadcast News Workshop'99 Proceedings*, Morgan Kaufmann (1999), 249.

39. Manning, C.D., Raghavan, P., and Schütze, H. *An Introduction to Information Retrieval*. Cambridge University Press, 2009.

40. Matsumoto, T. Human-Computer Cryptography: An Attempt. *Response*, (1996), 68-75.

41. Merriam. Website. *http://www.merriam-webster.com*. Abgerufen am: 19.08.2011.

42. Muslea, I., Minton, S., and Knoblock, C.A. Hierarchical Wrapper Induction for Semistructured Information Sources. *Autonomous Agents and MultiAgent Systems 4*, 1 (2001), 93-114.

43. Patel-Schneider, P.F., Hayes, P., and Horrocks, I. OWL Web Ontology Language Semantics and Abstract Syntax. *http://www.w3.org/TR/owl-semantics/*. Abgerufen am: 17.08.2011.

44. Pemberton, S. XHTML 1.0: The Extensible HyperText Markup Language. *http://www.w3.org/TR/xhtml1*. Abgerufen am: 29.06.2011.

45. Pin-Shan, P. The Entity-Relationship Unified View of Data Model--Toward a unified view of data. *Database 1*, 1 (1976), 9-36.

46. Raggett, D., Le Hors, A., and Jacobs, I. HTML 4.01 Specification. *http://www.w3.org/TR/1999/REC-html401-19991224/*. Abgerufen am: 15.08.2011.

47. Rogers, M. HTML vs. XHTML Version Statistics. *http://blog.powermapper.com/blog/post/HTML-vs-XHTML-Version-Statistics.aspx*. Abgerufen am: 30.07.2011.

48. Sahuguet, A. and Azavant, F. Building intelligent Web applications using lightweight wrappers. *Data & Knowledge Engineering 36*, 3 (2001), 283-316.

49. Soderland, S. Learning Information Extraction Rules for Semi-Structured and Free Text. *Machine Learning 34*, 1 (1999), 233-272.

50. Stonebraker, M. and Hellerstein, J.M. Content Integration for E-Business. *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, ACM New York, NY, USA (2001), 552-560.

51. Vlist, E.V.D. XML Schema. *Informatik Spektrum 25*, 2002, 363-366.

52. Wall, L., Christiansen, T., and Orwant, J. *Programming Perl*. O'Reilly, 2000.

53. Zhai, Y. Web data extraction based on partial tree alignment. 14th international conference on World Wide Web (2005), 76-85.

54. ECMAScript Language Specification. Standard ECMA-262. Edition 5.1. *http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf*. Abgerufen am: 01.08.2011.

8 Abkürzungen und Akronyme

Ajax	Asynchronous JavaScript and XML
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart
DOM	Document Object Model
EH	Extraction Heuristics
ERM	Entity-Relationship-Modell
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IT	Information Technology
MDR	Mining Data Record
MVC	Model View Controller
NLP	Natural language processing
PTA	Partial Tree Alignment
SaaS	Software as a service
URL	Uniform Resource Locator
WPF	Windows Presentation Foundation
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language
XPath	XML Path Language
XSD	XML Schema Document

9 Anhang

9.1 Beispiel einer Ontologie: Book Data Schema

```
<DataSchema z:Id="i1" xmlns="http://www.xScraper.net"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
xmlns:z="http://schemas.microsoft.com/2003/10/Serialization/">
   <Name>Book</Name>
   <ObjectName>Book</ObjectName>
   <ObjectAttributes>
      <ObjectAttribute z:Id="i2">
         <IsObligatory>true</IsObligatory>
         <Name>Author</Name>
         <RecognitionPattern>^[A-Z]'?[a-zA-Z \,\-\.]*$
         </RecognitionPattern>
      </ObjectAttribute>
      <ObjectAttribute z:Id="i3">
         <IsObligatory>true</IsObligatory>
         <Name>Title</Name>
         <RecognitionPattern>[A-Z0-9][A-Za-z0-9\s\-\,\.\?\!]+
         </RecognitionPattern>
      </ObjectAttribute>
      <ObjectAttribute z:Id="i4">
         <IsObligatory>true</IsObligatory>
         <Name>Price</Name>
         <RecognitionPattern>[CHF|EUR|€]+[ ]*[\d\,\.]+|[\d\,\.]+
         [ ]*[CHF|EUR|€]+</RecognitionPattern>
      </ObjectAttribute>
      <ObjectAttribute z:Id="i5">
         <IsObligatory>false</IsObligatory>
         <Name>Publishing date</Name>
         <RecognitionPattern>[\d]{4}</RecognitionPattern>
      </ObjectAttribute>
      <ObjectAttribute z:Id="i6">
         <IsObligatory>false</IsObligatory>
         <Name>Availability</Name>
         <RecognitionPattern>lieferbar|[v|V]ersandbereit
         </RecognitionPattern>
      </ObjectAttribute>
   </ObjectAttributes>
</DataSchema>
```

