

Die approbierte Originalversion dieser
Dissertation ist in der Hauptbibliothek der
Technischen Universität Wien aufgestellt und
zugänglich.

<http://www.ub.tuwien.ac.at>



The approved original version of this thesis is
available at the main library of the Vienna
University of Technology.

<http://www.ub.tuwien.ac.at/eng>



FAKULTÄT
FÜR INFORMATIK

Faculty of Informatics

Web Accessibility for the Blind Through Visual Representation Analysis

DISSERTATION

zur Erlangung des akademischen Grades

Doktor/in der technischen Wissenschaften

eingereicht von

Ruslan Fayzrakhmanov

Matrikelnummer 0728003

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Prof. Dr. Reinhard Pichler

Diese Dissertation haben begutachtet:

(Prof. Dr. Reinhard Pichler)

(Prof. Dr. Alessandro Proveti)

Wien, 03.12.2013

(Ruslan Fayzrakhmanov)

Web Accessibility for the Blind Through Visual Representation Analysis

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktor/in der technischen Wissenschaften

by

Ruslan Fayzrakhmanov

Registration Number 0728003

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Prof. Dr. Reinhard Pichler

The dissertation has been reviewed by:

(Prof. Dr. Reinhard Pichler)

(Prof. Dr. Alessandro Provetti)

Wien, 03.12.2013

(Ruslan Fayzrakhmanov)

Erklärung zur Verfassung der Arbeit

Ruslan Fayzrakhmanov
Favoritenstraße 9-11/184-2, A-1040 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum)

(Unterschrift Verfasserin)

Acknowledgements

This thesis would not have been possible without the support and encouragement of many people, who I would like to acknowledge in this section:

I am very thankful to Prof. Thomas Eiter for giving me the unique opportunity to undertake my Ph.D. studies at the Vienna University of Technology.

My sincere thanks to Katrin Seyr for her unceasing support and attention during my work on the thesis.

I am very grateful to my advisor, Prof. Reinhard Pichler who helped me to systematize the results of my research, ameliorate presentation of the material, and who also endeavored to ensure the completion of this work.

I would like to express my deepest gratitude to my second advisor, Dr. Robert Baumgartner who gave me invaluable suggestions on my own scientific research and thanks to whom, I was able to attain practical experience by participating in scientific projects and conferences.

I would also like to mention Prof. Georg Gottlob, who always encouraged me to pursue this creative work.

I very much appreciate the scientific experience and practical knowledge which I acquired through all the productive and dynamic work alongside my inspirational colleagues and friends from the DBAI group. I am particularly grateful to Bernhard Krüpl-Sypien, Wolfgang Holzinger, and Max Christopher Göbel. It was a lot of fun to work with them. The contributions which are included and discussed within this thesis are mainly based on cooperative work with our research projects.

I very much appreciate the recommendations provided by Prof. Yevgen Borodin regarding this thesis as they were extremely helpful in improving my work.

Many thanks to Alexey A. Viktorov and Jürgen Schwingshandl, who helped me in conducting a survey of users who are blind and evaluating the proposed approaches.

I would like to thank Stephen Longley for proofreading this thesis and ensuring its readability.

Last but by no means least, I am very thankful to my wonderful, loving and understanding parents and brother who by their example, advice and continuous support inspired me in my research.

This thesis has been supported by the Erasmus Mundus External Cooperation Window Programme of the European Union, by the Austrian Forschungsförderungsgesellschaft FFG under grants 819563 (ABBA project) and 829614 (TAMCROW project), and by the Austrian Science Fund (FWF) under grant P25207-N23.

Abstract

One of the most important attributes of a civilized society is based on the principle of equal opportunities for self-development and self-actualization within one's life. In today's contemporary information society, this principle can be realized by enhancing the accessibility of information resources for various groups of the population. Information is an inalienable part of today's life. This fact is clearly evident in the ongoing development and expansion of the World Wide Web (the Web)—a huge information platform that has provided vast opportunities for people by making it possible to effectively solve various tasks in business, education, science, and our everyday lives. With the help of the Web, a person can pay bills, buy products and services, complete university degrees online, search for and read articles, keep contact with their friends and so much more.

However, these opportunities are not available for everybody. For example, people who are blind encounter a lot of problems surfing the Web. This is a direct result of the majority of web pages being orientated at sighted users and not taking into account users who are blind. Blind users perceive the world differently and therefore need effective mechanisms for navigation through the elements on a web page and the thorough understanding of their semantic role in order to obtain an overview of a page in a few moments and perform a search. Current approaches provide us with limited solutions in the pursuit of particular goals such as web page segmentation or text summarization. Moreover, most approaches consider non-visual web page representations, such as source code (X/HTML and XML) and tree representations (the tag tree and DOM tree) which do not reflect the semantics that is analyzed by sighted users and posed exclusively on the rendered web page. This fact explains the limitations of such methods.

The goal of this thesis is to enhance the accessibility of web pages for blind users. In order to achieve this goal, we propose a complex solution covering the problems of automatic understanding of a web page's functional elements on the rendered web pages and navigation. Firstly, we provide the means for web page analysis and understanding on the visual level which carries the main information presented not only by the textual content, but also by the layout, styles and other visual characteristics perceivable by sighted users. Secondly, the challenges in improving web mobility of blind users based on the recognized logical elements of a web page are considered in detail and solutions are proposed.

Considering the complexity in the analysis of visual characteristics, close attention is paid to the modeling of a web page's visual features defined in the Unified Ontological Model (UOM) as well as developing methods and techniques for querying the model and approaches for realizing methods for web page understanding and information extraction. The uniqueness of the Unified Ontological Model (UOM) is based on its integration of different aspects of a web page in one

consistent model, such as geometry of the layout with quantitative and qualitative characteristics, interface with different functional elements and structures, and the DOM tree. The proposed cross-platform framework, Web Page Processing System (WPPS), ensures an efficient generation of the UOM according to the provided configuration and conveys an application programming interface (API) for developing methods and algorithms of web page processing. The WPPS demonstrated its effectiveness in developing methods of web page segmentation, web object identification and information extraction. For enhancing web accessibility based on the identified web page elements, the Multi-Axial Navigation Model (MANM) and navigation methodology are proposed in this thesis. Blindzilla, a cross-platform system which demonstrated its effectiveness in an evaluation with blind users, implements these concepts.

Kurzfassung

Chancengleichheit bei selbständiger Entwicklung und Aktualisierung für alle ist ein wichtiges Merkmal zivilisierter Gesellschaften. In der heutigen Informationsgesellschaft findet dieser Aspekt eine seiner Umsetzungen in der Verbesserung des Zugangs verschiedener Bevölkerungsgruppen zu Informationsressourcen. Information ist ein unabdingbarer Teil des heutigen Lebens. Diese Tatsache verdeutlicht die laufende Entwicklung und Expansion des World Wide Web (des Internets) – einer riesigen Informationsplattform, die für Menschen große Chancen eröffnet hat, indem sie die effiziente Lösung von Aufgaben im Geschäftsleben, in der Bildung, der Wissenschaft und in unserem täglichen Leben ermöglicht. Mit Hilfe des Internets kann man Rechnungen bezahlen, Produkte und Dienstleistungen kaufen, an einer Fernuniversität studieren, nach Artikeln suchen und sie lesen, Kontakt mit FreundInnen halten und vieles mehr.

Diese Möglichkeiten stehen jedoch nicht allen offen. Zum Beispiel sind blinde Menschen beim Surfen im Internet mit vielen Problemen konfrontiert. Dies liegt an der Orientierung der überwiegenden Mehrheit von Webseiten an sehenden BenutzerInnen, ohne Berücksichtigung von blinden BenutzerInnen. Blinde BenutzerInnen nehmen die Welt anders wahr und benötigen daher effiziente Mechanismen zur Navigation durch die Elemente einer Webseite und zum Verstehen von deren semantischer Rolle, um in kurzer Zeit einen Überblick über eine Seite zu bekommen und diese durchsuchen zu können. Aktuelle Zugänge bieten eingeschränkte Lösungen, die bestimmte Ziele verfolgen, seien diese die Segmentierung von Webseiten oder Textzusammenfassungen etc. Zudem muss erwähnt werden, dass die meisten Zugänge sich auf nichtvisuelle Webseitendarstellungen wie den Quellcode (X/HTML und XML) und Baumdarstellungen (Tag-Baum oder DOM-Knotenbaum) beziehen, die nicht die Semantik widerspiegeln, die von sehenden BenutzerInnen analysiert und ausschließlich auf der dargestellten Webseite abgebildet wird. Diese Tatsache erklärt die begrenzten Möglichkeiten derartiger Methoden.

Das Ziel dieser Dissertation ist es, die Zugänglichkeit von Webseiten für blinde BenutzerInnen zu verbessern. Um dieses Ziel zu erreichen, schlagen wir eine komplexe Lösung vor, die alle Probleme des automatischen Verstehens der funktionalen Elemente einer Webseite auf den dargestellten Webseiten und ihrer Navigation abdeckt. Zunächst stellen wir Mittel zur Analyse und zum Verstehen einer Webseite auf der visuellen Ebene zur Verfügung, die die Hauptinformation transportieren, die nicht nur durch den Textinhalt, sondern auch durch Layout, Stile und andere Eigenschaften, die von sehenden BenutzerInnen wahrgenommen werden können, vermittelt wird. Zweitens werden die Probleme bei der Verbesserung der Internetmobilität von blinden BenutzerInnen auf der Basis der erkannten logischen Elemente einer Webseite im Detail untersucht und Lösungen vorgeschlagen.

Angesichts der Komplexität der Analyse der visuellen Eigenschaften liegt ein Hauptaugenmerk dieser Dissertation auf der Modellierung der visuellen Merkmale einer Webseite nach dem Unified Ontological Model (UOM) und der Entwicklung von Methoden und Techniken zum Abfragen des Modells und von Zugängen zur Realisierung von Methoden für das Verstehen von Webseiten und der darauf basierenden Extraktion von Informationen. Die Einzigartigkeit des UOM besteht in der Integration verschiedener Aspekte einer Webseite, wie etwa der Geometrie des Layouts mit quantitativen und qualitativen Eigenschaften, Schnittstellen mit verschiedenen funktionellen Elementen und Strukturen und dem DOM-Knotenbaum in einem konsistenten Modell. Der vorgeschlagene plattformübergreifende Rahmen, Web Page Processing System (WPPS), sichert eine effiziente Generierung des UOM laut der spezifizierten Konfiguration und stellt dem Programmierer eine Schnittstelle zur Anwendungsprogrammierung (Application Programming Interface – API) zur Entwicklung von Methoden und Algorithmen der Webseitenverarbeitung zur Verfügung. Das WPPS hat seine Effizienz bei der Entwicklung von Methoden zur Webseitensegmentierung, Webobjektidentifikation und Informationsextraktion bewiesen. Zur Verbesserung des Internetzugangs auf der Basis von identifizierbaren Webseitenelementen schlägt diese Dissertation das multiaxiale Navigationsmodell (MANM) und die multiaxiale Navigationsmethode vor. Ein plattformübergreifendes System, Blindzilla, das seine Effizienz in der Evaluation mit blinden BenutzerInnen gezeigt hat, setzt diese Konzepte um.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	The State of the Art: Challenges	2
1.3	Aim and Objectives of this Thesis	4
1.4	Theoretical Contribution	6
1.5	Practical Contribution	7
1.6	Publications	8
1.7	Structure of the Thesis	9
2	State of the Art and Related Work	11
2.1	Web Accessibility for the Blind	12
2.1.1	Standards and Guidelines	12
2.1.2	Hardware	13
2.1.3	Software	14
2.2	Web Page Navigation	18
2.2.1	Contemporary Realized Navigation Approaches	18
2.2.2	Metaphor of Spatial Navigation	19
2.2.3	Conclusion on Web Page Navigation	20
2.3	Web Page Processing	22
2.3.1	Web Information Extraction	22
2.3.2	Web Page Understanding	24
2.4	Web Page Representations	24
2.4.1	Standard Forms of Web Page Representation	25
2.4.2	Source code	26
2.4.3	Textual representation	27
2.4.4	Tree representation	28
2.4.5	CSS Specifications for Rendering Web Pages	30
2.4.6	Quantitative Visual Representation	35
2.4.7	Qualitative Visual Representation	36
2.4.8	Analysis of the Considered Models	43
2.5	Semantic Web Approach	46
2.6	Ontology and Logical Inference	47
2.6.1	Ontology Languages	49

2.6.2	Inference Rules	54
2.6.3	Standard Ontology Reasoning Services	57
2.6.4	SPARQL	57
2.7	Discussion	58
3	Modeling the Geometric Structure of a Web Page	61
3.1	Existing Geometric Structures	62
3.2	Quantitative and Qualitative Models	63
3.3	Web Page Canvas and Unit of Measure	65
3.4	A Web Page Structure, Geometric Object and its Attributes	66
3.5	Spatial Relations	68
3.6	Topological Relations	68
3.6.1	RCC8	69
3.6.2	Refinement of the RCC for Blocks	71
3.7	Direction Relations	74
3.7.1	Quantitative Direction	74
3.7.2	Qualitative Direction	75
3.8	Distance Relations	77
3.8.1	Quantitative Distance	77
3.8.2	Qualitative Distance	79
3.9	Alignment Relations	79
3.10	Interval Relations	81
3.10.1	Main Concepts	82
3.10.2	Fuzziness in Interval Relations	82
3.10.3	Centering Relation	86
3.10.4	Two-Dimensional Interval Relations And Centering	86
3.11	Analysis of the Spatial Relations and Visibility of CSS Boxes	91
3.11.1	WPPS-HTML-DS1 Dataset	91
3.11.2	RCC8	91
3.11.3	Quantitative Direction	94
3.11.4	Quantitative Distance	95
3.11.5	Alignment relations	96
3.11.6	CSS Box Visibility	96
3.12	Discussion	97
4	The Unified Ontological Model of a Web Page	101
4.1	Web Page Authoring and Visual Representation	102
4.2	Overview of the Unified Ontological Model for the Web Page Processing	104
4.3	Properties of Relations	108
4.4	The Physical Model of a Web Page	109
4.4.1	Extended DOM	110
4.4.2	Block-based Geometric Model	113
4.4.3	Interface Model	123
4.5	The Logical Model of a Web Page	127

4.6	Discussion	127
5	Web Page Processing	129
5.1	A Principle of Web Page Processing based on the Unified Ontological Model	129
5.2	An Object-Oriented Abstraction for the Unified Ontological Model	131
5.2.1	Declarative and Imperative Approaches	131
5.2.2	A Required Abstraction	132
5.2.3	Ontology in Object-Oriented Applications	134
5.2.4	A Bridged Adapter	135
5.3	WPPS: A System for Web Page Processing	138
5.3.1	Architecture	138
5.3.2	WPPS Configuration	140
5.3.3	Physical Model Instantiation	150
5.3.4	WPPS API	155
5.3.5	WPPS GUI	158
5.4	Developing Methods by Means of the WPPS Framework	158
5.4.1	WPPS Methods Development Life Cycle	158
5.4.2	Basic Examples of WPPS Methods	161
5.5	WPPS Evaluation	166
5.5.1	Goal and Objectives	166
5.5.2	WPPS Parameters and Queries	167
5.5.3	Evaluation Wrappers	168
5.5.4	Performance Analysis	171
5.5.5	Conclusions on the WPPS Evaluation	188
5.6	WPPS in the Problem of Basic Web Object Identification	190
5.7	Discussion	194
6	Web Accessibility: A Multi-Axial Navigation	197
6.1	Ameliorating Blind Users' Mobility	198
6.1.1	Navigable Web Page Objects	199
6.1.2	Web Page Mobility	201
6.2	The Multi-Axial Navigation Model	203
6.2.1	Main Concepts	203
6.2.2	Example of a Navigation Model	204
6.2.3	Formal Presentation of Axes	208
6.2.4	Ontology	211
6.3	Building the Multi-Axial Navigation Model	211
6.4	Methodology of Navigation	213
6.4.1	Observation	213
6.4.2	Locomotion	215
6.5	Blindzilla: Implementation	218
6.5.1	System Architecture	218
6.5.2	Model Generation Component	220
6.5.3	Navigation Component	221

6.6	Blindzilla: Evaluation	223
6.7	Discussion	226
7	Conclusion and Future Work	229
7.1	Results	229
7.1.1	Web Page Processing	229
7.1.2	Web Accessibility	232
7.2	Future Work	233
7.2.1	Web Page Processing	233
7.2.2	Web Accessibility	234
A	Queries for Spatial Relations Analysis	237
A.1	RCC8	237
A.2	Quantitative Directions	241
A.3	Quantitative Distances Between Border Projections	241
A.4	Alignment relations	242
B	A Survey of Blind Users of the Web	247
B.1	Demographics of Respondents	248
B.2	Expertise and Preferences	249
B.3	Becoming Familiar with New Web Pages	251
B.4	General Navigation	253
B.5	Navigation Through Tables and Lists	255
B.6	Accessibility of Web Pages by Genres	257
B.6.1	Information Search	258
B.6.2	Social Media	259
B.6.3	News Websites	260
B.6.4	Web Forums	261
B.6.5	On-line Shops	262
B.6.6	Weblogs	263
	Bibliography	265
	Acronyms	281

Introduction

The power of the Web is in its universality. Access by everyone regardless of disability is an essential aspect.

— Tim Berners-Lee, *W3C Director and inventor of the World Wide Web*

1.1 Motivation

The World Wide Web (the Web) plays a crucial role in the development of a modern information society [119, 298]. The Web is a vast repository of information formed mainly by the huge set of interconnected web pages. The peculiarities of web resources, such as presenting information in semistructured or unstructured forms and orientation on the average sighted user, defined a number of problems related to providing a resource in the required form. One such problem consists of accessibility and perception simplicity of web resources for the end user, which became the main reason for originating a theoretical and practical area of research—web accessibility. Web accessibility is aimed at resolving the problem of providing information in convenient forms for various segments of the population and different age categories. However, the main target group consists of people with auditory, cognitive, neurological, physical, speech, and visual impairments [56, 113, 230]. Enhancing the accessibility of web resources for disabled individuals can greatly improve their quality of life and enable them to be more integrated into society. The accessible part of the Web provides numerous possibilities for disabled people such as on-line shopping, on-line and self-education, and participation in various communities via the Internet. In addition, it can ensure the availability of various occupations where work can be performed remotely [212, Ch. 2].

The decision to focus on blind users of the Web as a target group was based on the number of reasons related to the inaccessibility of the most important constituent of a web page—visual. It is commonly known that web pages are oriented at sighted users (X/HTML was designed

as a visual formatting language). Thus, visual characteristics along with textual content carry essential information about the web page's logical structure and functional roles of elements [111]. As discovered by H. Petrie, F. Hamilton, and N. King [193] during their analysis of groups of individuals with different impairments (physically impaired, dyslexic, hearing impaired, partially sighted, and blind), blind users encountered more problems than the other groups during their work on the Web.

1.2 The State of the Art: Challenges

The problem of web accessibility is addressed in the works of R. Baguma, J.P. Bigham, Y. Borodin, T. Comber, C. Goble, F. Hamilton, V.L. Hanson, S. Harper, N. King, J.T. Lubega, J.R. Maltby, P. Neha, H. Petrie, T.V. Raman, J.T. Richards, M.A. Roschina, O.V. Shevkun, H. Simon, V.I. Shvetsov, H. Takagi, and Y. Yesilada.

There are two main directions for approaching the enhancement of web accessibility: 1) by means of developing standards and guidelines which lead to the creation of more accessible content for specific groups of users and corresponding assistive technology that provides easier access to the accessible content; 2) by means of developing approaches and tools for making inaccessible content accessible.

There are several standards and guidelines, such as WAI (Web Accessibility Initiative) guidelines [238], Section 508 guidelines [233], RNIB (Royal National Institute of Blind People) guidelines [208], AFB (American Foundation for the Blind) guidelines [8], and IBM guidelines [130], which describe in detail the requirements for creating content by web authors and accessing it by different groups of users (adults, visually impaired users, deaf, etc.) [113]. However, the most comprehensive ones are WAI guidelines provided by the W3C organization [239]. The development of standards is an important factor in the evolution of the Web, particularly in the inalienable aspiration to create a universally accessible platform. Unfortunately, this method of enhancing web accessibility experiences serious difficulties because an overwhelming amount of web resources do not follow any of the specified standards. This fact was discovered in the work of J.P. Bigham [29, 30], H. Takagi [226], and T. Watanabe [292]. Most web authors lack the time or knowledge of existing guidelines [137]. Moreover, it is worth mentioning that web accessibility standards always tend to be a step behind rapidly changing technologies.

Access to web page content is realized by the use of screen readers (e.g., JAWS [90], Window-Eyes [178], Apple's VoiceOver [11], NVDA [188], SuperNova [60]) and specialized browsers (e.g., PWWebspeak, BrookesTalk [303], Home Page Reader [13]), which can also be used along with Braille display [56]. These technologies allow blind users to read information represented on the display via a speech synthesizer [113, 201]. Based on the DOM tree and dependent on accessibility guidelines, these technologies are extremely limited by the huge amount of inaccessible web resources. Furthermore, the guidelines remain limited based on their comprehension of accessibility problems [31, 56]. In contrast, specialized browsers, such as Emacspeak [202] and HearSay [36, 222], utilize additional processing methods of the web page to enhance its accessibility. For example, HearSay together with its extensions, such as CSurf [171] and Dynamo [35], conduct an analysis for detecting a web page's logical structure and ameliorate user interaction with the web page.

The additional analysis of a web page considerably improves its accessibility. It can also be met in applied research such as Web Adaptation and task specific automation. Web Adaptation is aimed at transforming web pages into more accessible forms according to the user's needs and hardware used, and ultimately to make the use of standard screen readers more acceptable. Examples include SADIe [24], AxsJAX [48], and transcoding based on the Spatial Graph Grammar (SGG) [144]. Task specific automation makes it possible to accompany and support a blind user in specific tasks on some website, such as buying a new product, searching for a flight, or writing a post in a web forum. Automation is realized in solutions such as CoScripter [159], TrailBlazer [31], Ubiquity [65], and CoCo [157].

Existing web accessibility approaches and tools target the analysis of two main components: web page logical structure and navigation [225]. Providing blind users with the correct logical structure of a web page's content gives them the possibility to build an adequate mental model [226] as well as the possibility to understand the content of a web page. Most existing methods and tools (e.g., JAWS [90], NVDA [188], and WebAnywhere [33]) consider DOM tree or source code represented by the use of X/HTML to provide access to the content for blind users. These approaches are quite limited due to the fact that in regards to current web pages, these forms of representations are not the main information carriers of a web page's layout and its visual characteristics. A web page's visualization (which reflects web page semantics) is defined increasingly by CSS rules. Therefore, these methods are often ineffective and inaccurate. Moreover, a conceptual gap between the source code (i.e., XML and X/HTML code and thus the DOM tree) and layout structure is growing even larger [190]. In contrast, consideration of visual characteristics provides an opportunity to develop more effective and robust methods. This is due to the fact that there is a considerably smaller set of design patterns for representing various objects and data structures as opposed to the source code. Furthermore, methods based on a web page's visual representation can leverage principles that underlie the process of human object recognition (e.g., Gestalt theory) [142, 151] and which provides an opportunity to develop more robust approaches. The advantage of considering visual cues is also cited by P.S. Hiremath, S.P. Algur, E. Oro, M. Ruffolo, et al. in [123, 189].

Unfortunately, there is no standardized visual model suitable for automatic analysis. The CSS style sheets [260] are able to define rules for visual formatting of a web page but are useless in developing algorithms for the layout analysis. Computed CSS attributes along with DOM tree also provide limited possibilities for the analysis, especially making it impossible to do automatic spatial reasoning and making it more difficult to consider different spatial configurations of web objects. The issue of modeling a web page's visual representation is not studied in the works dedicated to web accessibility. Other research directions (e.g., Web Data Extraction, Web Personalization, Web Form and Web Page Understanding) related to web page analysis also do not pay enough attention to this problem. Existing models and structures reflecting a web page's visual characteristics are intended for solving specific problems. For instance, graph-based structure [144] used for transforming web pages for mobile devices and SDOM [190] underlying XPath language (extension of XPath 1.0) for selecting DOM nodes based on a small set of spatial relations. Web authors have to model new web page representation, features and relations every time they develop a new method for solving a specific problem because the model already developed is usually not suitable. As such, there is no universal extensible model of a web page's

visual representation which can be used for a wide range of tasks related to the analysis of a web page's visual representation. Therefore, one of the main objectives of this thesis is to develop this type of model.

Navigation is another important aspect that was mentioned earlier. Most of existing works which address the challenge of navigation for blind users within a web page [29, 97, 112, 207, 226] consider this problem from the perspective of both navigation according to the depth-first traversal over a DOM tree and spatial navigation on a two-dimensional web page's canvas [33]. In the absence of a proper visual model of a web page, the navigation methods based on visual features target interactions with CSS boxes which have their counterparts in the corresponding DOM tree. Thus, these approaches provide blind users with limited navigation possibilities based on different HTML tags which are very often misused.

It is important to note that in the absence of visual channel of perception, blind users generally receive information from web pages aurally via speakers and tactually via Braille displays that defines their one-dimensional sense of derivable information. Unfortunately, one-dimensional navigation according to high-dimensional information space of web resources is not considered in the literature. These issues clearly necessitate the development of new navigation methods which take into account both one-dimensional perception of information by blind users and consideration of efficient navigation through multi-dimensional information space of web pages.

1.3 Aim and Objectives of this Thesis

The *aim* of this research is to enhance web page accessibility based on the analysis of a web page's visual representation and its logical structure and providing navigation means by considering the semantics of a web page's elements. In order to achieve this goal, the following *objectives* were formulated:

1. Elaborate a unified extensible metamodel of web pages reflecting their quantitative and qualitative visual (e.g., layout, spatial features and relationships, color characteristics) and functional (e.g., types of logical elements, their features, role) characteristics. The metamodel should be applicable for a wide range of tasks related to the analysis of a web page's visual representation.

This objective allows the presentation of a web page's semantics perceived by the sighted user in one consistent model.

2. For the model proposed in the objective 1:

- a) Develop an approach to generate the model for specified web pages according to the provided requirements and methods for simplifying the model. The requirements defining a configuration of a web page's model include types of ontological concepts and roles which should be instantiated within the model, whereas the model simplification refers to removing elements which are not perceptible by sighted users and therefore should not be considered.

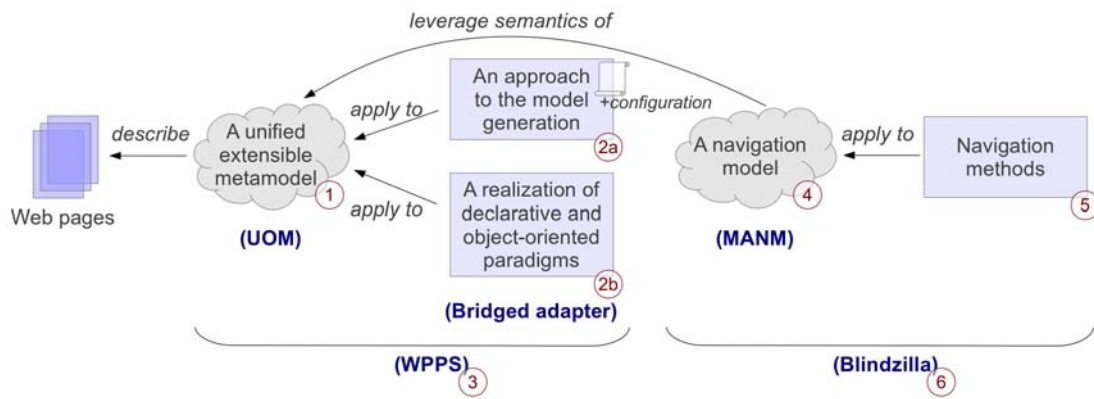


Figure 1.1: A schematic representation of the objectives and results

The objective permits the transfer of semantics perceived by the sighted user into the proposed model. Furthermore, this task is necessary for building a model of optimal configuration with the required elements. This ensures faster interaction with the model.

- b) Design an approach that allows the application of declarative and object-oriented programming principles in creating methods of a web page’s visual representation analysis. An interface provided to the object-oriented methods operating on the web page model should be robust against various valid configurations of the web page’s model. (This objective provides a possibility not only to create or apply various existing declarative and imperative approaches to web page layout analysis, but also to develop new methods with the combined usage of both paradigms while considering the advantages of both of them.)

The objective 2 provides developers with methods for developing approaches that enhance the accessibility of web pages and leveraging a semantically rich web page model.

3. Implement a framework to confirm the feasibility and efficiency of the approaches proposed in objectives 1 and 2.
4. Elaborate a general navigation model which provides the necessary constructs for building structures convenient for one-dimensional navigation; such a model should reflect both semantic and spatial relations between a web page’s logical objects.
5. Develop methods for efficient navigation via the navigation model proposed to ensure fast information searches and to assist blind users in building adequate mental models.
6. Implement a prototype for the proof of concepts realized in this thesis and confirmation of the feasibility and effectiveness of the overall approach.

The objectives specified are schematically illustrated in Figure 1.1.

1.4 Theoretical Contribution

This work is the result of research focusing on the problem of enhancing accessibility that is considered for the first time from the viewpoint of understanding a web page's visual representation and creating a navigation methodology to ensure quick access to required information [71]. The following results were achieved according to the objectives specified:

- In order to model a web page's visual representation that is suitable for developing methods in the fields of Web Accessibility, Web Page Understanding, and Information Extraction, the **Unified Ontological Model (UOM)** [73, 82, 151] was developed. The UOM is a domain ontology for describing web pages consisting of the Physical Model (PM), which expresses its aspects such as layout, interface, visual characteristics, and DOM trees, and the Logical Model (LM), which provides a semantic description of a web page's elements. The UOM excels in providing rich vocabulary to describe different aspects of a web page, uniting them in one consistent model. It is a convenient form for automatic web page processing. Ontological concepts and roles (reflecting quantitative and qualitative information) have been adopted from different theoretical and practical fields of research including document processing, interface design, topology, spatial reasoning and geographic information systems. [**Objective 1**]
- The automatic creation of the UOM with necessary configurations is required both for the efficiency of model generation with a predefined set of elements and efficient querying. Generation of the PM of the UOM is performed during depth-first traversal over DOM trees of a selected web page, rendered by the web browser engine. Corresponding CSS boxes are considered together with their computed CSS attributes during this phase. Based on the data collected, necessary elements of the PM are automatically created together with features, relationships and reasoner applied. The LM is built later as a result of the application of web page processing methods. [**Objective 2.a**] [76, 77]

A *bridged adapter* software design pattern proposed in this thesis makes it possible to apply the object-oriented paradigm regardless of a certain configuration of the UOM as well as declarative methods (for instance, SPARQL queries together with logical rules). The *bridged adapter* allows computation of different qualitative features and relations (linguistic variables) by request taking into consideration various predefined fuzzinesses. The application of different paradigms increases the efficiency in developing new approaches and methods for Web Page Understanding and Web Information Extraction. [**Objective 2.b**] [76, 77]

In addition, an application programming interface (API) was designed for the purpose of applying various selective queries on the model and processing the results acquired. The functions of the API are grouped into three categories: selectors, processing functions and statistical functions. Selectors are used for selecting certain subsets of objects from the UOM by the object type, some predicate specified, or from the predefined area on a web page's canvas (in this case R-trees are involved for efficiency). Processing and statistical functions are intended for grouping, filtering results of query evaluation and statistical analysis. [**Objective 2.b**] [76, 77]

- The **Multi-Axial Navigation Model (MANM)** [21, 151] was developed according to objective 4. The **MANM** is a domain ontology and intended for establishing various orderings (serializations) of the subsets of objects from the UOM as well as necessary semantic and spatial relations for navigation. Thus, the **MANM** does not only contain information about possible reading orders but possible transitions from one order to another according to their semantic or spatial closeness as well. The **MANM** is the first attempt to provide a universal means for building structures which allow one-dimensional navigation through multi-dimensional information space. This approach is fully compatible with aural and tactual perception of a web page's content which is one-dimensional compared to visual perception. **[Objective 4]**
- Navigation methods were designed on the basis of the **MANM** [83, 85]. They are based on the investigation of blind people's mobility [38, 112] and cognitive approach to navigation [132, 194], and provide the necessary means for efficient navigation through logical objects. In contrast, most existing solutions are based on the DOM tree and consider navigation as a traversal through the DOM elements or elements of the same type according to the depth-first traversal. These approaches not only limit users in the variety of navigation locomotion, but also in the lack of necessary information of the logical structure of a web page. **[Objective 5]**

1.5 Practical Contribution

- The **Web Page Processing System (WPPS)** was developed [76] to create new methods for web page processing (e.g., web data extraction or web object identification). **WPPS**, a highly configurable cross-platform Eclipse RCP-based application with Firefox (XULRunner v.1.9.2) integrated, is the first framework which allows an implementation of a wide range of methods adopted from the fields of Web Information Extraction, Document Understanding and Web Page Accessibility. It automatically generates an instance of the PM for a web page according to the provided configuration and builds the LM as a result of applying processing methods (e.g., wrappers) on the instance of the PM. The framework allows the developer to use both declarative and object-oriented paradigms due to the bridged adapter software design pattern realized. In addition, the framework makes it possible to control the topological organization of the UOM sub-models, modes of creating objects, their attributes and relations. The configuration of the UOM and the framework enable developers to control the execution of their implemented method. The developed API provides the necessary functionality for querying the UOM, processing the results and building the LM. The framework integrates R-trees for querying spatial objects efficiently. **WPPS** showed its efficiency during its application in the **ABBA** [236] and **TAMCROW** [237] projects of TU Vienna. **[Objective 3]**
- A cross-platform Eclipse RCP-based tools (referred to as **Blindzilla** prototypes) were developed to evaluate the overall methods assuring enhancement of web page accessibility and mobility of blind users. This prototype is an implementation of the proposed approaches with the **MANM** and navigation methods as the most important components [236]. Based on the **WPPS** framework, it performs a segmentation and analysis of web pages for building the corresponding **MANM**. The blind user is provided with keystrokes to navigate through the **MANM**, where navigation is implemented according to the methodology developed. The

system is equipped with speech synthesizers such as Free TTS and ESpeak. Blindzilla proved its effectiveness during an evaluation by a small group of users who were able to perform a predefined set of tasks more than 3 time faster on average compared to other contemporary screen readers such as JAWS, Window-Eyes, and FireVox. [**Objective 6**]

1.6 Publications

The results of this research have been published at international conferences and journals.

- **Web object identification for web automation and meta-search.**
I. Kordomatis, C. Herzog, R.R. Fayzrakhmanov, B. Krüpl-Sypien, W. Holzinger, and R. Baumgartner.
The 3rd International Conference on Web Intelligence, Mining and Semantics (WIMS'13), Madrid, Spain, 12–14 June, 2013, article No. 13, New York, 2013. ACM.
- **Feature-based object identification for web automation.**
C. Herzog, I. Kordomatis, W. Holzinger, R.R. Fayzrakhmanov, and B. Krüpl-Sypien.
The 28th Annual ACM Symposium on Applied Computing (SAC'13), Web Technologies Track, Coimbra, Portugal, 18–22 March, 2013, pages 742–749, Coimbra, 2013. ACM.
- **WPPS: A framework for web page processing.**
R. R. Fayzrakhmanov.
The 13th International Conference on Web Information Systems Engineering (WISE'12), Demo Session, Paphos, Cyprus, 28–30 November, 2012, pages 800–803. Springer, 2012.
- **A blocks-based geometric model of web pages for automatic processing and information extraction.**
R. R. Fayzrakhmanov.
Science and Business: Development Ways, 9(15):56–64, 2012.
- **WPPS: A novel and comprehensive framework for web page understanding and information extraction.**
R. R. Fayzrakhmanov.
The International Conference IADIS WWW/Internet, Madrid, 18–21 October, 2012, pages 19–26, Madrid, 2012. IADIS Press.
- **A versatile model for web page representation, information extraction and content re-packaging.**
B. Krüpl-Sypien, R.R. Fayzrakhmanov, W. Holzinger, M. Panzenböck, and R. Baumgartner.
The 11th ACM Symposium on Document Engineering (DocEng2011), Mountain View, USA, 19–22 September, 2011, pages 129–138, New York, 2011. ACM.
- **Information extraction from web pages based on their visual representation.**
R.R. Fayzrakhmanov.
The 11th International Conference on Web Engineering (ICWE'11), PhD Symposium, Paphos, Cyprus, 20–24 June, 2011, pages 342–346, Heidelberg, 2011. Springer.
- **Multiaxial navigation system for improving web accessibility.**
R.R. Fayzrakhmanov, B. Krüpl, and W. Holzinger.

The 2nd International Internet Conference on Innovative Technologies: Theory, Tools, Implementation (INNOTECH'2010), Perm, Russia, 2010, pages 130–136, Perm, 2010. Perm State Technical University.

- **Web 2.0 vision for the blind.**

R. Baumgartner, R.R. Fayzrakhmanov, W. Holzinger, B. Krüpl, M.C. Göbel, D. Klein, and R. Gattringer.

Web Science Conference 2010 (WebSci'10), Raleigh, USA, 26–27 April, 2010, pages 1–8, 2010.

- **Modelling web navigation with the user in mind.**

R.R. Fayzrakhmanov, M.C. Göbel, W. Holzinger, B. Krüpl, A. Mager, and R. Baumgartner.

The International Cross Disciplinary Conference on Web Accessibility (W4A'2010), Raleigh, USA, 26–27 April, 2010, article No. 14, New York, 2010. ACM.

- **A unified ontology-based web page model for improving accessibility.**

R.R. Fayzrakhmanov, M.C. Göbel, W. Holzinger, B. Krüpl, and R. Baumgartner.

The 19th International Conference on World Wide Web (WWW'2010), Raleigh, USA, April 26–30, 2010, pages 1087–1088, New York, 2010. ACM.

1.7 Structure of the Thesis

This thesis is structured as follows:

Chapter 2 discusses the state of the art in the field of Web Accessibility, existing approaches, tools, and relevant issues. We highlight the importance of analyzing and processing web pages for providing necessary semantics and effective navigation methods. We consider web page processing from the viewpoint of the well developed areas of Web Information Extraction and Web Page Understanding. This chapter also describes different models for web page representations analyzing literature in the fields of Web Accessibility, Web Information Extraction, Document Understanding, and Web Page Understanding. In addition, it gives a brief overview of ontology languages and technology within the context of the Semantic Web. [71]

In **Chapter 3**, we introduce the main concepts and definitions which underlie the models, approaches, and tools presented in this thesis. In particular, we give definitions of quantitative and qualitative models as well as define a canvas, geometric objects, their attributes and relations. We specify the main spatial relations such as topological, direction, distance, alignment and those based on the Allen's interval relations which are expressly or by implication used in different areas related to web page processing and document understanding. We propose the refinement of RCC8 relations while considering peculiarities of the rectangular shape of geometric objects. Furthermore, we introduce fuzzy equality relation which takes into account inaccuracies on a web page canvas. All the relations introduced are expressed via Two-Dimensional Interval Relations with Centering (2DIRC) which provides a possibility to eliminate invalid combinations of different types of spatial relations between geometric objects. The chapter concludes with the presentation of various statistical characteristics of the introduced spatial relations and their analysis. [73, 81]

Chapter 4 mainly focuses on developing and defining the UOM, in which a web page's visual representation plays a crucial role. This chapter also presents a web page's conceptual model reflecting its different representations and aspects and which was used in formalizations of the UOM. [77, 82, 151]

In **Chapter 5**, we discuss a web page processing which results in instantiating the LM. We introduce a bridged adapter software design pattern to provide an object-oriented abstraction of the ontology and the WPPS framework which implements the proposed UOM and principles of web page processing. We also propose an approach providing interface for developing methods based on declarative and object-oriented paradigms. [76, 77, 84, 121, 145]

In **Chapter 6**, we introduce the main approaches to overcome challenges that blind users encounter when they utilize the Web. In particular, we discuss issues related to building a mental model relevant to a web page and blind users' mobility. We specify the MANM together with a navigation methodology that presents the possibility of more conscious and efficient navigation on web pages. [21, 74, 83, 85, 151]

Chapter 7 summarizes the results achieved and provides an outlook for possible research directions in the future.

State of the Art and Related Work

Disability is a complex phenomenon, reflecting the interaction between features of a person's body and features of the society in which he or she lives. Overcoming the difficulties faced by people with disabilities requires interventions to remove environmental and social barriers.

— World Health Organization, 2013

(<http://www.who.int/topics/disabilities/en/>)

In this chapter, we present the state of the art in the field of Web Accessibility referring to users who are blind and consider the advantages and disadvantages in existing standards, methods, and tools (see Section 2.1). The crucial aspect in the accessibility of information resources is related to their navigation characteristics and the means realizing navigation functionality, that is discussed in Section 2.2, where we identify the main issues related to web navigation. Furthermore, we consider the challenge of enhancing web page accessibility from the viewpoint of Web Page Understanding (WPU) and Web Information Extraction (WIE), presented in Section 2.3. These fields of research are rich in different methods of web page processing and identifying various patterns and regularities on a web page. In Section 2.4, we for the first time conduct a survey and comparative analysis of different models and web page representations used in research areas such as Web Accessibility, Web Information Extraction and Web Page and Document Understanding. We also highlight the importance of considering visual cues, effectiveness of methods analyzing visual features, and identify the main challenges regarding web page representations which we resolve within the scope of this thesis. In Section 2.5, we pay particular attention to the importance of considering technologies of the Semantic Web, a unified platform for building accessible and effective information resources. Section 2.6 discusses ontology and automatic reasoning from the viewpoint of its application in the Semantic Web and as a modeling language, which we leverage for modeling the unified representation of the web page. Section 2.7 concludes this chapter.

2.1 Web Accessibility for the Blind

Web accessibility refers to the practice of making pages on the Web accessible to all users, especially to those with disabilities [113,230]. There are plenty of different approaches developed so far for enhancing web page accessibility, however, the best practice has not yet been achieved. This is due to the constant application of new technologies and evolution of the Web, that in turn requires the development of new approaches and tools in making the Web an accessible platform.

Several main aspects need to be considered to ensure web accessibility: standards and guidelines (see Section 2.1.1), hardware (see Section 2.1.2), and software (see Section 2.1.3). Specialized hardware and software are usually called *assistive technology* [113] or by the more precise term *typhlot technology* [212] (from the Greek word “τυφλός”—blind).

2.1.1 Standards and Guidelines

The importance of incorporating web accessibility was highlighted by the World Wide Web Consortium (W3C) which established relevant department for the Web Accessibility Initiative (WAI) [238]. The latter proposes corresponding guidelines and specifications addressing people of different age and disabilities (visual, auditory, physical, speech, cognitive, and neurological disabilities):

- Web Content Accessibility Guidelines (WCAG) [258] describe techniques for developing web pages accessible for a wide range of people with disabilities.
- Authoring Tool Accessibility Guidelines (ATAG) [265] provide recommendations for designing web content authoring tools that are both more accessible to authors with disabilities and designed to support the production of more accessible web content according to WCAG.
- User Agent Accessibility Guidelines (UAAG) [284] specify recommendations for developing accessible web agents taking into account WCAG. User agents include web browsers, media players, and other types of software that retrieve and render web pages.
- Accessible Rich Internet Applications (WAI-ARIA) [259] is a specification for enhancing accessibility of contemporary interactive web pages—web applications—which distinguish by relatively complex interface and dynamic content. The WAI-ARIA recommendation defines a set of attributes for HTML elements which provide assistive technology (typhlot technology) with information of a web page’s logical structure, types of interface elements, their behavior, and state.

The web accessibility guidelines of WAI are recognized as world-wide standards. They are constantly improved by W3C and supplemented in accordance to investigated shortcomings and development of new technologies. One of the advantage of W3C standards is the active participation in their specification by various organizations, people with disabilities, government, and research laboratories from all over the world. For example, IBM introduced a checklist¹ which is compliant with WCAG.