9.2 Beispiel eines Scrapingszenarios: Thalia

```
<Scenario z:Id="i1" xmlns="http://www.xScraper.net"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
xmlns:z="http://schemas.microsoft.com/2003/10/Serialization/">
   <Actions>
      <XSAction z:Id="i2" i:type="StartPageAction">
         <Header>Open start page</Header>
         <Scenario z:Ref="i1"/>
         <Url>http://www.thalia.de</Url>
```

```
      </XSAction>
      <XSAction z:Id="i3" i:type="ClickAction">
         <Header>Click "Bücher"</Header>
         <Scenario i:nil="true"/>
         <XPath>//div[@id='navigation']/ul[1]/li[3]/span[1]/a[1]/img[1]</XPath>
      </XSAction>
      <XSAction z:Id="i4" i:type="ClickAction">
         <Header>Click "Krimis &amp;amp; Thriller"</Header>
         <Scenario i:nil="true"/>
         <XPath>//div[@id='navigation']/ul[1]/li[1]/ul[1]/li[23]/a[1]</XPath>
      </XSAction>
      <XSAction z:Id="i5" i:type="ClickAction">
         <Header>Click "Neuheiten"</Header>
         <Scenario i:nil="true"/>
         <XPath>//div[@id='navigation']/ul[1]/li[1]/ul[1]/li[2]/a[1]</XPath>
      </XSAction>
      <XSAction z:Id="i6" i:type="RepeatAction">
         <Header>Repeat</Header>
         <Scenario z:Ref="i1"/>
         <InnerActions>
            <XSAction z:Id="i7" i:type="ExtractListDataAction">
               <Header>Extract list data</Header>
               <Scenario z:Ref="i1"/>
               <NoLevelTags
xmlns:a="http://schemas.microsoft.com/2003/10/Serialization/Arrays">
                  <a:string>a</a:string>
                  <a:string>strong</a:string>
               </NoLevelTags>
               <RecordsXpath
xmlns:a="http://schemas.microsoft.com/2003/10/Serialization/Arrays">
                  <a:string>//div[contains(@class,'stArticleLeft')]</a:string>
               </RecordsXpath>
            </XSAction>
            <XSAction z:Id="i8" i:type="NextPageAction">
               <Header>Go to next page</Header>
               <Scenario z:Ref="i1"/>
               <ImageSource>http://www.thalia.de/buch-
resources/mandant/thalia/img/arrows_right_next_black.gif</ImageSource>
               <IsLimitedPagesCount>true</IsLimitedPagesCount>
               <MaxPagesCount>5</MaxPagesCount>
               <NextLinkXPath/>
               <NextPageActionType>ImageSource</NextPageActionType>
            </XSAction>
         </InnerActions>
      </XSAction>
      <XSAction z:Id="i9" i:type="ClassifyAction">
         <Header>Classify</Header>
         <Scenario z:Ref="i1"/>
         <DataSchema>C:\Users\Evgeny\Documents\_XScraper\Book Data Sche-
ma.xds</DataSchema>
         <Mappings>
            <ColumnMapping z:Id="i10">
               <DefaultValue>n.A.</DefaultValue>
               <ExtractionPattern/>
```

```
      <Source>/div[1]/h3[1]</Source>
      <Target z:Id="i11">
        <IsObligatory>true</IsObligatory>
        <Name>Author</Name>
        <RecognitionPattern>^[A-Z]'?[a-zA-Z \,\-\.]*$
        </RecognitionPattern>
      </Target>
    </ColumnMapping>
    <ColumnMapping z:Id="i12">
      <DefaultValue i:nil="true"/>
      <ExtractionPattern/>
      <Source>/div[1]/h2[1]</Source>
      <Target z:Id="i13">
        <IsObligatory>true</IsObligatory>
        <Name>Title</Name>
        <RecognitionPattern>[A-Z0-9][A-Za-z0-9\s\-\,\.\?\!]+
        </RecognitionPattern>
      </Target>
    </ColumnMapping>
    <ColumnMapping z:Id="i14">
      <DefaultValue>n.A.</DefaultValue>
      <ExtractionPattern>[CHF|EUR|€]+[ ]*[\d\,\.]+|[\d\,\.]+[ ]*
      [CHF|EUR|€]+</ExtractionPattern>
      <Source i:nil="true"/>
      <Target z:Id="i15">
        <IsObligatory>true</IsObligatory>
        <Name>Price</Name>
        <RecognitionPattern>[CHF|EUR|€]+[ ]*[\d\,\.]+|[\d\,\.]+[ ]*
        [CHF|EUR|€]+</RecognitionPattern>
      </Target>
    </ColumnMapping>
    <ColumnMapping z:Id="i16">
      <DefaultValue>n.A.</DefaultValue>
      <ExtractionPattern>[\d]{4}</ExtractionPattern>
      <Source i:nil="true"/>
      <Target z:Id="i17">
        <IsObligatory>false</IsObligatory>
        <Name>Publishing date</Name>
        <RecognitionPattern>[\d]{4}</RecognitionPattern>
      </Target>
    </ColumnMapping>
    <ColumnMapping z:Id="i18">
      <DefaultValue>n.A.</DefaultValue>
      <ExtractionPattern>lieferbar|versandbereit</ExtractionPattern>
      <Source i:nil="true"/>
      <Target z:Id="i19">
        <IsObligatory>false</IsObligatory>
        <Name>Availability</Name>
        <RecognitionPattern>
        lieferbar|[v|V]ersandbereit</RecognitionPattern>
      </Target>
    </ColumnMapping>
  </Mappings>
```

```
        <OfflineColumns
xmlns:a="http://schemas.microsoft.com/2003/10/Serialization/Arrays">
          <a:string>/div[1]/h2[1]</a:string>
          <a:string>/div[1]/h3[1]</a:string>
          <a:string>/div[1]/p[2]</a:string>
          <a:string>/div[1]</a:string>
        </OfflineColumns>
      </XSAction>
      <XSAction z:Id="i20" i:type="ShowDataAction">
        <Header>Show results</Header>
        <Scenario z:Ref="i1"/>
      </XSAction>
   </Actions>
   <Properties z:Id="i21">
      <delayAfterInput>500</delayAfterInput>
   </Properties>
</Scenario>
```

Erklärung

Hiermit versichere ich, diese Arbeit selbstständig verfasst und nur die angegebenen Quellen benutzt zu haben.

Unterschrift:

_____