¹<http://www-03.ibm.com/able/guidelines/web/accessweb.html>

An increasing number of countries considers web accessibility as a necessity, incorporating it into their regulations. In Austria, for instance, government websites are required to follow accessibility standards such as WAI. In the USA, the rights of people with disabilities are protected by Section 508 [233], which requires USA government websites to be accessible to all according to the amendments performed in 1998. In Russia, government standard GOST P 52872-2007 [86], enacted in 2009, defines the rules which should be taken into account to make web pages more accessible. The newly introduced regulations also forced companies to form divisions to monitor their compliance with the accessibility standards and to consider accessibility as an important aspect. For further information regarding the accessibility guidelines and standards, the interested reader may refer to [113, P. II, Web Accessibility and Guidelines].

Existing accessibility standards help web authors (content creators) and developers of web agents and assistive technology to provide convenient access to web resources for various groups of the population (including people who are blind). However, regardless of the significance of the accepted standards, they possess a number of shortcomings. For instance, in [137], B. Kelly et al. analyzed W3C WAI guidelines and identified some weaknesses such as: ambiguities in formulations, no mentioning of widely used Web technologies such as PDF and Flash, unreasonable complexity and much more. Furthermore, in their research [193], H. Petrie et al. collected home pages from 100 different websites for detailed evaluation by 50 people with a variety of disabilities. It was discovered by the authors that the participants encountered 585 identified problems, and among the identified problems, 45% of them were not in violation of any checkpoints listed by WAI. Inaccessibility of technically accessible web pages is also studied in [297].

In addition, there were several reproves done regarding the GOST P 52872-2007, in particular, unreasonable use of quantitative characteristics [213] (e.g., content of popular web pages should be in its size less than two-three screens of text; number of links on a web page should be less than 15) and incorrect use of some terms and concepts [196].

Accessibility of a web page can be automatically checked by the Web Accessibility evaluation tools, such as A-Checker² and WAVE³. It is worth mentioning that most of today's web pages do not follow accessibility guidelines and standards [137]. This is due to the fact that the vast majority of developers and managers are not experts in web accessibility technologies [137], and making web resources more accessible requires additional time and money. Therefore, in spite of the experimentally proven effectiveness of using header tags $\langle H1-6 \rangle$ [226, 292], less than a half of web pages actually introduce them [29]. Also, according to [29], only 56.9% of images on the web pages visited by the participants of the experiment had properly assigned alternative text. Thus, the Disability Rights Commission conducted an extensive user evaluation and reported that most websites (81%) fail to satisfy basic accessibility requirements [113, p. 63].

2.1.2 Hardware

Hardware used by people who are blind or visually impaired is mostly represented by keyboard, microphone, speakers and Braille display. A keyboard is used for interaction with the computer:

²<http://checker.atrc.utoronto.ca/>

³<http://wave.webaim.org/>



Figure 2.1: Braille Edge 40 with 40 cells (<http://www.hims-inc.com/products/braille-edge-40/>)

typing text, sending commands. It plays the role as a *tactile input*. A microphone (*aural/speech input*) is also used for the same function as a keyboard, when a *speech recognition* software is applied (e.g. Sphinx, Microsoft Tellme speech engine, Dragon Dictate). Providing *aural output*, speakers are used for listening to audio information including the textual information which is transformed into speech by means of *text-to-speech engines* (e.g. FreeTTS, eSpeak, Festival). The work with Braille display (see Figure 2.1) requires knowledge of Braille code. This hardware establishes *tactile output* and therefore enables a blind person to read information thanks to the tactile perception.

Thus, there are two main information channels used by the blind user during her interaction with the computer: aural-speech and tactile. Their mutual use increases the effectiveness and efficiency of the user compared to work without Braille display (tactile channel) [56]. Braille display is mostly helpful while reading complex text, special characters, formulae, and for spell checking. Sometimes blind users use only Braille devices when they are tired of listening to their screen readers [56, p. 20]. This fact was also confirmed by one of the cooperators of the ABBA project [236], of which the author of this thesis took part.

Taking into account various output devices, W3C provides standards which enable developing web documents for various media types [276] (e.g. Braille display) [260, Sec. 7] and interfaces (e.g. speech interface [253, 272]).

It is worth mentioning that according to the inquiry conducted by WebAIM in 2012 [296], 72% of respondents do not use Braille display and mainly rely on the audio output⁴.

2.1.3 Software

Typhlotechnology software includes screen readers, specialized browsers, web adaptation and transcoding systems, speech recognition systems, optical character recognition systems, spelling and grammar checkers, etc. Most of these technologies are integrated in contemporary operating systems by default (e.g. Windows 7, Ubuntu 13.04, OS X 10.9). Of particular interest is Vinux, a specialized operating system based on Ubuntu, optimized for the needs of blind and partially sighted users. However, integrated typhlotechnologies provided together with operating systems out of the box are usually quite limited.

⁴Similar results are acquired in the survey conducted by the author of this thesis, see Section B.2.

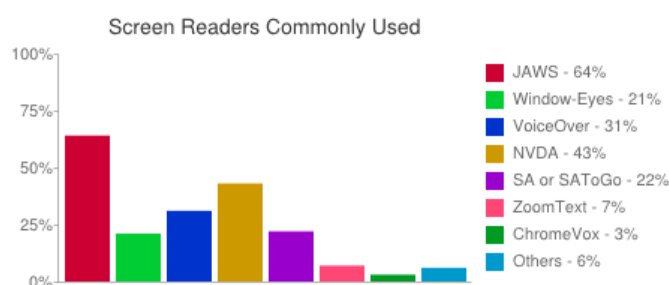


Figure 2.2: “Which of the following desktop/laptop screen readers do you commonly use?”. The inquiry conducted by WebAIM in 2012 [296]

In this section, we consider technologies and products most relevant to this thesis. Other surveys of existing assistive technologies can be found in [35, 113].

Screen Readers

A screen reader is a computer software that enables a visually impaired user to read the contents of a visual display thanks to the provided spoken feedback [201]. These systems have been developed since the 1980^s and are oriented on various information representations (Word, PDF, X/HTML, DOM, etc.) and operating systems. Two main modes of operation can be distinguished for the screen readers: *view mode* (for the navigation through the elements on the screen) and *edit mode* (for the interaction with elements on the screen, e.g. filling the forms). In this work, we consider screen readers that provide access to the web page content. Among them, there are desktop screen readers such as JAWS⁵ [90], Window-Eyes [178], Apple’s VoiceOver [11], NVDA [188], COBRA [20], SuperNova [60], Fire Vox [47] (Firefox browser extension), and web screen readers (which have their main functionality implemented by the corresponding web services) such as System Access to Go [210] and WebAnywhere [32]. The most commonly used tools⁶ are illustrated in Figure 2.2 according to the inquiry performed by WebAIM in 2012 [296]. These systems are primarily based on the analysis of the DOM trees (or source code) of a web page without providing information about the logical structure. This binds blind users to navigate web pages based on the concepts tied to the HTML tags, for instance, H1–6, TABLE, P, A, which are very often misused or not used at all [29, 30, 137, 226, 292]. For example, tables are very often (about 88% of the cases) used to organize the layout of a web page and the alignment of web page elements, but not to organize tabular data [146]. Therefore, web pages not compliant with accessibility standards are not accessible even via contemporary screen readers. Thus, supplementary analysis of the web page layout is required.

Due to this fact, VoiceOver performs some limited analysis of web pages with their Web Spots technology. It considers visual design of a web page and conveys additional virtual tags (web spots) which can be used as landmarks to make navigation on a web page more convenient and effective. VoiceOver also provides visual references to enable blind and sighted users to work

⁵JAWS uses term “virtual cursor mode” for the view mode and “forms mode” for the edit mode.

⁶Commonly used screen readers are also presented in a survey conducted by the author of this thesis in Section B.2.

together. For instance, the blind user can navigate a web page using the touch-sensitive trackpad which reflects its visual formatting (layout). The current focus is highlighted on the screen.

Most of the contemporary screen readers handle CSS 3 Speech Properties [272] as well as WAI-ARIA markup for AJAX live regions [259]. However, Fire Vox goes beyond that with its support of AxsJAX scripts [48] which enable injection of WAI-ARIA-based accessibility into AJAX applications.

In spite of the fact that screen reading technology aspires to incorporate the latest standards introduced by W3C, they are still lagging behind [67, 158]; this obliges developers of web pages to take into account not only accepted guidelines, but also the ability of contemporary screen readers.

Specialized Browsers and Browser Augmentation

“A *specialized browser* is one which is designed to render Web pages in a non-visual form” [113, p. 153], therefore they are also called *non-visual browsers*. These web browsers analyze loaded web pages and provide an interactive interface to the blind user with audio output. The first specialized web browsers that spoke web content emerged in early 1995. In contrast to the screen reading technologies which are limited by the application programming interface (API) provided by the web browser, specialized browsers have more control over web page rendering and representing it in a more convenient form for the end user. These results can also be achieved via browser augmentation. For example, by the browser extensions, add-ons, or bookmarklets, which also have access to a rich API of the web browser. Similar to screen readers, specialized browsers and browser augmenting software provide two modes of operation: view and edit.

Examples of the earlier systems are PWWebspeak (add-on for the Netscape browser), BrookesTalk [303], Lynx, Home Page Reader (plug-in to Internet Explorer) [13]. Examples of the later specialized browsers are Emacspeak [202], Goal-oriented web browser [66], and HearSay [36,222] together with its extensions, such as CSurf [171] and Dynamo [35]. Emacspeak is more than a browser, it is a speech interface and an audio desktop. It provides task specific interfaces, for instance, for making search queries in the Web, checking a Google calendar, weather, etc. A goal-oriented web browser enables a user to automate its browsing interaction. HearSay together with its extensions interacts with the web browser, such as Firefox and Internet Explorer, via corresponding extensions and conducts an automatic analysis based on the DOM tree and computed CSS attributes. The analysis comprises web page segmentation, capturing dynamic changes on a web page, identifying relevant context on a new web page based on the link activated, recommendation of labels for the web form elements, and so on. A user dialog interface is provided based on vxmlSurfer which utilizes VoiceXML [253].

A detail survey of specialized browsers can be found in [113, P. III, Specialized Browsers].

Web Adaptation and Transcoding

In the field of Web Accessibility, *web adaptation* is aimed at transforming a web page into accessible representation according to the user’s needs and abilities [5, 223]. The automatic modification of the original content before it reaches the end user, performed “on-the-fly,” is

usually called *transcoding*. This technology with its application to web accessibility has matured from circa 1992 [113, P. III, Transcoding].

Some representative examples of transcoding systems include Dante [302] (annotation-based transcoding system which is characterized by a metaphor for non-visual physical navigation in space [97]), eAccessibilityEngine [5] (user-adaptable system which takes into account personal disabilities and enables application of corresponding sequence of transformations over web pages for ameliorating their accessibility), WebInSight [30] (automatically adds alternative text to images based on the context analysis, optical character recognition techniques and manual labeling), SADIe [24] (rule-based and annotation-based transcoding system which utilizes CSS style sheets for distinguishing semantics of partial contents on a page), transcoding based on the Spatial Graph Grammar (SGG) [144] (grammar-based approach with its application for web page adaptation for mobile devices), and AxsJAX [48, 168] (a framework which allows transcoding web pages by injecting them with ARIA metadata).

Most of the specialized browsers, such as HearSay and Lynx, perform a procedure of web page transcoding. For example, as stated in [35, p. 6], HearSay algorithms can also be used on a transcoding proxy.

Web Automation

Web automation is “a technology that reduces the human effort necessary to perform typical web browsing activities such as form filling and clicking links, by performing these actions on behalf of the user” [198]. Web automation is usually performed by manual invocation of the necessary script, presented as a sequence of actions, or by a certain event. A script can be written manually or automatically derived, for instance, with a technique such as *programming by example (PBE)*.

Some representative examples are Creo [66] (PBE module integrated into Goal-oriented web browser which allows users to generate a general-purpose script with a single example), CoScripter [159] (PDE-based system which enables users to record, play back, edit, and share generated scripts; the latter are both human- and machine-understandable), TrailBlazer [31] (it assists blind users in performing certain web-based tasks while dynamically generating non-visual interface based on the database of scripts provided by CoScripter and a relevant script is suggested to the user based on a provided short task description), Ubiquity [65] (a multilingual textual interface for the Firefox browser enabling rapid information retrieval and task execution), CoCo [157] (framework which provides the user with necessary scripts and assists her with web tasks, utilizing ActionShot and shared database of CoScripter; the scripts are provided according to the short textual commands of the user), etc.

In [199], Y. Puzis et al. introduce a model-based approach that can predict the next browsing action of the user taking into account her browsing behavior.

It is worth mentioning that navigation and navigation interface are crucial functional components of the contemporary Web-oriented typhlotechnology. In the next section, we will consider concepts of web navigation, primarily, web page navigation, briefly mention the main approaches and principles, and describe the essential problems resolved in this thesis.

2.2 Web Page Navigation

Navigation is the most important aspect in Web Accessibility and requires special attention. According to the theory of cognitive walkthrough for the Web [34] (a general-purpose usability inspection method for websites), the navigation actions of the sighted user within the web page can be divided into the following two steps and represented as processes: “Step one is an attention process that (visually) parses a web page into subregions and attends to the subregion of the page that is semantically most similar to the user goal. Step two is an action selection process that selects and acts on a widget from the attended-to subregion, the widget semantically most similar to the user goal” [226, p. 2–3]. Thus, navigation actions of the sighted user on a web page is mainly based on a hierarchical nature. Sequential access to the information usually takes place when regions cannot be further split (for instance, a paragraph of the textual content).

In contrast, due to the peculiarities of the aural information perception and the technologies used (see Sections 2.1.2 and 2.1.3), web page navigation has a one-dimensional (sequential) character for the blind. This fact is also mentioned in [226]. Thus, sequential navigation is realized in contemporary screen readers and specialized browsers as the main navigation method. Braille display also contributes to the sequential navigation and usually exposes the text corresponding to the current focus on a web page.

2.2.1 Contemporary Realized Navigation Approaches

For efficient navigation, contemporary screen readers and specialized browsers follow the accessibility guidelines (see Section 2.1.1) which force web developers to expose semantics of a web page’s elements within the source code (using HTML tags and attributes properly, leveraging ARIA roles). Regarding the source code of a web page, screen reading technologies usually serialize a DOM tree representation according to the *tree order* [260, App. E] (corresponding to the depth-first traversal) and provide an access to web page content for blind users based on this serialization. The user is also provided with relevant navigation API which can be accessed via the keyboard short keys (e.g. for NVDA, JAWS), trackpad (e.g. for VoiceOver) or microphone (e.g. Microsoft Tellme speech engine). Commands of the view mode include navigation based on the element type (defined by the name of HTML tags, attributes and their ARIA roles), possibility to jump across the text lines (e.g. reading every tenth line of the text) and text-level navigation (e.g. for reading sentences, words, and letters).

In [226], H. Takagi et al. propose an “ideal mental model” of the blind user, which can be formed if a web page is authored according to the accessibility standards, and if the blind user applies corresponding navigation commands. For instance, with the sequential representation of web page elements, a proper use of heading tags can help the blind user to navigate among subregions and build an adequate logical model of a web page. As shown in Figure 2.3, thanks to the heading tags, a blind user can easily recognize boundaries of web page fragments and navigate through them. Thus, in contrast to visual navigation, non-visual navigation is an effort to derive a fragmented (possibly hierarchical) representation of a web page from its sequential representation.

Contemporary screen readers also have a possibility to navigate tables row by row, column by column, and cell by cell as well as lists, item by item. This navigation is also provided according

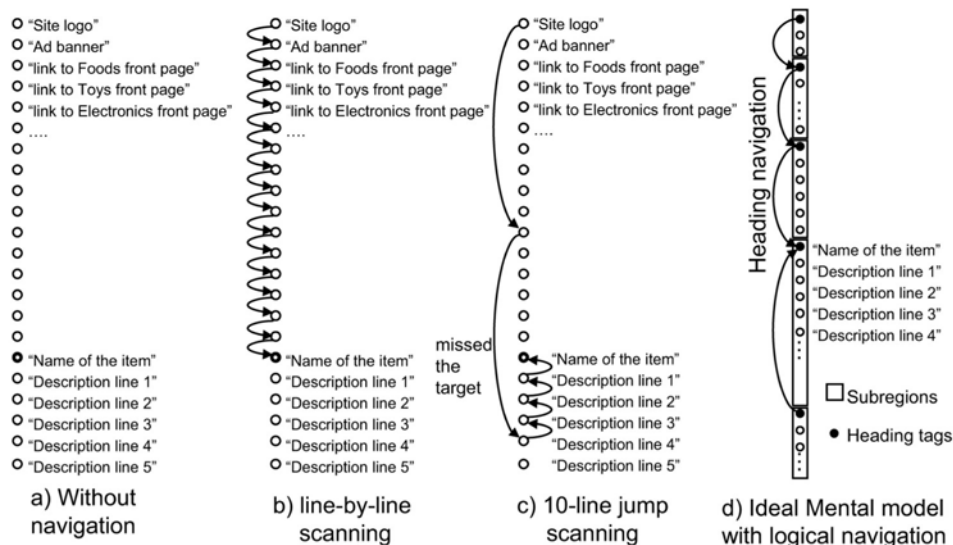


Figure 2.3: Mental model of a web page and models of scanning navigation [226]

to the source code of a web page. That is, locomotion is done over HTML tables and HTML lists.

There are technologies which give the visually impaired user a possibility to investigate spatial allocation of various elements on a rendered web page. For instance, as was mentioned in Section 2.1.3, VoiceOver additionally provides a possibility to navigate through the web page via a touch-sensitive trackpad which reflects visual formatting (layout) of a web page. W3C provides a specification of the focus navigation in its working draft [267, Sec. 9.2.2]; it enables establishing four main navigation directions for each element: up, right, down, and left.

We would like to mention the work of Y. Borodin et al. [37], where the authors survey various screen reader browsing strategies.

2.2.2 Metaphor of Spatial Navigation

Regarding the spatial allocation of objects, it is important to note works which consider web page navigation from the viewpoint of spatial navigation. Regardless of the disability, A. Dieberger [59] proposes the use of a city metaphor to support navigation in complex information spaces. In the workshop report [132] of CHI'97, the authors clarify definitions of navigation in electronic information environments and consider such a navigation from the viewpoint of the psychology of navigation in the real world. In [97, 112], C. Goble, S. Harper, and R. Stevens draw a similar analogy between navigation in the Web and navigation in the real world for visually impaired people. The authors introduce the main mechanisms and principles in web navigation as well as describe problems which blind users face. A general travel model presented by the authors and which can be applied for web navigation is presented in Figure 2.4. In [97, 112], the authors utilize the term *mobility* which "is defined as the confident ease of movement within the environment and the accuracy of navigation." "*Environment* is the context that the traveler travels through and includes the way the landscape is rendered and perceived." *Navigation* "suggests an ability to

move within the local environment from point A to point B, either by the use of pre-planning using maps or fore knowledge, or by navigating “on-the-fly.” The authors also present concepts such as *mobility techniques*, *mobility objects*, and *mobility principles* (see Figure 2.5). Mobility techniques refer to main actions performed by the blind users when navigating a web page. Mobility objects are various web page elements which play different role in the process of interaction of the user with the web page. Mobility principles represent the application of principles of real word mobility of blind users to the information space. The authors also make a comparative analysis of web navigation by the sighted user and the blind user and highlight the main principles which should be taken into account by the designers of hypermedia and user agents. These principles of web mobility are incorporated in corresponding guidelines [112] which mainly require web developers to explicitly expose landmarks and navigation cues which can be further used by the blind for efficient navigation through the web page content according to its logical structure.

2.2.3 Conclusion on Web Page Navigation

In spite of the fact that a lot of effort was made in the area of web accessibility, contemporary screen reading technologies are still not efficient and satisfactory enough. As shown in [226], blind users are more than ten times slower than sighted users. Shortcomings of the screen readers are also investigated in [29].

Thus, based on the analysis of the modern typhlotechnology and approaches of navigation, we distinguish three crucial groups of problems within the area of Web Accessibility, with which we propose some solutions relative to the scope of this thesis:

I. Information Perception Technology. One of the issues includes information flow for blind users, since listening to text is slower than scanning it visually [97]. We believe that this problem could be rectified in the future by introducing new technologies which could leverage aural and tactile perception more effective, however it is out of the scope of this thesis.

II. Semantic Metadata. Another factor is the lack of semantic information about objects read and their functional role. This is a problem not only because the majority of web pages do not follow accessibility guidelines (see Section 2.1.1) [29, 113, 137], but also because current accessibility standards do not provide the possibility to express semantics of complex web objects and their semantic relationships in a form convenient for the blind user.

A challenge of converting inaccessible web pages into accessible ones compliant with accessibility guidelines relates to web adaptation and transcoding technologies (see Section 2.1.3). This process includes *analysis* of a web page, *identification* of inaccessible patterns to be corrected, and *annotation* or *transformation* of the inaccessible content (*content re-packaging*). Problems of analysis of a web page, its understanding, and identification of various design patterns are mainly addressed in the areas of WPU and WIE (see Section 2.3). In this thesis, we consider problems of enriching web page content with semantics from the viewpoint of WPU and WIE applied for different representations of the web page. We also pay a great deal of attention on

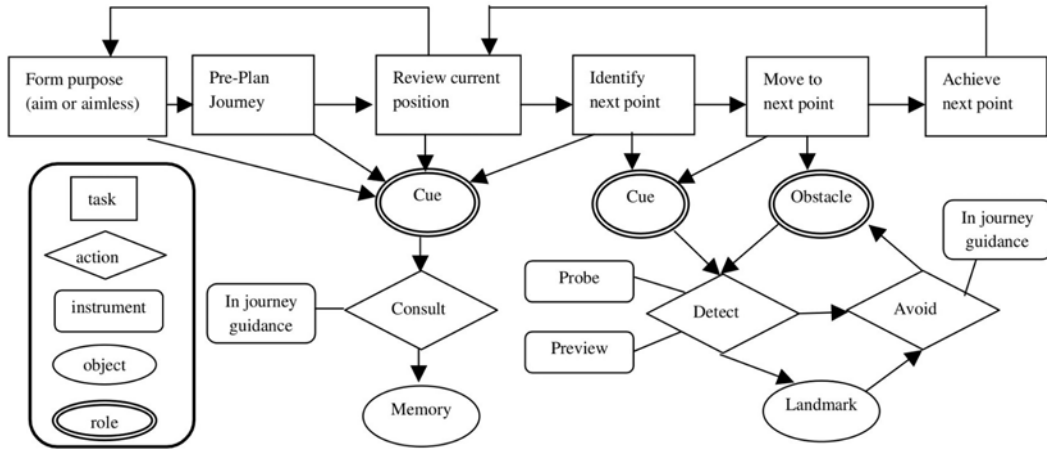


Figure 2.4: A flow-based travel model [97]

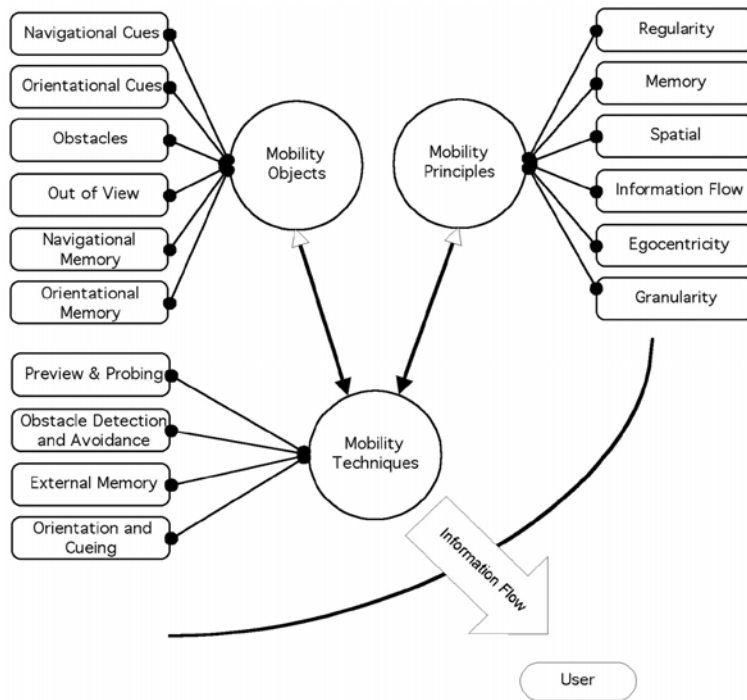


Figure 2.5: Combined techniques, objects, and principles enhancing web page mobility [112]

various representation models which can be suitable for web page processing and enhancing web accessibility (see Section 2.4).

We consider the challenge of providing semantically rich information for web page content as a problem related to Semantic Web technology (see Section 2.5). In particular, providing vocabularies with semantic metadata and materializing corresponding ontological concepts and relations (see Section 2.6). Ontology enables the presentation of information in a form accessible both for humans and computer.

In regards to these issues, Chapter 3 is dedicated to modeling a web page layout, Chapter 4 introduces a unified ontological model describing a visual appearance of the web page, and Chapter 5 presents a WPPS framework for web page processing which is also used for ameliorating web page accessibility.

III. Navigation. As was mentioned, a model used for navigation in contemporary screen readers is in general represented by the serialized sequence of readable DOM nodes. Therefore, such a navigation does not reflect web page logical models established primarily by the web page layout, that causes incorrect understanding of a web page content by the user of a screen reader. Current navigation methods provide a sequential access to various types of HTML elements, which is realized by a rich set of commands. However as practice shows, average users use a small set of functions of their screen reader, since the mental load of using a browser together with assistive technology and forming relevant mental model of the web page is very high [226, 234].

Navigation models and methods play an important role in quick and correct understanding of a web page logical model and information search. We believe that navigation processes should be further investigated and enhanced as well as integration of the one-dimensional navigation with the principles of spatial perception. Thus, we develop an alternative advanced navigation model and navigation methodology introduced in Chapter 6.

2.3 Web Page Processing

In this section, we introduce the umbrella term “*web page processing*” for Web Information Extraction (WIE) and Web Page Understanding (WPU). This concept was loosely inspired by Document Understanding [2, 110, 116, 141, 228], a field of research where the term “*document processing*” is related to the analysis and understanding of mainly raster (scanned) documents (other formats, such as formatted textual and PDF, are also considered). Web Page Processing (WPP) plays an important role in research fields such as Information Search [19, 187], Web Data Mining [124, 163], Web Adaptation [102], Web Accessibility [113, 171], Business Intelligence [23], Information Integration [28], and so on [87].

In Section 5.1, we give a definition of WPP in the application to the Unified Ontological Model (UOM) developed and proposed in this thesis.

2.3.1 Web Information Extraction

Web Information Extraction (WIE) is related to the identification of the relevant facts on a web page and their representation in the structured form. The problem of information extraction

can be considered as a problem of querying unstructured information resources and acquiring the structured results. The main means of assessing the effectiveness of a WIE system are *precision* and *recall* [177], adopted from Information Retrieval. Thus, the challenge of developing WIE, which within the required time interval extracts information from certain sources, with the required precision and recall is of high concern. For realizing WIE, researchers and developers apply different methods and approaches, including data mining and machine learning [164], logic programming [22], automaton-based methods [128], various heuristics [6, 169], as well as approaches based on Natural Language Processing (NLP) [55] and ontologies [126, 185]. Overview of various WIE and primarily Web Data Extraction (WDE) methods, approaches, and tools can be found in [45, 87, 152, 156, 209].

WIE refers to the well-known fields of research such as Information Extraction (IE) and WDE. Examples of IE tools are NoDoSE [1], RAPIER [44], and Crystal [218]. IE methods extract relevant facts, such as events, appointments, and quotations, from textual content presented in natural language [179]. It is worth mentioning that classical methods of IE, which are mainly dedicated to the analysis of plain text and based on NLP techniques, cannot be directly applied to a web page. This is due to the fact that web pages usually have a complex structure with elements possessing various semantic roles (e.g. navigation menu, table, main content, link, button) and providing different functionality (e.g. calendar, link, button, element with drag-and-drop function). Furthermore, contemporary web pages are web applications with rich interface and thus cannot be treated as formatted textual documents. Therefore, IE methods applied to web pages require a supplementary analysis of the web page structure. Due to the presence of multi-media content, IE from web pages can be accompanied with the application of OCR (optical character recognition) methods. The structure of a web page can be presented by its source code, DOM tree or visual representation. Various web page representations leveraged in WPP are considered in Section 2.4.

WDE targets data to be identified on a web page, for example, the name and price of a product, timetable of a flight, or opening hours. WDE methods mainly analyze the structure of a web page operating over source code, DOM tree or visual representation (see Section 2.4) and have very limited analysis of textual content which usually boils down to the application of regular expressions. The majority of contemporary web pages and mainly those from the Deep Web [25] are generated by the user request (“on-the-fly”) based on the data stored in the back-end databases. These systems are usually called web content management systems. Thus, the problem of extracting web data is very often considered as an issue of data records extraction from a certain database with unknown schema and mapping it into the database with a known schema. Examples of tools realizing this approach are IEPAD [46], ExAlg [12], and DEPTA [304].

WIE consists of two main phases: *wrapper induction* and *direct information extraction* (or wrapper application). A *wrapper* is a template, description, or program for extracting relevant data or information. A wrapper is created during the first phase. It can be performed manually, in semi-automatic, or automatic manner [45]. Wrappers reflect intrinsic (e.g. color, part of speech, HTML tag) and relative (e.g. alignment, sequence of elements within the source code, position in the DOM tree) features of information to be identified. However, the most common techniques are based on absolute or relative position of the required information object within some structural representation of a web page (i.e. source code [57], DOM tree [98], graph-based structure [309], etc.). In the second phase, the wrapper is applied for the certain set of web pages. Extracted

structured information as the result of this phase is then integrated into other applications.

2.3.2 Web Page Understanding

Web Page Understanding (WPU) is related to understanding the logical structure of a web page and its elements. The most common challenges include segmentation of the web page on logically consistent blocks [41, 95, 104, 300], search form understanding [62, 91, 138, 305], and web page labeling [187]. All these methods are used for improving effectiveness of information search [118, 187] and enhancing web accessibility [104]. Very often a web page understanding process is incorporated into the information extraction methods. For example, L. Li et al. in [160] and also W. Liu et al. in [166] adapt VIPS, a web page segmentation algorithm, for data records extraction. Moreover, metrics such as precision and recall used in WIE are also applied in WPU [163, 169, 305]. Regarding Web Accessibility, we would like to note the work of H. Guo et al. [104], in which the authors introduce a web page segmentation algorithm to make web pages more accessible for the blind [171]. Y. Yesilada et al. [301] leverage VIPS-based segmentation algorithm in transcoding web pages based on the eye-tracking data for the users of small screen devices and disabled people. The importance of the correct fragmentation of a web page for its accessibility is indicated in [112].

WPU has a relation to Document Understanding, which has a similar goal but a different object of research—raster document. Some techniques applied for scanned documents can also be applied for web pages after certain necessary modifications. For instance, the XY-cut algorithm [106, 176, 182] invented for document segmentation was adapted for web pages [95, 150]. Moreover, some table recognition algorithms for web pages are based on the knowledge acquired from the field of Document Understanding [94, 95, 117]. Certain methods are also applicable both for scanned documents and web pages [144]. These facts are also confirmed by using similar structures both for scanned documents and web pages (see Section 2.4.7).

2.4 Web Page Representations

In this section, we focus on various web page representations established by W3C standards as well as models and structures used in various tasks within the fields of WIE and WPU (see Section 2.3). We also give an answer to the question: “are there any universal models suitable for developing various methods for web page processing?” Thus, such a model should have a versatile character, providing the developer with the possibility to investigate various peculiarities of a web page to be able to develop effective and efficient methods for various purposes such as WIE, WPU, and Web Accessibility.

In Section 2.4.1, we discuss the standard forms of web page representation introduced by W3C and actively used in web technologies. In Section 2.4.2, the most popular methods which work on the source code level are discussed. Section 2.4.3 pays attention to some of the prominent approaches which consider a textual content of the web page and ignore tags. Section 2.4.4 presents tree-based models of a web page and gives examples of methods which operate on these representations. Section 2.4.5 describes CSS 2.1 [260], a W3C standard defining a rendering process in the web browser, and the CSS Object Model (CSSOM), the main visual component of

the rendered web page. Section 2.4.5 provides us with information necessary for the analysis conducted in Chapter 3 and the development of the UOM introduced in Chapter 4. Section 2.4.6 and Section 2.4.7 discuss the main quantitative and qualitative characteristics of the web page and corresponding models of the web page leveraged in different works. In Section 2.4.8, we conduct an analysis of existing models and substantiate the necessity of developing a new unified model (see Chapters 3 and 4) and a web page processing system (see Chapter 5).

2.4.1 Standard Forms of Web Page Representation

We distinguish the three main representations of a web page which are considered suitable for analysis and information processing: *source code*, *DOM tree*, and *visual representation*.

Each of these representations has a certain purpose. **Source code** is a convenient form for preserving web pages in the file systems and transferring it through the Web. It is specified by the following W3C standards:

- HTML [241]—the HyperText Markup Language for authoring web pages;
- XML [256]—the Extensible Markup Language (XML), a subset of the Standard Generalized Markup Language (SGML), for representing structural information in the form of XML documents which are both machine-readable and human-readable.
- XHTML [246]—the Extensible HyperText Markup Language, a family of XML markup languages, which represent HTML in the form of valid XML Documents.

A web page in the form of source code is created by the web developers manually or automatically with the help of web authoring tools, such as KompoZer⁷, Adobe Dreamweaver⁸, and Amaya⁹. Methods applied to this form of web page representation are described in Section 2.4.2.

The **DOM tree** is essentially used for the automatic processing and querying with the help of existing XML technologies, such as XPath, XSLT, and XQuery. This representation is standardized by the W3C specification:

- DOM [247]—the Document Object Model (DOM), cross-platform and language-independent interface for accessing, manipulating, and representing HTML, XHTML and XML documents.

The salient methods corresponding to this presentation are presented in Section 2.4.4.

Visual representation enables the sighted end user to see a web document in the convenient form. It contains the most valuable information defined by the web designer, and it is oriented on the end user. W3C provides the following standard for this representation:

- CSS¹⁰ [260]—the Cascading Style Sheets, is a style sheet language used for describing the presentation semantics of a document written in a markup language.

⁷<http://kompozer.net>

⁸<http://www.adobe.com/products/dreamweaver.html>

⁹<http://www.w3.org/Amaya/>

¹⁰All CSS specifications are presented at <http://www.w3.org/Style/CSS/Overview.en.html>

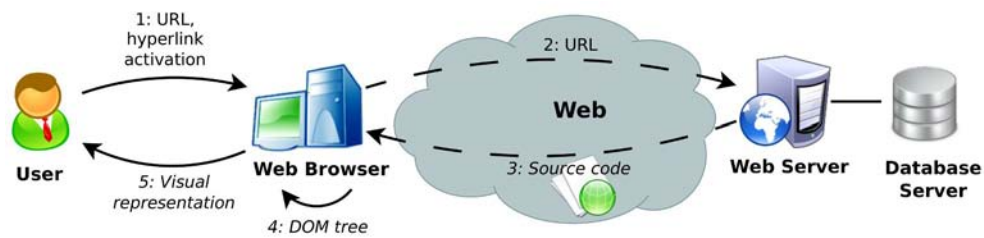


Figure 2.6: Data flow in the generalized representation of the process of querying a web page by the user

Section 2.4.5 gives a detailed description of this standard. Section 2.4.6 does a brief survey of existing methods that take into account quantitative visual and spatial characteristics. Section 2.4.7 discuss methods and various qualitative models of the rendered web page presented by different researchers.

These standardized representations play an important role when the user surfs the Web. Figure 2.6 illustrates the process of the user request of a web page. **1.** The request can be initiated by entering a URL explicitly in the web browser, clicking a link, or by sending the request via web forms (for instance accessing information in the Deep Web). **2.** Then, the web browser sends the request to the Web. **3.** A target web server received the request returns the web page in the form of source code (i.e. X/HTML, XML). The client (the web browser) in turn acquires the web page and sends additional requests to get files linked with the current web page: multimedia files, CSS style sheets, scripts, other web pages (e.g. related via frames), etc. **4.** Based on the files received, the web browser builds a DOM tree, or DOM trees if there are other web pages related to the requested one. (The number of trees corresponds to the number of web pages downloaded.) For the constructed DOM trees, a web browser applies all necessary program scripts, which in turn can change the generated DOM trees as well as request for uploading supplementary web documents. **5.** For the DOM tree generated, a web browser visualizes web page elements. It is worth mentioning that contemporary web browsers (e.g. Firefox, Safari, Internet Explorer, Opera) visualize a web page in conjunction to uploading web resources and generating DOM trees.

2.4.2 Source code

The source code of a web page is usually represented by HTML [241] and XHTML [246], and less often by XML [256]. These forms of web page representation are convenient for preserving them on the web servers and transferring them through the Web. These languages are used to mark a text with the specific tags which can possess certain semantics and functional roles. XML is dedicated to structural formatting information and constructing structured documents, whereas HTML and XHTML with their predefined set of tags, are utilized for semi-structural formatting and their main application is authoring a web page. Thus, tags defined in X/HTML have specific presentation semantics and interface related functional roles. For instance, `TABLE` tag is used for forming a table layout, `A` tag denotes a hyperlink. In contrast to X/HTML, XML tags do not possess such a semantics, however, it can be additionally defined, for example, by means of XML

Schema, CSS style sheets and JavaScript.

IEPAD [46] developed by Ch.-H. Chang operates on the source code level for HTML web pages. It implements automatic approach which does not need training examples. The approach is based on the fact that the repetitive logical structures, web objects (such as lists of products on a web page of an on-line shop, list of posts in the forum, etc.), in most cases have repetitive structural elements in the source code. The reason for this is that they are generated with the use of the same template. Thus, wrapper generation in IEPAD is based on the analyses of repetitions in the sequence of tags. IEPAD uses *PAT tree* for pattern discovery and *center star* approximation algorithm for multiple string alignment in the process of extraction pattern generalization.

RoadRunner [57] introduced by V. Crescenzi et al. is another automatic extraction system. In contrast to IEPAD, RoadRunner discovers regularities analyzing several data-intensive web pages, for instance, web pages of the same website. During the wrapper generation, the tool compares pairs of web pages for detecting similarities and differences in their structure. For the matching process, the authors of this system introduce *ACME (Align, Collapse under Mismatch) technique*. The extraction mechanism is based on a *union-free regular expressions*. RoadRunner is able to handle optional attributes and variety in the sequence order of attributes.

A. Arasu et al. [12] describe the process of generating a web page on the server side as a process of encoding the data selected from a database, with the application of a certain template. Thus, the authors give a formal definition to the structured data and propose a model for page creation that describes how data is encoded using a template. The model requires attributes to appear in the same relative position with respect to the values of other attributes in a tuple (data record). The authors also introduce concepts such as *equivalence classes* and *differentiating roles* which are utilized in the wrapper generation process. The approach is realized in the tool ExAlg [12].

It is also worth mentioning the following WIE tools operating on the source code level: WIEN [153], SoftMealy [128], DEByE [155], which are supervised wrapper induction systems, in addition to WebL [140], and TSIMMIS [109], which are frameworks for building web wrappers manually.

2.4.3 Textual representation

Textual representation is one of the forms of web page representation. By this form we mean a textual content of a web page. Examples of tools which build this representation are specialized web browsers such as Lynx¹¹, W3M¹², Links¹³, and Elinks¹⁴. These applications are sometimes used by blind users due to the accessibility of the textual content.

Textual representation is usually generated by the analysis of the source code. All markup is removed, and all textual content is laid out according to the presentation semantics of X/HTML tags itself (without consideration of the corresponding CSS style sheets). For instance, for the TABLE tag, space symbol and tabulation can be used for formatting table content as well as symbols like “|” and “-” to depict a table border.

¹¹<http://lynx.isc.org/>

¹²<http://w3m.sourceforge.net/>

¹³<http://links.sourceforge.net/>

¹⁴<http://elinks.or.cz/>

This representation is considered in the IE systems analyzing plain or formatted text and extracting facts (e.g. events, assertions) [19, Sec. 3.2], [55]. Wrappers of these systems can be represented in the form of logical rules, automaton, or templates. Most famous techniques for information extraction are based on regular languages, for example, regular expressions [45, 68]. They can be used for extracting date, time, price, etc. Information extraction systems have a direct relation to NLP [179]. Thus, the analysis of text can include techniques such as sentence splitting, part-of-speech tagging, named entity recognition, etc.

RegExpTokenizer developed by R.R. Fayzrakhmanov [69], [68, Sec. 2.7, Sec. 3.2] is an information extraction tool which leverages regular expressions and requires a user to manually define a wrapper. RegExpTokenizer provides the user with additional constructs which extend expressiveness of regular expressions. Thus, the user can impose additional constraints for the length and value (string or numerical) of the returned string. It is also possible to define new concepts specified by the extended language as well as use them in the definition of other concepts. Thus, a wrapper can be defined by the set of interdependent definitions which form a graph of dependencies, however, only an acyclic graph is allowed. All concepts defined earlier in wrappers can be reused in the definitions of new wrappers.

GATE¹⁵ [58] is a development environment for implementing and applying methods of NLP. It can work with various formats such as X/HTML, XML, RTF, and SGML which are converted into textual representation. GATE conveys the user with *JAPE (a Java Annotation Patterns Engine)*, which allows the user to apply regular expressions over annotations in documents. GATE has a set of predefined methods for sentence splitting, part of speech tagging, named entity recognition, etc. It also supports ontologies, provides means for machine learning and web crawling.

Other noteworthy NLP tools include LAPIS¹⁶, The Dragon Toolkit¹⁷, MontyLingua¹⁸, RAPIER [44], NoDoSE [1], Crystal [218], and OpenCalais¹⁹ web service.

Textual representation of a web page is out of the scope of this thesis, due to its orientation on the analysis of a web page's structural characteristics reflecting the logical structure. Various logical objects of a web page can possess different functional roles and contain text of different genre and topic. Thus, we are convinced that text analysis should be performed rather on the level of logical objects than the whole web page.

2.4.4 Tree representation

When referring to tree representation, we mean tree-like structures which are directly built based on the source code (markup), for instance, a DOM tree or a *tag tree*. A DOM tree is a tree which reflects the structure of the source code spelled in X/HTML or XML, and it is compliant with Document Object Model (DOM) [247] specified by W3C. DOM is a cross-platform and language-independent interface which conveys an API for accessing and interacting with valid HTML and well-formed XHTML and XML documents. In the DOM tree, nodes can be of

¹⁵<http://gate.ac.uk/>

¹⁶<http://groups.csail.mit.edu/uid/lapis/>

¹⁷<http://dragon.ischool.drexel.edu/>

¹⁸<http://web.media.mit.edu/~hugo/montylingua/>

¹⁹<http://www.opencalais.com/>

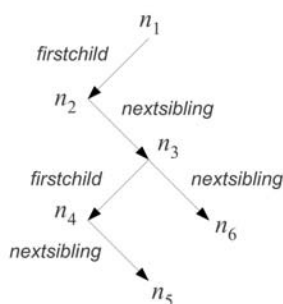


Figure 2.7: Unranked finite trees for web page representation [98]

different types including document (which is the root of the tree), element (which corresponds to tags in the source code), comment, and attribute. Usually for web pages, the majority of nodes are elements and text nodes. DOM can be accessed both by procedural and object-oriented languages. In the web browser, the DOM tree is built by the rendering engine (or layout engine), which also can deal with invalid HTML (for example, in case of the absence of required closing tags, or if the order of opening and closing tags is wrong). Thus, bijection between the source code and the DOM tree is only possible for valid well-defined source code and without application of scripts which can modify the DOM tree generated. Also of interest is Tidy²⁰, a specialized tools which deals with invalid code.

Tag tree is usually represented in the form of *unranked finite trees*, with the example illustrated in Figure 2.7. Nodes of such a tree usually correspond to tags in the source code (or element nodes in the DOM tree).

Lixto Visual Wrapper [23, 98] developed by R. Baumgartner et al. is a visual and interactive wrapper generation and data extraction tool. It enables the user to generate a wrapper by means of visual interactions indicating those web page elements which should be extracted. The user can additionally set constraints on HTML attributes and the position of the element to be extracted within the HTML table or HTML list. It is also possible to specify relations such as “before,” “after,” “contains” between elements. Based on the examples and constraints provided by means of interaction with graphical user interface (GUI), Lixto Visual Wrapper translates specification into *ELog program* which operates over the DOM tree. Thus, all relations and constraints are defined over the DOM tree. For example, relations “before” and “after” are computed according to the depth-first traversal over the DOM tree. *Elog* [22] is a logic-based declarative extraction language with Datalog-like syntax and semantics. It is an essential part of Lixto Visual Wrapper.

FiVaTech [135] developed by M. Kayed and Ch. H.Chang is dedicated to extracting objects with repetitive structure. The system analyzes a set of web pages represented by their DOM trees and builds a generalized model—*fixed/variant pattern tree*—proposed by the authors. This structure reflects the main interrelations between elements recognized during the analysis of the DOM trees. The authors propose methods for deriving from the fixed/variant pattern tree a template which underlies the corresponding web pages generated on the server side. A derived template is used further as a wrapper for the web pages generated according to this template.

²⁰<http://tidy.sourceforge.net>

The authors combine several techniques: alignment, pattern mining as well as the idea of tree templates for solving the problem of *page-level* [45] template construction. It is experimentally shown that FiVaTech has a precision much higher than ExAlg [12] (analyzing the source code, see Section 2.4.2) and is comparable with other *record-level* extraction systems like ViPER [214] (analyzing the DOM tree together with spatial characteristics expressed quantitatively, see Section 2.4.6) and MSE [307] (taking into account the DOM tree together with quantitative visual characteristics as well as interval relations, see Section 2.4.7).

The method proposed by B. Liu et al. in [164] and realized in the MDR tool is based on two assumptions: first, data records are usually presented in contiguous region (data region) of a web page and encoded using similar HTML tags; second, data records of the same contiguous region usually have the same parent node. The authors experimentally prove that MDR is more effective than IEPAD [46] (which analyzes the source code) and Omini [39] (which operates on a tag tree of a web page).

OXPath [92] is an efficient language for WDE and interaction with web pages. It extends XPath 1.0 and supplements it with additional functionality which enables it to specify various interactions with DOM elements. For instance, it is possible with OXPath to fill out the forms and navigate through the web pages. OXPath has additional axes such as web form elements axis and links axis. OXPath also provides an access to the computed CSS attributes.

It is also worth mentioning WDE tools such as DeLa [290] and Thresher [125] as well as the method for record extraction presented in [177].

2.4.5 CSS Specifications for Rendering Web Pages

Cascading Style Sheets (CSS)²¹ [211] developed by the W3C organization [239] is a style sheet language used for describing the presentation semantics of a document written in a markup language, such as HTML, XHTML, or XML. Thereby, CSS is aimed at different types of *documents* [260, Sec. 3], *media types* (e.g. “screen,” “print,” “braille” [260, Sec. 7], “3d-glasses” [276]), and *user agents* (“programs that interpret a document written in the *document language* and applies associated *style sheets* according to the terms of *this* specification” [260, Sec. 3]). However, according to the aim and objectives of this work (see Section 1.3), we consider CSS with regard to a web page as a document, screen as a media type, and a web browser as a user agent.

CSS is designed both for the developers of web browsers and creators of websites (*web authors*). It allows the potential to develop and convey a web resource to the user in the form which looks and behaves the same on different platforms and web browsers. This language gives the web author a possibility to control the process of visualizing the elements of a DOM tree with the web browsers. It is known to be good practice to separate document content from the document presentation, a CSS style sheet—collection of the CSS rules. It makes web authoring and maintenance of websites more efficient and simple. The presentation semantics of this language covers elements such as layout, colors, and fonts.

Currently, there are four levels of CSS. Each CSS level builds upon the last, typically extending it with additional features and concepts. In this work, we refer to the most recent

²¹<http://www.w3.org/Style/CSS/Overview.en.html>

CSS recommendation of level 2 revision 1 (CSS 2.1) from the 7th of June 2011 [260] which is implemented in most contemporary web browsers. It is based on CSS 2 which in turn is based on CSS 1. CSS 2.1 is an improved version of CSS 2; it does not contain unsupported or not fully interoperable features and additionally covers already implemented browser extensions. In contrast to CSS 1, CSS 2.1 supports media-specific styles that allow web authors to tailor the presentation of their documents to visual browsers, aural devices, printers, braille devices, etc. It also supports positioning of the content, table layout, lists, and internationalization. Most of the specifications of level 3 and all of level 4 are drafts and can become a W3C recommendation in the future. In contrast to “CSS Level 2” which is represented as one document, CSS 3 and CSS 4 consist of several modules which address functional aspects of document rendition and layout. Work of W3C on CSS 4 address future problems which are not covered by CSS 3.

We give some definitions according to the CSS 2.1 specification, which is important for this thesis.

A *CSS style sheet* is a set of statements that specify presentation of a document.

The rendering of a web page is performed on the *canvas*. The term *canvas* describes “the space where the formatting structure is rendered” [260, Sec. 2]. It represents the space where all elements of the DOM tree are visualized. The canvas is infinite for each dimension of the space, however the rendition of the web page is performed according to the size of a viewport.

A *viewport* is an important component of the web browser for viewing web resources. A viewport is a viewing (rectangular, as is usual in practice) area on the screen through which users consult a document [260, Sec. 9]. If the user changes the size of the viewport, a web browser’s engine may change the document’s layout. Moreover, if the size of the viewport is smaller than the canvas of a web page, the web browser should offer a scrolling mechanism.

There are various *units of measure* which can be applied for the elements of a web page [275]: distance units (relative and absolute lengths), angles, times, frequencies, and resolutions. However, for the identification of boxes’ location on the rendered web page, absolute length units, such as centimeters, millimeters, inches, pixels, points, and picas, are important. For most web browsers, a *pixel* is a common unit of measure which can be represented by the real number.

In the recommendation [260, Sec. 4.3.2] of CSS 2.1 and candidate recommendation [275, Sec. 5.2] of CSS 3, a pixel unit is related to the *reference pixel*. It is the visual angle of one pixel on a device with a pixel density of 96 dpi and a distance from the reader of an arm’s length. For a nominal arm’s length of 71 cm (28 inches), the visual angle is therefore about 0.0213 degrees. For reading at arm’s length, 1 px thus corresponds to about 0.26 mm (1/96 inch) [275].

Other basic concepts and definitions can be found in [260, Sec. 3].

Visual Box Model

The CSS box model [260, Sec. 8] describes the rectangular boxes that are generated for elements of the DOM tree and laid out according to the *visual formatting model*. Boxes consist of four main components such as 1) a *content*, 2) *padding*, 3) *border*, and 4) *margin* areas. Each of the box’s areas comprise the four segments: top, right, bottom, and left. Figure 2.8 illustrates box model and the terminology used to refer to pieces of margin, border, and padding, for instance, top margin (TM), right border (RB), and left padding (LP). The perimeter of each of the four areas is called an *edge*. The areas can be degenerated into the empty region.

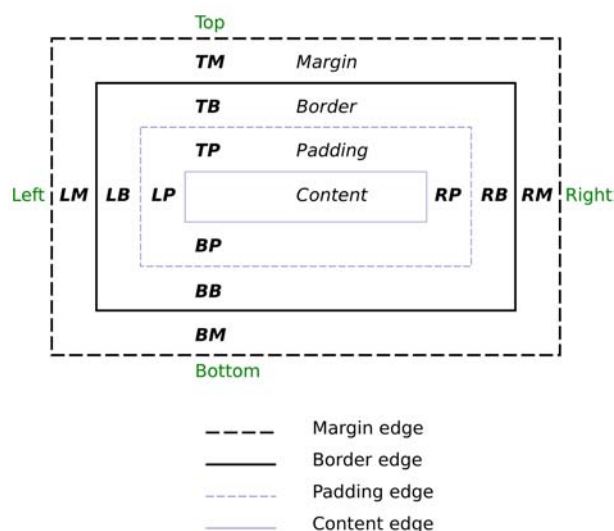


Figure 2.8: CSS box model

The *content* has a rectangular shape. It carries the main meaning and can contain textual, multi-media information as well as other boxes. The main spatial parameters of the content are width and height. Background of the content can be transparent, of some specific color or presented by image. The *padding* is an area which is laid out inside the box and wraps the content. It has a background of the box. The *border* serves as the visual framing of the content and possess the properties such as `border-width`, `border-style`, and `border-color`. The *margin* is a transparent area which makes a standoff with other adjacent boxes.

Figure 2.9 illustrates an example of the box model for the inline block of the visual formatting model. A block of this type can span several rows and respectively form so called *client rectangles* [264, 286] which are also CSS boxes. As we can see, top and bottom margins are absent for the inline box. The distance between lines for the inline box is controlled by the parameter `line-height` of the containing block box [245].

It is essential to note that the box has an additional component, an *outline* (see [260, Sec. 18.4] and [267, Sec. 7]) that is intended for establishing a visual focus on the element. The outline is mostly represented by the rectangle (however, it may be non-rectangular) with a border which in turn has the same configurable attributes as the border of the corresponding box. The outline in contrast to box's border is not divided on the top, right, bottom, or left components. It also separates from the corresponding box in terms of the visual formatting model. The outline is drawn on top of the box and does not influence the position or size of the box, or of any other boxes [260, App. E].

W3C extends box model in their candidate recommendations and working drafts dedicated to the "CSS Level 3." For instance, a working draft [254] introduces extensions such as *compact boxes* which gives a developer additional means for placing boxes on a web page's canvas. A candidate recommendation [266] and working draft [267, Sec. 6] extend concepts of a background and border with various possibilities of positioning background image, rounding the corners of

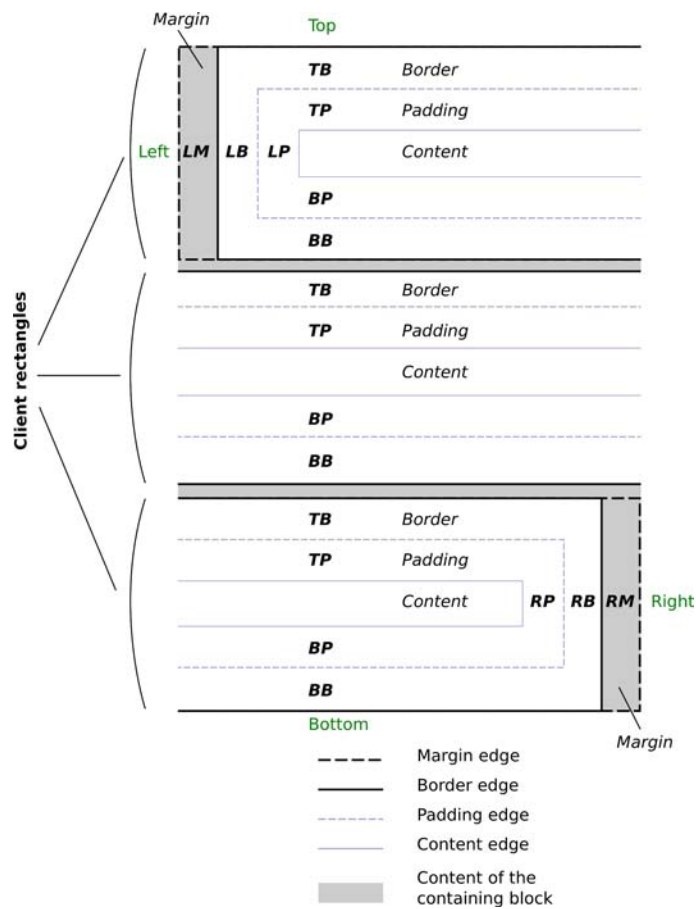


Figure 2.9: CSS box model for the inline block

a box's border, using images as a border, and so on. Moreover, a working draft [273] specifies various types of transformation (translation, rotation, and scaling) of boxes in two- and three-dimensional space.

Visual Formatting Model

Visual formatting of elements of the DOM tree is performed by the user agent (a web browser's engine in this case) which renders a document (a web page) according to the visual formatting model [260, Sec. 9,10], CSS rules of a document, and the position and size of a viewport.

To be precise, the following are concepts that underlie the visual formatting model:

- box model [260, Sec. 8] and types of boxes (block box, inline box, etc.),
- positioning scheme (normal flow, float, and absolute positioning),
- relationships between elements in the document tree (DOM tree),

- external information (e.g., viewport size, intrinsic dimensions of images, etc.).

We would like to focus on the types of boxes and positioning scheme, since these concepts are the most important in understanding the principle of rendering the layout for elements of the DOM tree; the box model is presented in Section 2.4.5.

There are two main groups of box's types (the type of a box is specified by the `display` property):

1. *block-level elements*,
2. *inline-level elements*.

Block-level elements are those elements of the DOM tree which are formatted visually as *blocks*—rectangular regions (e.g., paragraphs, headers). The following values of the `display` property make an element block-level: `block`, `list-item`, and `table`. *Inline-level elements* are those elements of the DOM tree which do not form new blocks of content; the content is distributed in lines (e.g., emphasized pieces of text within a paragraph, inline images; see Figure 2.9). The following values of the `display` property make an element inline-level: `inline`, `inline-table`, and `inline-block`. There are also terms such as *anonymous block boxes* and *anonymous inline boxes* that correspond to the visualized leaves of the DOM tree which have siblings and participate in the *block* or *inline formatting contexts* respectively. For other types of boxes, we refer interested readers to [260].

Elements in the DOM tree may generate more than one box, as in the case of tables (generate *table wrapper box* which contains *table box* and any *caption boxes*) [260, Sec. 17] and list items (generate *principal block box* and *marker box*) [260, Sec. 12].

There are three main positioning schemes which define a lay out of boxes:

1. *normal flow*,
2. *floats*,
3. *absolute positioning*.

Normal flow includes *block formatting* of block-level elements, *inline formatting* of inline-level elements, and *relative positioning* of block-level and inline-level elements. All boxes are placed one after another at the top of a containing block during the depth-first traversal over a DOM tree. In the block formatting contexts, boxes are laid out vertically, whereas, in the inline formatting, they are laid out horizontally (as the letters in words in text) one after another until there is no more room, then starting a new line below. Every line in the inline formatting context of the corresponding *containing block* is a *line box*. When an inline box exceeds the width of a line box, it is split into several boxes and these boxes are distributed across several line boxes (see Figure 2.9) [260, Sec. 9.4.2]. Once a box has been placed according to the normal flow or floated, it may be shifted relative to its position according to the relative positioning schema. The value `relative` of the attribute `position` enables such positioning. A *float* is a box that is shifted to the left or right on the current line of the corresponding normal flow context until its outer edge touches the containing block edge or the outer edge of another float. The most

important characteristic of a float is that the content of the containing block may flow along its side. Attribute `float` controls the floating. In the *absolute positioning model* (with `position` attribute value equal to `absolute` or `fixed`), a box does not have a relation to the normal flow and it is assigned a position with respect to a containing block. An absolutely positioned box establishes a new containing block.

It is also worth mentioning specifications, where other layout models and boxes' types are introduced, such as:

- “CSS Flexible Box Layout Module” [270],
- “CSS Multi-Column Layout Module” [262],

which are W3C's candidate recommendations, and

- “CSS Basic Box Model” [254],
- “CSS Regions Module Level 3” [271],
- “CSS Grid Positioning Module Level 3” [255],
- “CSS Template Layout Module” [263],
- “CSS Lists and Counters Module Level 3” [261],
- “CSS Exclusions and Shapes Module Level 3” [269],
- “CSS Box Alignment Module Level 3” [268],
- “CSS Basic User Interface Module Level 3,” where the additions to the box model are specified [267, Sec. 6],

which are working drafts that represent an on-going work of the W3C.

Working draft “CSS Transitions” specifies various means for animating a document's content [274].

2.4.6 Quantitative Visual Representation

Quantitative information in the description of spatial, visual characteristics of web pages is utilized in various works dedicated to the problems in the fields of WIE and WPU [123, 200, 214, 299]. The major quantitative information considered is coordinates of the CSS boxes, their width and height, and color encoded in RGB. Thus, the quantitative visual representation of the web page is defined by the set of elements with attributes and relations expressed quantitatively (e.g. distance in pixels). Further, we give a brief survey of works that leverage this representation. A relation of the works into this group of web page representation is conditional due to the fact that quantitative visual and spatial features are usually used together with tree representation (DOM or tag tree).

P.S. Hiremath et al. propose an automatic approach VSAP, in which quantitative information is analyzed for mining data regions [124], data records and data attributes [122, 123]. The authors experimentally show the effectiveness of their heuristic method in contrast to the purely DOM-based MDR [164] and NET [165], which additionally takes into account some spatial attributes (e.g. width).

G.M. Atiqur Rahaman et al. [200] present the method MDRMTA for structured object mining based on maximum text content comparison. The proposed method operates over tag tree and considers spatial extension of visualized nodes. The method introduced by the authors outperforms MDR [164].

H. Zhao et al. introduce ViNTs [306], a system for automatic wrapper generation for any given search engine. The tool analyzes regularities on a visual level considering the left indent of elements and the shapes formed by their left border as well as regularities in the HTML tag tree.

K. Simon et al. introduce ViPER [214], an automatic web data extraction tool specializing on data records. It analyzes regularities within the DOM tree together with quantitative visual features, such as position, height, width, and distance. The distinguishing features of this approach are an application of *general suffix trees* as an alternative to *edit-distance algorithms* and consideration of projection profiles of elements to enhance the data record separation process. ViPER, as well as ViNTs mentioned earlier, outperforms MDR [164]. ViPER also shows results better than ViNTs for data records with relatively complex structure.

Y.-F. Tseng et al. [235] aim at finding regular patterns for data objects extraction. The authors analyze DOM tree as well as spatial extension of web page elements. In particular, they introduce *entropy-based method* for regular pattern mining.

P. Lou et al. [169] from HP Laboratories introduce heuristics for article extraction (i.e. news stories, encyclopedia entries, or a single blog post).

In [219], Spengler et al. consider both tag tree and visual representations of a web page for the automatic content extraction with examples of news articles. The problem is formulated as a classification task, where every region of web page layout is associated with the respective label from a predefined set. The authors apply *loopy conditional random field* and represent a web page as a graph, which reflects a sequence of leaf nodes according to depth-first traversal over a tag tree, and relations between neighboring regions that have similar visual features, such as size, color, and style.

2.4.7 Qualitative Visual Representation

The term “qualitative” is widely used in the spatial cognition community, for example, with E. Clementini [50], A.G. Cohn [51], A.U. Frank [89], and J. Kong [144]. Qualitative characteristics possess a number of advantages over quantitative. In particular, they are used to provide information in the form understandable both for human and computer. This fact is a basis for conducting automatic spatial reasoning which is in some sense an analogy of the human reasoning. Interested readers are encouraged to refer to [3] for details of spatial reasoning and spatial logics.

We distinguish four main groups of web page representations which are used in WPU, Document Understanding (DU), and WIE, and which are further described:

- Models based on inclusion relationship,
- Models based on direction and alignment relationships,
- Models based on interval relations,
- Models rich with variety of spatial relationships.

It is worth mentioning that methods utilizing web page models based on the qualitative features usually do not operate over DOM tree.

Models Based on Inclusion Relationship

A model of a document based on the inclusion relationship can be represented as a tree, where a child node that corresponds to the element on a document canvas is topologically inside its parent node. Visualized elements of the document are usually modeled by the minimum bounding rectangles. For the web page, the nodes and mainly leaf nodes are associated with CSS boxes, and this representation can partially coincide with the structure of a DOM tree, especially for the *non-positioned inline-level elements* from the *normal flow* of the same *stacking context* (see Section 2.4.5) [260, Sec. 9]. This representation reflects a web page's logical structure incorporated within its layout. It is usually acquired by applying segmentation algorithms related to WPU and DU (see Section 2.3.2) and which has a so-called *segmentation tree* as the output. The most well-known algorithms for getting this representation of a web page are XY-cut and VIPS.

XY-Cut Algorithm was originally developed for scanned (raster) documents; it is a top-down segmentation technique which decomposes a page recursively into a set of rectangular blocks based on the analysis of the *projection profiles* [106, 182]. XY-Cut also has some modifications, for instance, one optimized for detecting reading order for scanned pages [176]. XY-Cut and its modifications are applicable both for raster documents [63, 106, 148, 176, 182] and web pages [61, 95, 150, 311]. For example, XY-cut algorithm is used in the approaches of identifying [150] and extracting [95] table data from web pages in the meta-search systems.

VIPS Algorithm was developed by D. Cai et al. [40,41]; it analyzes DOM tree and some visual features. The algorithm segments web page according to the size of the gaps between adjacent objects and additional heuristic rules. VIPS is leveraged for enhancing efficiency of information search [42], it is applied in the problems of data record [160, 166] and attribute [310] extraction and for automatic wrapper generation [167]. R.R. Mehta et al. [175] extend this algorithm with naive Bayes classifier for segmenting web page taking into account *topical similarity* of segments.

Algorithm similar to VIPS, but with simpler heuristics, was introduced earlier by X.-D. Gu et al. [102] with its application in the area of web adaptation. Later, P. Zhong et al. [308] use this method together with a proposed approach based on a generalized hidden Markov model to attach semantic labels to the tree structure of a web page and thus ameliorate efficiency of WIE.

H. Guo et al. [104] propose alternative techniques for web page segmentation which leverages geometric characteristics. A formed segmentation tree is used in CSurf [171], an extension of specialized browser HearSay [36, 222] for blind users (see Section 2.1.3).

E. Oro et al. [189] represent a web page in a tree-like structure PDOM with containment relation reflecting spatial allocation of web page elements. A PDOM also stores certain computed CSS attributes. The authors propose an approach SILA for data record extraction which leverages this structure. The authors show that their solution outperforms web information extraction systems such as MDR [164] (analyzing tag tree) and ViNTs [306] (analyzing tag tree in conjunction with some quantitative visual and spatial characteristics).

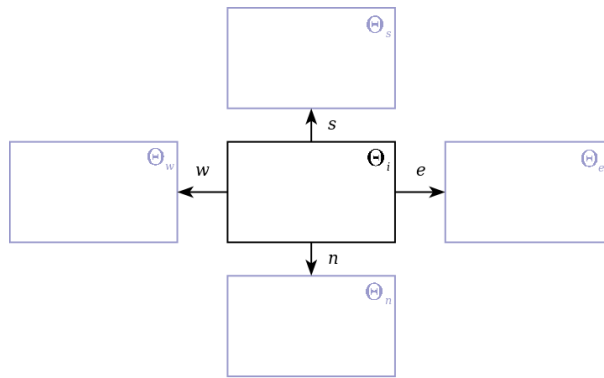


Figure 2.10: Illustration of example model based on direction relationships

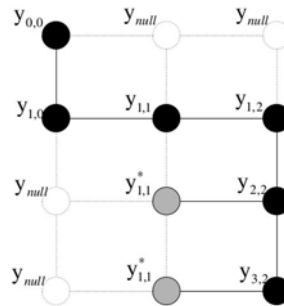


Figure 2.11: Object block representation [309]

Models Based on Direction and Alignment Relationships

Models Based on Direction Relationships can be represented on the whole as a set of visual objects with direction relationships defined between them, where every value of qualitative relationships is a value of corresponding linguistic variable. The most widely used relationships are “north” (or “top”), “south” (or “bottom”), west (or “left”), east (or “right”), presented in Figure 2.10. A similar model is described by T. Hassan in [114, 116] reflecting geometric structure of PDF documents in the form of *adjacency graph*, where four orthogonal direction relations are defined between adjacent objects. This structure is used for the data extraction from PDF documents. J. Zhu et al. [309] represent a region of a web page to be extracted (so-called object block) as a grid, where nodes correspond to basic elements of a web page within the region (see Figure 2.11). Several nodes can correspond to the same web page element for consistency. This can happen if an object has more than one element of the same direction (e.g. $y_{1,1}$ can have $y_{1,2}$, $y_{2,2}$, and $y_{3,2}$ to the right). This representation is used for WIE based on *two-dimensional conditional random fields*. X. Li et al. [161] model scanned documents in the form of directed labeled graph, where vertexes are rectangular areas (blocks) that correspond to text, graphics and other basic objects. Arcs define relations between adjacent blocks. There are four relations including “horizontal,” “vertical,” “diagonal-left,” “diagonal-right.” This graph is used by the

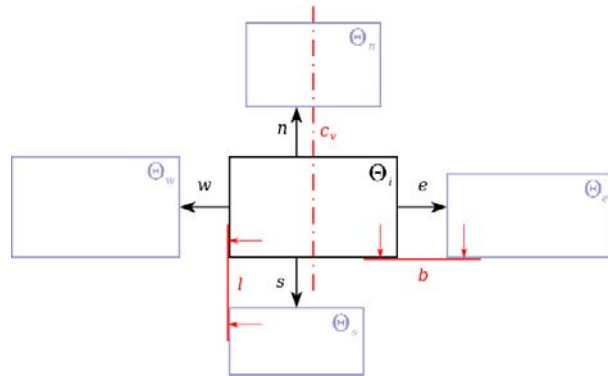


Figure 2.12: Illustration of example model based on direction and alignment relationships

authors in the task of document classification.

Models Based on Direction and Alignment Relationships can be symbolically illustrated as in Figure 2.12. There are two types of alignment in accordance with the Cartesian axes: horizontal (“left aligned,” “right aligned,” “centered vertically”) and vertical (“top aligned,” “bottom aligned,” “centered horizontally”). For example, M. Kovacevic et al. [146] represent a web page as an adjacency multigraph, where every node represents a basic HTML objects (text, images, form controls, etc.), and every edge represents relationships between the adjacent HTML objects such as “immediately before,” “immediately after,” “immediately left to,” and “immediately right to.” It is a weighted graph where, for every arc, distance between corresponding objects is assigned in pixels as well as a boolean value which indicates a presence of alignment. The authors use this representation for the analysis of various basic patterns (e.g. vertical and horizontal link lists, paragraphs) in the task of web page classification.

Models based on direction and alignment relations can be applied both for raster documents and web pages for solving various challenges including analysis [114] and classification [146, 161] of documents, information extraction [115, 116, 309], table identification [117, 149], table data extraction [93, 94], and web form understanding [305].

Models Based on Interval Relations

Temporal Interval Relations. In 1983, J. F. Allen introduced in his work [7] an interval-based temporal logic with a computationally effective reasoning algorithm based on constraint propagation. For the temporal intervals, the author presented 13 jointly exhaustive and pairwise disjoint (JEPD) relations (see Figure 2.13): “before,” “equal,” “meets,” “overlaps,” “during,” “starts,” “finishes” and six inversions of them.

Model of Walischewski. In his works [288, 289], H. Walischewski describes a geometric model of a document by means of the interval relations applied to the two-dimensional rectangular objects and *inclusion relations* acquired after document analysis. The model is realized as attributed directed graph. A set of vertexes V represents a set of basic layout objects such as page,

Relation	Symbol	Symbol for inverse	Pictorial example
X before Y	<	>	XXX YYY
X equal Y	=	=	XXX YYY
X meets Y	m	mi	XXXXYY
X overlaps Y	o	oi	XXX YYY
X during Y	d	di	XXX YYYYYYY
X starts Y	s	si	XXX YYYYY
X finishes Y	f	fi	XXX YYYYY

Figure 2.13: The thirteen Allen’s interval relations [7]

bounding block, line, word, and character. The vertex attributes are pairs $A_V = \langle l, c \rangle$, where l denotes a type of the basic layout object, c holds a logic label of the object. Edges are set between a child and its parent as well as between siblings in the tree reflecting the inclusion relation. Each edge attribute $A_E = \langle \vec{h}, \vec{v} \rangle$ with $\vec{h}, \vec{v} \in \{0, 1\}^{13}$, where each dimension corresponds to a certain interval relation. \vec{h} represents the interval relations between projections on axis x and \vec{v} is on axis y . Numbers 0 and 1 in the dimensions of the vectors denote absence or presence of the corresponding interval relation between the projections respectively. This representation of the scanned documents is used by the author for document understanding and was evaluated by examples of envelopes.

Model of Aiello. In [2], Aiello et al. propose a consideration of a document as a set of its possible layout (geometric) \mathcal{G} and logical \mathcal{L} structures. Each layout structure $g^i \in \mathcal{G}$ consists of a set \mathcal{O}_g^i of geometric objects and a set \mathcal{R}_g^i of geometric relations among them. The set of logical structures is defined in a similar way. Each logical structure $l^i \in \mathcal{L}$ consists of a set \mathcal{O}_l^i of logical objects, which are set in correspondence to geometric objects from \mathcal{O}_g^i , and a set \mathcal{R}_l^i of logical relations among them. The logical structure is proposed to be represented as a weighted graph. According to this approach, the authors model geometric objects (picture and text blocks) as a rectangle with attributes such as aspect ratio, font style, and number of lines. Between geometric objects, the authors set *thick boundary rectangle relations* which are JEPD (see Figure 2.14). These relations are based on interval relations and additionally take into account width of the boundary of the rectangular objects. This representation is used for document understanding and reading order detection [2, 4].

In [307], H. Zhao et al. introduce wrapper generation algorithm MSE for extracting query results from the search engines. In contrast to their previous approach with ViNTs [306] (where DOM tree and some quantitative visual attributes are taken into account), MSE enables identification of multiple sections (dynamically generated in response to a user query) and more effective differentiation of sections and records. The authors analyze DOM tree, position and attributes of elements expressed quantitatively, and some interval relations.

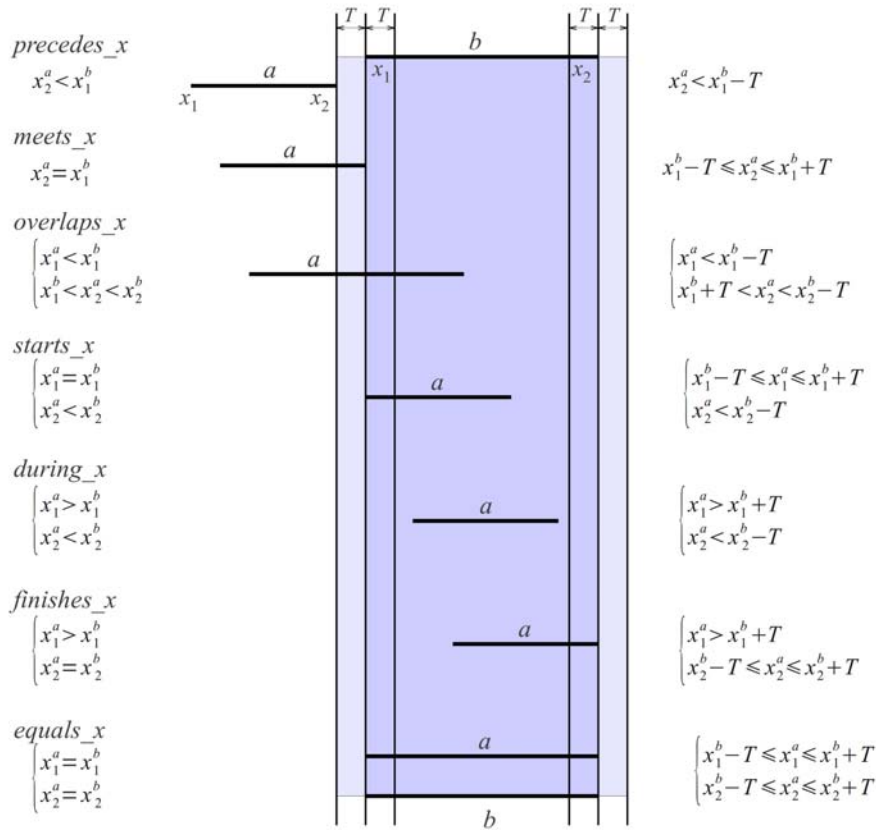


Figure 2.14: Thick boundary rectangle relations [2]

Thus, geometric models based on the interval relations are used for the analysis and understanding of graphical representation of scanned documents [2, 4, 141, 288, 289], layout template generation [215], web page classification [54], and web data extraction [307]. There are several works dedicated to the interval algebra [7, 154, 162, 184], its two-dimensional extension [15], and n-dimensional [17]. Interval algebra is also used for the interpretation of space relationships and automatic logical reasoning [3, 181].

Models Rich With Variety of Spatial Relationships

There are other models which incorporate various types of relationships adopted from the area of spatial logics and spatial cognition. The most prominent are representations used in a *spatial graph grammar* and *spatial document object model*.

Graph representation for the spatial graph grammar. J. Kong et al. introduce *spatial graph grammar* (SGG) [144] with its application in adaptive web design (see Section 2.1.3). The SGG incorporates spatial notions into the abstract syntax and takes both the connectivity and

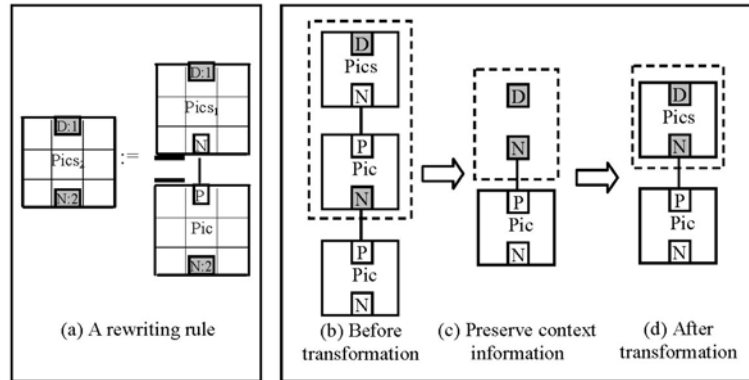


Figure 2.15: Rewriting rules in SGG and their application [144]

spatial relationships among objects as the precondition of a graph transformation. It allows one to specify visual languages as well as GUI. The authors leverage four groups of spatial relationships: topological *Top* (“touch,” “in,” “cross,” “overlap,” and “disjoint”), direction *Dir* (eight cone-based directions such as “north,” “north-east,” “east,” etc.), distance *Dis* (e.g. “close” and “far”), and alignment *Align*.

The SGG is defined for a marked graph $G = \langle N^G, E^G, s^G, t^G, g^G, mark^G \rangle$, where N^G is the set of nodes, E^G is the set of edges, $N^G.V^G$ is the set of vertexes constructing N^G , $s^G : E^G \rightarrow N^G.V^G$ and $t^G : E^G \rightarrow N^G.V^G$ are two functions that specify the source and target points of an edge, g^G is the spatial signature defined on the node set N^G , $mark^G$ is a marking function for vertexes used in the rewriting rules for preserving context information. A spatial signature is a function $g^G : N^G \times N^G \rightarrow Top \times Dir \times Dis \times Align$. Figure 2.15 illustrates a rewriting rule and example of its application. Pic, Pic_{s1}, Pic_{s2} represent nodes; D:1, N:2, N, P are vertexes; numbers 1 and 2 in the vertexes are labels according to the marking function.

Spatial Document Object Model (SDOM). In [190], E. Oro et al. introduce SXPath language which extends XPath 1.0 [243] and allows for navigating DOM structures as well as for exploiting the spatial layout of DOM nodes of a rendered web page. The authors propose this language, which maintains polynomial time combined complexity, for use in the problems of WDE. SXPath is evaluated over a *spatial document object model* (SDOM) which extends DOM with additional spatial relations. In particular, SDOM is defined as a *node labeled sibling ordered tree* enriched with relations of *rectangular algebra* (*interval relations* applied for two-dimensional space). It is described formally as $SDOM = \langle V, R_{\downarrow}, R_{\Rightarrow}, A, f_s \rangle$, where V is the set of labeled DOM nodes; R_{\downarrow} is the *firstchild* relation; R_{\Rightarrow} is the *nextsibling* relation; $A \subseteq V_v \times V_v$ is the set of arcs that represent spatial relations between pairs of nodes in $V_v \subseteq V$, visualized on screen; $f_s : A \rightarrow R_{rec}$ is the function that assigns to each element in A a relation of rectangular algebra in R_{rec} . There are also three order relations between nodes: *document total order* according to the XPath specification [243], *directional total orders* which reflect four direction relations, and *containment partial order*. By means of SXPath, the authors provide spatial axes which

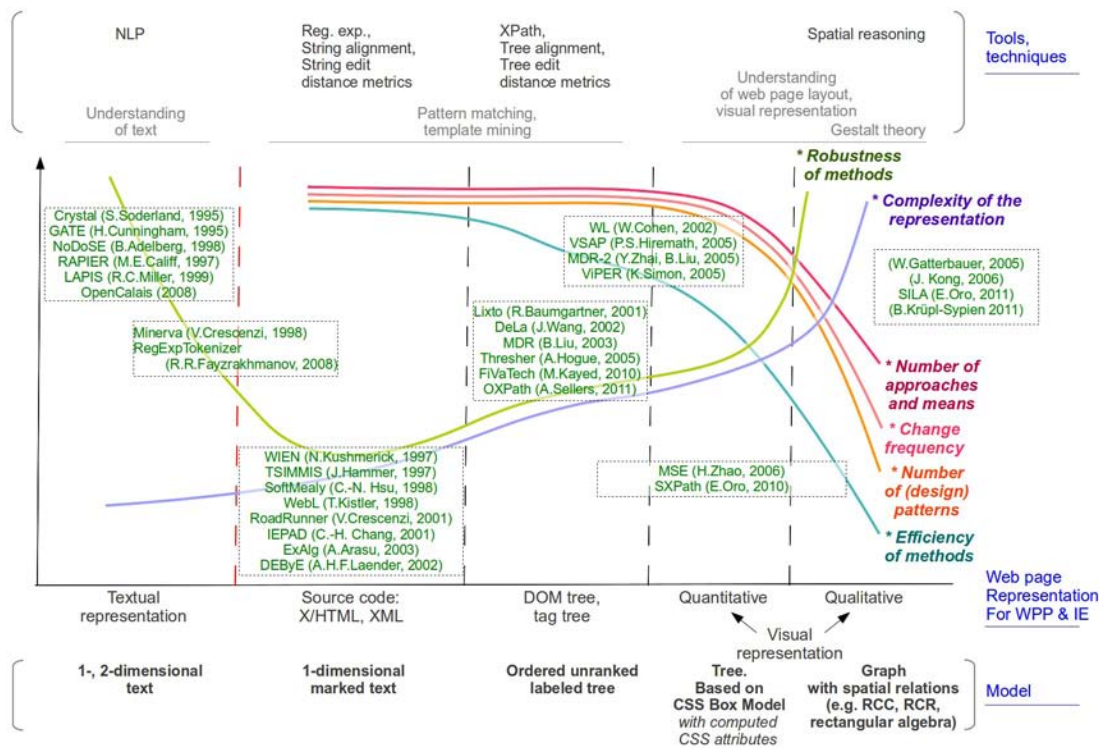


Figure 2.16: Survey of contemporary tools and techniques applied for different web page representations

reflect directions (based on Rectangular Cardinal Relations (RCR)) and topology of the elements, expressed by means of rectangular algebra. Relations from Region Connection Calculus (RCC) such as “contained,” “container,” and “equivalent” are used in the capacity of topological relations.

In contrast to XPath, SXPath enables creation of queries independent from DOM tree, based on spatial relations (direction and topological). Thus, SXPath gives a possibility of creating wrappers more robust and more independent from changes on the source code level. Moreover, query expressions based on the spatial relations are more intuitive for the user than expressions based on the relations defined in the DOM.

2.4.8 Analysis of the Considered Models

We distinguish four main web page representations: textual, source code, tree, and visual. The latter we divide into quantitative and qualitative. Figure 2.16 schematically gives a general overview of various methods and techniques applied for different web page representations.

A **textual representation** (see Section 2.4.3) is considered in classical tasks of IE, such as event extraction and named entity recognition [179], and thus has a direct interest of the NLP community. Information extraction from one-dimensional text usually includes grammatical and syntactic analysis. However, due to the fact that a text file can also contain data structures, such

as lists and tables, there are also techniques which process a two-dimensional representation of a text. The textual representation is out of the scope of this thesis which is dedicated to the analysis of structural characteristics of a web page, established by the source code, DOM tree and visual formatting.

A **source code** (see Section 2.4.2) of a web page written in X/HTML or XML is a marked text which reflects the structural characteristics of the content. Most of the relevant WIE techniques are based on regular expressions and string alignment with string edit distance metrics utilized.

A **tree structure** (see Section 2.4.4) is generally presented by DOM tree or tag tree (usually modeled as an ordered unranked labeled tree) based on the source code. A tree structure is isomorphic to the source code if the latter is valid and well-defined, and scripts which can change the tree are not applied. DOM tree is an internal representation of a web page within the web browser. Therefore, in terms of contemporary interactive web applications that also leverages scripts (e.g. JavaScript) which can change the DOM tree, an analysis of the tree structure is more relevant than an analysis of the source code. Most of the well-known techniques which operate on this representation are distinguished by using XML technologies (e.g. XPath), tree alignment, and tree edit distance.

A **quantitative visual representation** (see Section 2.4.6) is usually leveraged together with tree representation. Therefore, it is usually represented as a tree enriched with data acquired from the CSSOM computed by the web browser engine.

A **qualitative visual representation** (see Section 2.4.7) is usually modeled as a graph which reflects a set of elements of different types with different relations defined on this set. The relations specified for the rendered web page are mainly spatial relationships between web page elements, such as topology (e.g. based on RCC), distance and direction (e.g. based on RCR) expressed qualitatively [3]. The analysis of the rendered web page mainly boils down to the analysis of its spatial characteristics (its layout), and has a direct relation to spatial reasoning.

As it is symbolically depicted in Figure 2.16, *complexity* of the web page representation increases from the textual representation to the qualitative visual, however *robustness* of the methods greatly rises for the textual and the quantitative visual representations. We confirm this tendency by the fact that textual and visual representations are the natural forms of information representation for the human being. Therefore, methods operating on these views often reflect and simulate some processes which the human being uses, for example, part of speech recognition, sentence splitting, entity recognition for the text, and analysis of relative spatial allocation of elements including their size, color, and typographical characteristics for the rendered web page. Also important is the *Gestalt theory* [143, 291] which investigates the peculiarities of human perception. Gestalt theory is mentioned in [151] as an important means for the analysis of the web page visual representation. Some aspects of Gestalt theory are also used in the analysis of the quantitative spatial characteristics presented in [299]. From the source code to the visual representation, the graph of robustness is also related to the *change frequency* and *number of corresponding design patterns*. The source code and DOM tree are prone to greatly more frequent changes compared to the visual representation. Furthermore, the number of various visual design patterns and corresponding ways of spatially arranging information objects is greatly less than the set of various ways of coding them.

The superiority of methods based on the DOM tree over methods based on the source code is mainly confirmed in [135] (where the authors claim that FiVaTech is more effective than ExAlg [12]) and [164] (where the authors demonstrate that MDR outperforms IEPAD [46]). Furthermore, in [23, 98] the authors demonstrate an effectiveness of leveraging the DOM tree in contrast to the source code by example of Lixto Visual Wrapper. Effectiveness of the methods which additionally analyze quantitative spatial characteristics in contrast to the methods merely based on the treelike structure is proved in [123, 200, 214, 306], where the authors compare their approaches (VSAP, MDRMTA, ViPER, and ViNTs respectively) to MDR [164]. The superiority of the methods considering qualitative spatial characteristics against methods considering quantitative spatial characteristics is demonstrated in [189], where the authors compare their approach SILA to ViNTs [306]. SILA also outperforms tree-based MDR.

In [304], the authors introduce MDR-2, a more robust approach relative to the MDR, considering additionally certain quantitative spatial characteristics. In [307], the authors introduce more advanced method MSE than earlier developed ViNTs [306], in addition to leveraging some interval relations.

In general, all of the mentioned web page representations are still actively used for WPP (see Section 2.3). A representation can be chosen based on the task and the type of information to be extracted or analyzed. For example, if a web page mainly contains textual information (e.g. a monologic text) and the task of IE is posed, the textual representation is the most suitable. The source code and DOM tree (or tag tree) are best for the WDE from the web pages which change relatively seldom, have relatively simple structures (e.g. the source code reflects the structure of web objects), or have regularities in their structures which can be mined and reflect features of the object to be extracted (e.g. web page of the Deep Web generated with the specific template from the back-end database). The source code and DOM tree are well-studied, have corresponding standards, and plenty of various approaches and tools. Rendered (visualized) web pages are considered if the analysis of spatial configuration or visual characteristics is required, for example, table recognition, product list extraction, web page segmentation. Unfortunately, there is no common model of the visual representation of a web page suitable for WPP and relatively few methods are developed. Targeting the visual representation, the developers always face the problem of acquiring visual characteristics and developing an appropriate structure or model which can reflect the necessary information used for the specific problem in a convenient form.

Thus, in regards to the conducted analysis, we single out two main challenges which we aim to solve in this thesis:

- In contrast to the source code and DOM tree, there is no unified model or standard for visual representation of the web page suitable for WPP. In developing new methods, the researcher always encounters the problem of designing a new model or structure which can hold all the necessary features and relationships. Therefore, we pose a goal of developing a unified model which reflects basis visual, spatial, and functional characteristics of a web page. Moreover, due to the fact that quantitative visual features are used in conjunction with the DOM tree, we believe that it is important to additionally integrate the DOM tree model. If there is a need to apply methods which operate on the source code level, DOM tree can be trivially serialized into this representation.

Chapters 3 and 4 are dedicated to this challenge.

- There is a relatively small amount of methods leveraging visual cues. Various visual features and relationships are used in different methods, and therefore the issue of discovering the appropriate visual cues which will favour the development of efficient approaches of WPP is of paramount concern. Thus, we pose the challenge of developing a framework for investigating and developing effective and efficient methods operating over the visual representation (see Chapter 5). Moreover, due to the presence of declarative and imperative approaches in the area of WPP, we consider the possibility of applying them within the framework presented in this thesis.

2.5 Semantic Web Approach

The Semantic Web [27] is a proposal to build an infrastructure of machine-readable semantics for the data on the Web [229, p. 159]. The development of the Semantic Web is based on the multi-disciplinary approach, applying knowledge from different research fields and technologies, including artificial intelligence, database and content management systems, as well as knowledge-based systems, machine learning, natural language processing, distributed computing, service-oriented architecture, agents, and grid computing. The Semantic Web can be viewed from two main perspectives reflecting its *information* and *computational aspects* [134, Sec. 1.2]. The information aspect, which includes the problems of information representations, relates to the Semantic Web Content molded primarily by data and metadata. Data can be presented in structured (e.g., relational), semi-structured (e.g. RDF, XML), and unstructured forms (e.g. plain text) with embedded metadata descriptions. Metadata descriptions are used to annotate data on the Web, whereas the ontologies and schemata provide the underlying vocabulary and semantics for the metadata annotations. Ontologies play an important role in the Semantic Web, providing well-defined knowledge and means for reasoning to materialize implicit knowledge as well as create domain- and application-specific views on the underlying content. The *computational aspect* of the Semantic Web relates to the complexity of computing infrastructures and communication between computational entities.

The overall architecture of the Semantic Web proposed by Tim Berners-Lee is depicted in Figure 2.17. RDF(S) layer refers to RDF and RDFS languages, where RDF enables making statements about web resources, and RDFS provides modeling primitives for organizing web resources into hierarchies. RDFS can be viewed as a primitive language for building ontologies. Ontology layer provides rich ontological language for representing more complex relationships between web resources. Logic layer is intended to enhance the ontology language and allow the writing of application-specific declarative knowledge. The *Proof layer* involves the actual deductive process as well as the representation of proofs in web languages and proof validation [9, Sec. 1.4]. The *Trust layer* refers to the digital signature and recommendations by trusted agents and rating agencies. This layer is also related to the web of trust concept.

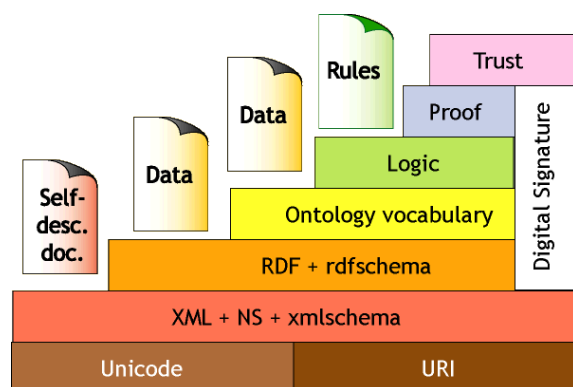


Figure 2.17: The layer model of the Semantic Web architecture (<http://www.w3.org/2001/09/06-ecdl/slide17-0.html>)

2.6 Ontology and Logical Inference

Historically, the term “ontology” (from the Greek word *Οντολογία*) originated in philosophy and is traditionally listed as a part of metaphysics—the major branch of philosophy. An ontology is a particular theory about the nature of being or the kinds of existence. It deals with questions concerning what entities exist or can be said to exist, and how such entities can be related to each other, either grouped or subdivided according to their similarities and differences. In the application to Information Science, *ontology* is a formalism whose purpose is to support humans or machines in sharing some common knowledge in a structured way [229, p. 255]. This knowledge is represented as a set of concepts within a domain and relationships between pairs of concepts providing a support for reasoning about entities within the domain modeled. The most famous definition is the following:

An ontology is a formal, explicit specification of a shared conceptualization.
(T.R. Gruber [101], R. Studer et al. [224])

We illustrate this definition in Figure 2.18. A *conceptualization* is an abstract representation of some aspect of the world which is of interest to the users of the ontology. The conceptualization defines the main concepts, entities and relationships between them reflecting the spatial, temporal, and logical interrelations on a domain space. The term “explicit” in the definition refers to the fact that constructs used in the specification must be explicitly defined and the users of the ontology must agree on them. *Formal* means that the specification is encoded in a precisely defined language whose properties are well-known and understood. Examples can be the languages used in the fields of Knowledge Representation and Artificial Intelligence. *Shared* means that the ontology is meant to be available for the group of people and various applications. [229, p. 256]

A detailed definition of a formal ontology is given by N. Guarino:

An ontology is a logical theory accounting for the intended meaning of a formal vocabulary, i.e. its ontological commitment to a particular conceptualization of the world. The intended models of a logical language using such a vocabulary

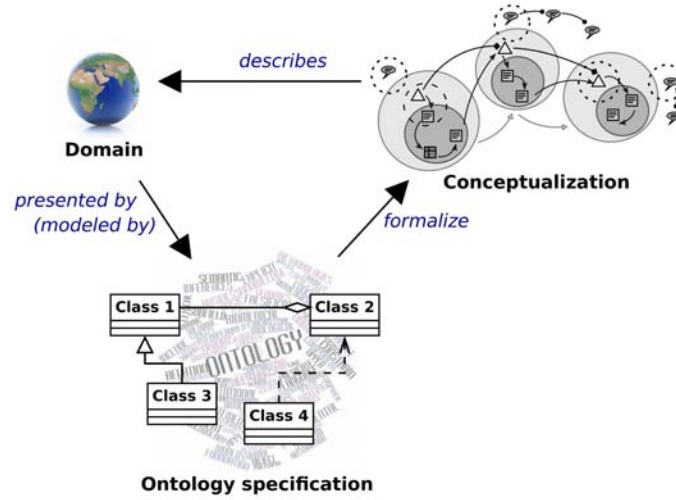


Figure 2.18: Gruber's and Studer's ontology definition

are constrained by its ontological commitment. An ontology indirectly reflects this commitment (and the underlying conceptualization) by approximating these intended models. (N. Guarino [103])

It is important to note that an ontology is *language-dependent*, while a conceptualization is *language-independent*. Thus, a vocabulary (represented by the set of concepts, relationships, and their features) and formal language are important attributes of the ontology.

We give the following formal definition of the ontology \mathcal{O} , that underlies the UOM (see Section 4.2) and the Multi-Axial Navigation Model (MANM) (see Section 6.2):

Definition 2.1 (Ontology). *Ontology \mathcal{O} is a tuple $\langle C, H_C, R_C, H_R, I, \iota, R_I, A \rangle$ that consists of the following elements: The set of ontological concepts C is arranged in a subsumption lattice H_C . The set of relationships R_C defines various relations on the set of concepts $R_{C-C} \subseteq R_C$ and between the set of concepts and a set of various data types $R_{C-D} \subseteq R_C$ ($R_{C-C} \cap R_{C-D} = \emptyset$). H_R is a disjoint subsumption hierarchies of the relations in R_{C-C} and R_{C-D} . Elements C, H_C, R_C, H_R correspond to the schema of the ontology \mathcal{O} , whereas I, R_I are the instantiations of the corresponding concepts and relationships. I is a set of instances of corresponding concepts in C , where $\iota : I_i \rightarrow 2^C$ sets such a correspondence. $R_{I-I} \subseteq R_I$ is a set of triples of type $\langle I_j, R_k, I_l \rangle$, $R_{I-D} \subseteq R_I$ is a set of triples of type $\langle I_p, R_q, D_r \rangle$ where I_j, I_l , and I_p are instances of concepts in C , $R_k \in R_{C-C}$, and $R_q \in R_{C-D}$. A set of axioms defined in A specifies additional constraints and dependencies, that can be used to infer implicit knowledge.*

An ontology, as a specification of a certain conceptualization of a domain of interest, provides the mechanisms for modeling the domain leveraging the ontology modeling languages (see Section 2.6.1) and reasoning [170] upon it (see Section 2.6.2). Furthermore, there are query languages available for retrieving certain individuals, concepts, and relations that satisfy specific conditions, e.g. HiLeX [173], nRQL [107], OWL-QL [88], RQL, and SPARQL (see Section 2.6.4).

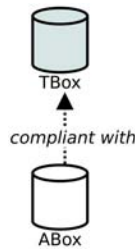


Figure 2.19: Knowledge base from the perspective of Description Logics (DL)

Focusing on web-oriented problems, such as web accessibility, web page understanding, and web information extraction, we consider ontologies from the perspective of the Web and the Semantic Web in particular, while playing attention to languages such as RDF(S), OWL, and SPARQL.

2.6.1 Ontology Languages

Ontologies offer the provision of rich semantics that is accessible both by human and machine and enables them to automatize the process of reasoning. In this section, we consider ontology languages from the perspective of their application for modeling the UOM, which describes the web page representation, and the MANM, enhancing navigation characteristics of web pages, as well as their integration into the Semantic Web.

Description Logics

Description Logics (DL) [14] is one of the most widespread formal knowledge representation language used for specifying ontologies and knowledge bases by means of *terminological (TBox)* and *assertional (ABox) components*. A basis of a DL is the *description language* which allows its users to build complex concepts and roles out of atomic ones. These descriptions are used in the TBox and introduce the terminology (vocabulary) of an application domain, imposing additional non definitional constraints on the concepts' and roles' interpretation. According to the Definition 2.1, the TBox describes the schema of the ontology \mathcal{O} with concepts from \mathcal{C} and roles from \mathbf{R}_C , whereas \mathbf{H}_C , \mathbf{H}_R represent subsumption lattices (taxonomy) based on concept and role inclusions; and \mathbf{A} defines supplementary constraints which can also contribute to the subsumption lattices. In the assertional component, ABox, facts about a specific application situation are stated by introducing named individuals (from \mathbf{I}) and relating them to concepts (by ι) and roles (from \mathbf{R}_I). Thus, in terms of DL, an ontology as a knowledge base is a pair $\langle \mathcal{T}, \mathcal{A} \rangle$ with ABox \mathcal{A} compliant with TBox \mathcal{T} (see Figure 2.19). *Reasoning* then over ABox and TBox allows us to derive implicit knowledge from the explicitly represented one [229, p. 5]. For the ontology \mathcal{O} , the reasoning mostly concerns axioms and statements introduced in \mathbf{A} as well as in \mathbf{H}_C and \mathbf{R}_C . DL has two important features: it does not make the *unique name assumption (UNA)* and the *closed world assumption (CWA)*. Absence of UNA means that two concepts or roles with different names may be allowed by some inference to be equivalent. Absence of CWA, and thus presence of the *open world assumption (OWA)*, means that lack of knowledge of a fact does not immediately imply knowledge of the negation of that fact.

Table 2.1: Syntax and semantics of \mathcal{ALC}

Constructor	Syntax	Semantics
top	\top	$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
bottom	\perp	$\perp^{\mathcal{I}} = \emptyset$
atomic concept	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
atomic role	R	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
<i>For C, D concepts and S a role name</i>		
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
universal (value) restriction	$\forall S.C$	$\{x \in \Delta^{\mathcal{I}} \mid \forall y [\langle x, y \rangle \in S^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}]\}$
existential restriction	$\exists S.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y [\langle x, y \rangle \in S^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}]\}$

Table 2.2: TBox and ABox additional constructors

Constructor	Syntax	Semantics
<i>Terminological axioms</i>		
equality	$C \equiv D$ ($S \equiv R$)	$C^{\mathcal{I}} = D^{\mathcal{I}}$ ($S^{\mathcal{I}} = R^{\mathcal{I}}$)
inclusion	$C \sqsubseteq D$ ($S \sqsubseteq R$)	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ ($S^{\mathcal{I}} \subseteq R^{\mathcal{I}}$)
<i>Assertional axioms</i>		
concept assertion	$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
role assertion	$R(a, b)$	$\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$

There are several dialects of description language, where \mathcal{ALC} is one of the basics. In Table 2.1, we introduce syntax and semantics of the main constructors of \mathcal{ALC} . The semantics of concept descriptions is given using the notion of interpretation \mathcal{I} , which assigns sets to concepts and binary relations to roles. An interpretation is a pair $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, where the domain $\Delta^{\mathcal{I}}$ is a non-empty set and $\cdot^{\mathcal{I}}$ is a function that assigns to every constructor (concepts and roles) elements from the domain $\Delta^{\mathcal{I}}$. There are many other constructors which are introduced in different dialects of languages in DL, e.g. (ref R) for reflexive role, (irr R) for irreflexivity, (sym R) for symmetry, (asy R) for asymmetry, (trans R) for transitivity, and (func R) for functional role. Supplementary constructs of TBox and ABox are introduced in Table 2.2. It is worth mentioning that for most DLs, the basic inference problems (see Section 2.6.3) are decidable, with complexities between \mathbb{P} and ExpTime .

In Example 2.1, we demonstrate the use of DL on some abstract example.

Example 2.1. *Statements such as “Alice is a blind user; a blind user is the user who is sightless. All blind users use assistive technology.” Can be expressed in DL in the following way:*

$\text{BlindUser} \equiv \text{User} \sqcap \text{Sightless},$
 $\text{BlindUser} \sqsubseteq \exists \text{use.AssistiveTech},$

BlindUser(alice).

Applying the reasoner, one can derive the subsumption lattice H_R of the ontology \mathcal{O} and infer that a concept BlindUser is subsumed by the concepts User and Sightless, as well as that Alice is a user and she is sightless. Moreover, having the assertions User(cheshireCat) and Sightless(cheshireCat) in ABox, a reasoner will infer that cheshireCat is also a blind user: BlindUser(cheshireCat).

RDF and RDFS

If HTML and the Web made all the online documents look like one huge book, RDF, schema, and inference languages will make all the data in the world look like one huge database.

— Tim Berners-Lee, *Weaving the Web*, 1999

The Resource Description Framework (RDF) [250] is a data model for representing information about resources in the Web. The first recommendation that specified metadata model and language (based on XML) of RDF was introduced in 1999 by the W3C [242]. RDF conveys a common framework for expressing information that can be shared among different applications without loss of meaning. This representation is required for the application of the automatic processing than for the users. RDF corresponds to one of the layers in the multi-layered model of the Semantic Web stack; it underlies the *ontological* (Web Ontology Language (OWL) and Rules) and *logical* (Logic framework and Proof) layers. RDF follows the W3C design principles of interoperability, extensibility, evolution and decentralization. Particularly, it was designed to have a simple model, with a formal semantics and provable inference, as well as an extensible vocabulary based on Uniform Resource Identifiers (URIs)—a compact string of characters for identifying an abstract or physical resource [26]. RDF allows anyone to make statements about any *resource*. In the RDF model, the *universe* to be modeled is a set of such resources that are essentially anything that can have a URI [229, p. 159]. However, RDF also supports anonymous resources which do not have a URI, known as *blank nodes*. A blank node can be used to represent a resource which either has no URI or for which no URI is known.

Information in the RDF model defines relations on the set of resources and data by means of statements which are triples of the form *subject-predicate-object*. *Predicate* is a binary relation between *subject* and *object* which are resources, however, the object can also be a *literal* representing atomic values. Literals represent data that is serialized into the string for which the data type can be explicitly specified, for instance, by means of XML Schema datatypes [285]. The RDF model can be represented as a labeled, directed multi-graph (i.e. graph which has pairs of vertexes interconnected by several arks), where nodes represent subject and object and whereas arcs are predicates.

The RDF specification includes a set of reserved words—RDFS [251] vocabulary—which are used to specify a type system (vocabulary) for the domain of discourse. Thus, RDF Schema (RDFS) introduces some simple ontological concepts, such as *class* (C) and concepts of subclass and subproperty, enabling the description of hierarchies of classes (H_C) and properties (H_R).

An RDF *property* is a predicate in RDF triples, while a class represents a collection of resources (I, ι). With RDFS one can also define the range and domain of the property.

There are several possible serializations of RDF, such as Notation 3, Turtle, and N-Triples, however the RDF/XML is most widespread due to its integration with the Semantic Web technology. Example 2.2 shows the main syntactic elements of RDF/XML, and also demonstrates some limitations of RDF(S) as a metadata model.

Example 2.2. *Statements presented in Example 2.1 can be represented in pure RDF(S), serialized into RDF/XML, as follows:*

```
1 <rdf:Property rdf:ID="use"/>
2 <rdfs:Class rdf:ID="AssistiveTech"/>
3 <rdfs:Class rdf:ID="BlindUser">
4   <rdfs:subClassOf rdf:resource="#Sightless"/>
5   <rdfs:subClassOf rdf:resource="#User"/>
6 </rdfs:Class>
7 <rdfs:Class rdf:ID="Sightless"/>
8 <rdfs:Class rdf:ID="User"/>
9 <rdf:Description rdf:ID="alice">
10  <rdf:type rdf:resource="#BlindUser"/>
11 </rdf:Description>
```

As can we see, RDF(S) is not expressive enough to specify that the BlindUser class is equal (not a subclass) to the intersection of classes Sightless and User. It is also impossible to state that all blind users use assistive technologies.

The expressive power of RDF and RDFS is deliberately very limited. RDF is only capable of representing binary predicates (R_I), while expressiveness of RDFS is limited to a subclass hierarchy (H_C) and a property hierarchy (H_R) [174].

Web Ontology Languages

OWL is an important step for making data on the Web more machine processable and reusable across applications.

— Tim Berners-Lee, 08/19/2003 W3C press release

The Web Ontology Language (OWL) was developed by W3C to satisfy the requirements for a language which supports the Semantic Web. OWL corresponds to the ontology layer in the Semantic Web's architecture and utilizes the features (semantics and syntax) of RDF(S). In contrast to RDF(S), OWL is an ontology language which provides a richer set of constructs for building complex ontologies. OWL introduces the term "OWL class" which is a subclass of "RDFS class." Furthermore, there is a distinction between object properties defined in the set of resources and data properties that set a correspondence between resources and literals. Taking into account an incompleteness of information on the Web, OWL makes open world assumption (OWA) and does not make unique name assumption (UNA) the same as DL. There are two main editions of the family of OWL: OWL 1 [248] and OWL 2 [279]. OWL 2 was developed later and extends the expressiveness of OWL 1. It includes additional syntactic sugar, introduces extended data type capabilities and constructs for properties, such as self-restriction ("local

reflexivity”), reflexive, irreflexive, and asymmetric object properties, disjoint properties, property chain inclusion, property qualified cardinality restrictions, etc. The detailed list of innovations can be found in [99, 278].

OWL provides three sublanguages: OWL Full, OWL DL, and OWL Lite. OWL Full is upward-compatible with RDF, both syntactically and semantically. It uses all the OWL language primitives and also allows the combination of these primitives in arbitrary ways with RDF and RDFS. This includes the possibility of changing the meaning of the predefined primitives. For example, in OWL Full, one could impose a cardinality constraint on the class of all classes, essentially limiting the number of classes that can be described in any ontology [220, Sec. 4.1]. Furthermore, a class can be treated simultaneously as a collection of individuals and as an individual in its own right [249]. It is important to note that the reasoning (see Section 2.6.3) over OWL Full ontologies is undecidable.

OWL DL is a sublanguage of OWL Full and conceptually based on DL. Every legal OWL DL document is a legal RDF document but not vice versa. OWL 1 DL corresponds to $SHOIN(\mathcal{D})$ DL, while OWL 2 DL corresponds to $SRIOQ(\mathcal{D})$ DL. In contrast to OWL Full, OWL DL permits efficient reasoning support: OWL DL provides the maximum expressiveness possible while retaining computational completeness, decidability, and the availability of practical reasoning algorithms. It has various reasoners available, such as Jena, Pellet, FaCT++, and HermiT. The reasoning (see Section 2.6.3) over OWL DL is $N2ExpTime$ hard.

OWL Lite further restricts OWL Full, being a subset of OWL DL. For example, it excludes enumerated classes, disjointness statements, and arbitrary cardinality. OWL 1 Lite corresponds to the $SHIF(\mathcal{D})$ DL. The reasoning over OWL Lite is in $PTime$.

In addition, OWL 2 has three main language profiles such as OWL 2 EL, OWL 2 QL, and OWL 2 RL which provide various expressiveness of OWL 2 and are designed to be implementable in $PTime$. In order to guarantee scalable reasoning, these profiles share some limitations regarding their expressiveness. In general, they disallow negation and disjunction, as these constructs complicate reasoning and have proved to be only rarely needed for modeling [279]. OWL 2 EL is based on DL EL++ and is designed for ontologies that contain a large number of classes or properties; OWL 2 QL is aimed primarily at query answering; OWL 2 RL is aimed at applications that require quite a bit of the expressivity provided by OWL 2 DL but also require scalable reasoning.

The OWL family of languages supports a variety of syntaxes (see Figure 2.20). It includes *abstract syntaxes*, such as Functional Syntax, and *exchange syntaxes*, such as RDF/XML, OWL/XML, Manchester Syntax, and Turtle [277], where RDF/XML is the primary exchange syntax that provides interoperability among OWL 2 tools. In Example 2.3, we demonstrate the serialization of an OWL ontology by means of RDF/XML syntax.

Example 2.3. *The expressiveness of OWL DL is enough to represent the statements introduced in Example 2.1. Translated into RDF/XML syntax, the corresponding ontology looks as follows:*

```

1 <owl:ObjectProperty rdf:ID="use"/>
2 <owl:Class rdf:ID="AssistiveTech"/>
3 <owl:Class rdf:ID="BlindUser">
4   <owl:equivalentClass>
5     <owl:Class>
6       <owl:intersectionOf rdf:parseType="Collection">

```

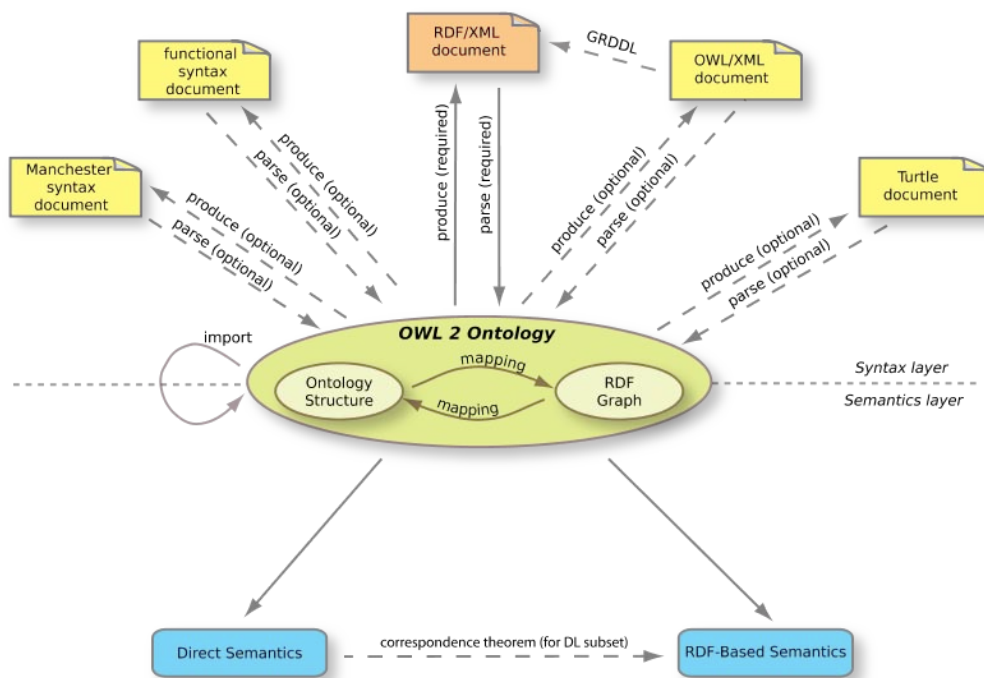


Figure 2.20: Syntax and Semantic layers of OWL 2 [277]

```

7   <rdf:Description rdf:about="#Sightless"/>
8   <rdf:Description rdf:about="#User"/>
9   </owl:intersectionOf>
10  </owl:Class>
11 </owl:equivalentClass>
12 <rdfs:subClassOf>
13   <owl:Restriction>
14     <owl:onProperty rdf:resource="#use"/>
15     <owl:someValuesFrom rdf:resource="#AssistiveTech"/>
16   </owl:Restriction>
17 </rdfs:subClassOf>
18 </owl:Class>
19 <owl:Class rdf:ID="Sightless"/>
20 <owl:Class rdf:ID="User"/>
21 <owl:NamedIndividual rdf:ID="alice">
22   <rdf:type rdf:resource="#BlindUser"/>
23 </owl:NamedIndividual>

```

2.6.2 Inference Rules

The languages of RDF and OWL can be viewed as specializations of predicate logic. In particular, OWL Lite and OWL DL correspond roughly to a DL, a subset of predicate logic for which efficient proof systems exist. Another subset of predicate logic with efficient proof systems comprises the so-called *rule systems* (also known as *Horn logic* or *definite logic programs*) [9, Sec. 5.1].

A rule has the form $A_1, A_2, \dots, A_n \rightarrow B$, where A_i form the *body (premises)* of the rule and B is the *head*; both A_i and B are atomic formulas. Depending on the interpretation of the rule, it can be either *deductive* or *reactive*. In case of a deductive rule, if the body is known to be true then the head is inferred as true. For a reactive rule, B represents an action which should be carried out if body is true. All the variables introduced in the rules are implicitly universally quantified. There are two main types of inference rules: *monotonic* and *non-monotonic*.

Monotonic rules (see Example 2.4) do not utilize negation (e.g. negation as failure or true negation). “The definite clause logic is monotonic in the sense that anything that could be concluded before a clause is added can still be concluded after it is added; adding knowledge does not reduce the set of propositions that can be derived” [195, Sec. 5.5.1].

Non-monotonic rules (see Example 2.5) use negation as failure or true negation and, as a result, some conclusions can be invalidated by adding more knowledge while applying inference rules. These rules are called *defeasible* because they can be defeated by other rules. Non-monotonic rules introduce additional complexity and the corresponding logic and logic programs are of high interest in the Artificial Intelligence community.

Example 2.4 (Monotonic Rules). *Continuing our discussion regarding the Example 2.1, for the imaginary world where all blind users use Blindzilla²², the following statement expressed in terms of Horn logic will hold: $BlindUser(X) \rightarrow use(X, blindzilla)$.*

The statement “If blind users bought the same assistive technology then they are customers of the same company” can be written accordingly as $BlindUser(X), BlindUser(Y), X \neq Y, bought(X, Z), bought(Y, Z), AssistiveTech(Z) \rightarrow customersOfSameCompany(X, Y)$. (The instance “blindzilla” and the predicates “bought” and “customersOfSameCompany” should be specified in the ontology).

Example 2.5 (Non-monotonic Rules). *Using true negation we can express the example statement “If an assistive technology is expensive and not multifunction then it is not acceptable” as follows: $AssistiveTech(X), Expensive(X), \neg Multifunction(X) \Rightarrow \neg Acceptable(X)$.*

A mapping between ontology and declarative rule-based languages such as RuleML and Semantic Web Rules Language (SWRL) is important for many aspects of the Semantic Web, and in particular for the language layering: “A key requirement for the Semantic Web architecture is to be able to layer rules on top of ontologies to create and reason with rulebases that mention vocabulary specified by ontology-based knowledge bases and to do so in a semantically coherent and powerful manner” [134, p. 122]. Furthermore, rule-based languages allow integrating additional expressivity into the ontology-based systems which is not covered by the ontology language.

Multiple approaches and languages have been developed to integrate rules- and DL-based ontologies. As we know, Horn logic and DL are orthogonal. One of the approaches to achieve their integration in one framework is to consider the intersection of both logics. Such an intersection is called **Description Logic Programs (DLP)**; it is the Horn-definable part of OWL or, in other words, the OWL-definable part of Horn logic. DLP captures a significant fragment of OWL DL, including the whole OWL DL fragment of RDF(S). Therefore, DLP has certain advantages, for

²²Assistive technology introduced in Chapter 6

instance: 1) there is freedom for the ontology modeler to use either OWL DL or rules; 2) from the implementation perspective, either DL reasoner or deductive rule systems can be used, that ensures interoperability with a variety of tools.

OWL 2 RL [280], the language profile of OWL 2, represents a rule subset of OWL 2. Its design was mainly inspired by DLP. The standard reasoning in OWL 2 RL is P_{Time} -complete and can be implemented using DLP. Among the reasoners available for this profile language, there are OWLIM, Jena, Oracle OWL Reasoner, etc.

In contrast to DLP, the **Semantic Web Rules Language (SWRL)** [252] combines OWL DL with function-free Horn logic, written in Datalog RuleML [283]. A rule in SWRL has the form $A_1, A_2, \dots, A_n \rightarrow B_1, B_2, \dots, B_m$, where the commas denote conjunction, and every atom (A_i and B_j) in the rule can be an OWL descriptor, OWL property, and every atom is either a variable, OWL individual, or OWL data value. An empty *antecedent* (premises, body) of the rule is treated as trivially true (i.e. satisfied by every interpretation), so the *consequent* (head) must also be satisfied by every interpretation; an empty consequent is treated as trivially false (i.e. not satisfied by any interpretation), so the antecedent must also not be satisfied by any interpretation. SWRL allows property chains which are only available in OWL 2 DL, and directly using the OWL expressions in the body or head of the rule. The latter enables seamless integration of rules and ontologies. Example 2.6 demonstrates the application of SWRL by example of the ontology introduced in Example 2.3.

Example 2.6. Statement “If a blind user uses assistive technology then she is efficient” in SWRL XML Concrete Syntax is presented as follows:

```

1 <ruleml:imp>
2   <ruleml:_body>
3     <swrlx:classAtom>
4       <owlx:IntersectionOf>
5         <owlx:Class owlx:name="BlindUser" />
6         <owlx:ObjectRestriction owlx:property="use">
7           <owlx:someValuesFrom owlx:class="AssistiveTech" />
8         </owlx:ObjectRestriction>
9       </owlx:IntersectionOf>
10      <ruleml:var>efficient</ruleml:var>
11    </swrlx:classAtom>
12  </ruleml:_body>
13  <ruleml:_head>
14    <swrlx:classAtom>
15      <owlx:Class owlx:name="Efficient"/>
16      <ruleml:var>efficient</ruleml:var>
17    </swrlx:classAtom>
18  </ruleml:_head>
19 </ruleml:imp>

```

It is essential to note **F-logic** [139] as a knowledge representation and ontology language. It stands in the same relationship to object-oriented programming as classical predicate calculus stands to relational database programming [134, Sec. 5.3.2].

A **Rule Markup Language (RuleML)** [283] is an important standardization effort for markup of rules on the Web. It is a family of rule markup languages that corresponds to different kinds of rule languages, e.g. derivation rules, integrity constraints, reaction rules. Datalog (function-free Horn logic) is the core of the RuleML. The RuleML Initiative also supports the

development of Rule Interchange Format (RIF) [287], a format proposed by W3C that allows logic rules to be exchanged between rule systems. We also want to note that despite the fact that RDF is not an ontology language, there are inference systems which are sound and complete for RDF semantics. The reasoning mainly focuses on the class-subclass, property-subproperty relations.

2.6.3 Standard Ontology Reasoning Services

There are four main reasoning services provided by reasoners [221, Sec. 3.2]:

Consistency checking. The reasoner checks if a given ontology \mathcal{O} is consistent, i.e. if there exists a model (a model-theoretic instance) for \mathcal{O} .

Satisfiability checking. The reasoner finds all unsatisfiable concepts in a given ontology \mathcal{O} . A concept is unsatisfiable if it cannot be instantiated.

Classification. The classification service returns for a given ontology \mathcal{O} and individual I a set of concepts which contain the individual.

Subsumption. The subsumption checking service checks whether the interpretation of A (the set of individuals contained in A) is a subset of the interpretation of B for a given ontology \mathcal{O} .

Among the reasoning tools it is worth mentioning FaCT++²³, Racer²⁴, Pellet²⁵, and Jena²⁶ [10].

2.6.4 SPARQL

SPARQL is the standard language for querying RDF documents. It provides a basic ontology query service. It is a W3C Recommendation [257] which consists of three separate specifications which describe a *query language*, a *data access protocol* which uses WSDL 2.0 for querying remote databases, and *query result formatting* based on XML. In this section we focus on the SPARQL query language. It consists of the following components [134, Sec. 4.3.2]:

Graph Patterns: The SPARQL query language is based on matching graph patterns. The simplest graph pattern is the triple pattern with the possibility of a variable instead of an RDF term in the subject, predicate, or object position. A combination of triple patterns gives a basic graph pattern that is utilized to identify a subset of the RDF graphs to be retrieved.

RDF Dataset: SPARQL can be executed against multiple RDF graphs, that allows an application to make queries which involve information from more than one graph.

Solution Modifiers: Query patterns matched generate an unordered collection of solutions which are then treated as a sequence, initially in no specific order. Sequence modifiers are then applied to specify a sequential order and information to be included in the result.

Query Form: The final sequence generated from the solution modifiers is used to present a query result. There are four query forms SELECT, CONSTRUCT, ASK, and DESCRIBE.

²³<http://owl.man.ac.uk/factplusplus/>

²⁴<http://www.racer-systems.com/>

²⁵<http://clarkparsia.com/pellet/>

²⁶<http://jena.apache.org/>

SPARQL is considered as one of the key technologies of the Semantic Web. It allows for a query to consist of triple patterns, conjunctions, disjunctions, and optional patterns. SPARQL provides an efficient query answering, while the evaluation of general SPARQL patterns is PSPACE-complete [192]. It is well supported by various applications including Jena, Pellet, Bigdata, and Sesame.

Example 2.7 demonstrates the use of SPARQL language.

Example 2.7. *For the ontology presented in Example 2.3, the SPARQL query “Get all users who use assistive technology but are not blind” will look like this:*

```
1 SELECT DISTINCT ?user
2 WHERE {
3 ?user rdf:type :User.
4 ?user :use ?soft.
5 ?soft rdf:type :AssistiveTech.
6 MINUS { ?user rdf:type :Sightless }
7 }
```

2.7 Discussion

In this chapter, we presented the state of the art in the field of Web Accessibility (Sections 2.1 and 2.2) and Web Page Processing (Sections 2.3 and 2.4). We also mention Semantic Web technologies as an important aspect of the accessible Web (Sections 2.5 and 2.6).

Among the issues concerning web accessibility, the crucial ones include: 1) absence of semantic description of the content of web pages and 2) insufficient investigation of various possibilities of navigation over the logical structure of a web page.

1. Providing the web pages with semantically rich content can give the blind user a possibility to understand the functional role of elements of the web page interface. For instance, to recognize navigation menu, title of the article, search field and know how to interact with them. Due to the fact that most of web pages do not follow W3C accessibility guidelines (see Section 2.1.1) and do not provide necessary semantic labels, automatic enhancement of accessibility for inaccessible web pages becomes a necessity. It is also confirmed by the active development of the specialized web browsers, transcoding technologies, and screen readers (see Section 2.1.3). We consider a problem of web page analysis for enhancing its accessibility from the viewpoint of WPU and WIE (see Section 2.3). Investigating various methods of WPP, we came to the conclusion that it is important to consider visual characteristics of the web page for developing robust methods (see Section 2.4). Due to the absence of a common model for representing rendered web pages convenient for WPP, we found the importance of developing the unified model of a web page. Thus, in Chapter 3, we formally describe geometric characteristics of web page elements taking into account fuzziness for different spatial relationships. In Chapter 4, we introduce the Unified Ontological Model (UOM) dedicated to the WPP and describing various aspects of the web page, including the visual, functional, and structural represented by the DOM trees. For the analysis of a web page based on the model proposed, we developed a WPPS framework presented in Chapter 5. It enables the development of WPP methods and enriching the web page with semantic labels.

2. Investigating different contemporary methods of web page navigation (Section 2.2), we came to the conclusion that contemporary methods and tools do not pay enough attention to the navigation (Section 2.2.3). In particular, we propose to consider the navigation not from the viewpoint of locomotion through the elements of the DOM tree as the contemporary screen readers do, but as a problem of one-dimensional navigation through the multi-dimensional logical structure of a web page. The one-dimensional character of navigation is unfortunately conditioned by the use of an aural perceptual channel with Braille display and speakers, and the absence of the visual perception. The enhancement of web page navigation is proposed in Chapter 6, where we introduce a Multi-Axial Navigation Model (MANM) together with a navigation methodology.

Modeling the Geometric Structure of a Web Page

Fundamental to this development is the analysis of geometrical structures in relation to the formal languages used to describe them, and the recognition of the special mathematical challenges—and opportunities—which such an analysis presents.

— Marco Aiello, Ian Pratt-Hartmann, Johan van Benthem, *Handbook of Spatial Logics*, 2007

In this chapter, we conduct an analysis of web pages' layout to identify the main features and spatial relationships which are further used to define a web page's geometric structure and which underlie the Block-based Geometric Model (BGM) (see Section 4.4.2). A geometric structure defines spatial configurations of elements that can be used for the object identification and web page understanding, which in turn plays an important role in enhancing the accessibility of web pages [83, 87, 121].

Section 3.1 provides a brief overview of various geometric representations of electronic documents and the relationships used. Then we specify terms such as *quantitative* and *qualitative models* (see Section 3.2). In Section 3.3, we introduce the concepts of *web page canvas* and discuss the utilized units of measure. In Section 3.4, we empirically define a concept of geometric object together with its attributes, and in Section 3.5, perform the analysis of various spatial relationships expressed quantitatively or qualitatively, in particular: topological (see Section 3.6), direction (see Section 3.7), distance (see Section 3.8), alignment (see Section 3.9), interval relations, and Two-Dimensional Interval Relations (2DIR) (see Section 3.10). In Section 3.10, we also introduce a fuzziness in the interval relations which is used to reflect peculiarities of human perception; furthermore, we express a major part of introduced qualitative relations via 2DIR, taking into account fuzziness. 2DIR give us the possibility to dramatically decrease the amount of possible contradictions in the mutual application of various types of spatial relationships.

In Section 3.11, we evaluate main spatial relations on the WPPS-HTML-DS1 dataset [72]. Section 3.12 concludes the chapter.

3.1 Existing Geometric Structures

A *geometric model* of an electronic document defines its structure, layout and plays an important role in research fields such as Document Understanding [227], Web Data Extraction [122, 288], and Document Classification [54]. In some ways, a geometric model is an abstraction of the original format of an electronic document, which allows the consideration of only the necessary information like spatial relations and geometric configurations. Thus, some forms of document layout representations are suitable for different types of documents, be it a scanned (raster) document, a PDF document, or a web page.

A significant amount of work is dedicated to representing a geometric structure of scanned documents [110, 227, 288]. However, this question is not usually addressed specifically to a web page. Web authors often confine themselves to the aspects of web page layout that are necessary for solving some particular problems; be it the extraction of web articles [169] or relatively simple objects [309] or web page classification [146] with the use of four direction relations between adjacent CSS boxes (see Section 2.4.7). Examples of works which address the problem of modeling web page layout as a whole are [144, 190].

The most widespread representation of a document layout is based on an *inclusion relation*, which makes it possible to represent the document structure as a tree. This representation can be acquired via document segmentation (e.g. X-Y cut algorithm [182] and its modifications [176], or the VIPS algorithm [41, 42]). It is used both for web pages and for scanned documents, playing an important role in tasks such as document understanding [63, 118] and web information extraction [308]. Inclusion relations provide an opportunity to operate “top-down,” analyzing smaller visual elements included in larger ones step by step [95]. Another way for geometric structure representation is mainly based on direction relationship, which is established between neighboring elements and represented as an adjacency graph [116, 309]. This representation—in conjunction with alignment—is used in different tasks including web page classification [146] and web form understanding [305]. In [288], Walischewski uses Allen’s *interval relations* [7] between x and y projections of *geometric objects* for document interpretation. Aiello et al. [2] introduce their modification of interval relations, TBRR, for tasks of document understanding and, in particular, for reading order detection. Models, based on interval relations and direction, allow the use of so-called “bottom-up” approaches which also give quite promising results. More complex geometry of electronic documents, which was applied to the web page and used for web adaptation tasks, is presented by J. Kong et al. in the Spatial Graph Grammar (SGG) formalism [144]. The authors use various spatial relationships: certain topological relations, distance notion, alignment and direction. In [190], E. Oro et al. enrich DOM tree of a web page with additional relations adopted from *rectangular algebra*.

Some methods do not utilize relations mentioned above (which qualitatively describe geometric structure) and mainly use quantitative visual characteristics (e.g. coordinates of a web page’s elements) along with DOM tree (e.g. extraction of data records [122] and web articles [169], see Section 2.4.6). Therefore, both quantitative and qualitative information are important.

A web page's layout differs from a scanned document's layout. Besides the main content (textual and multimedia), a web page includes a variety of web objects that have some semantic roles (e.g. navigation menu, web form elements, posts in a web forum). This fact defines web pages not only as resources of textual or multimedia information, but also as web applications. Moreover, the layout of web pages comply with the CSS specifications [260] that define a dynamic behavior of the layout which depends on the CSS rules of the web page, a web browser and the size of a viewport. In contrast to scanned documents, a web page provides some granularity, which is limited by CSS boxes [260, Sec. 8]. Information such as text and computed CSS attributes can be acquired from the web browser. CSS rules of a web page along with its Document Object Model (DOM) tree are used by the browser's layout engine merely for generating the visual representation of a web page. CSS rules are not able to describe qualitatively spatial configurations and qualitatively express positional information of visual elements. In some sense, a CSS-based representation of a web page with computed attributes of the laid out CSS boxes is the original representation of a web page, in the same way as a raster image for a scanned document.

These peculiarities define a necessity of developing a specific web pages' geometric model (i.e. the BGM) that can be utilized in various fields of research such as Web Page Understanding (WPU) and Web Information Extraction (WIE). Thus, we are intending to consider various spatial attributes and relationships in quantitative and qualitative representation (including the relations mentioned by J. Kong et. al. and E. Oro et al.), and introducing it in one formalism. We are interested in modeling various spatial and geometric characteristics which are adequate to the CSS [260] standards, reflect peculiarities of human perception of the layout and convenient for the web page processing.

3.2 Quantitative and Qualitative Models

Modeling different geometric structures of the web pages, we distinguish between quantitative and qualitative information, and between quantitative and qualitative models accordingly. Quantitative information provides us with exact information using specific units of measure. When considering the distance between geometric objects of a web page, it can be measured in pixels while for direction it is measured in angles. Precise information is used in various quantitative methods (some of them are listed in Section 2.4.6). In contrast, qualitative information is more natural for human perception and can be used in the automatic logical inferences. A statement that an image is *far* from the button (instead of specifying a distance in pixels, for instance, 2858px), or saying that a caption is *under* the image (instead of asserting that the angle between geometric centers of the caption and the image is 78.4°) is cognitively more eloquent and immediate; it is perceived more natural by humans. Regarding the reasoning, if we have the image *far* to the *east* of the button and radio button to the *east* of the image, one can infer that the radio button is also *far* to the *east* of the button. Methods which use qualitative visual characteristics are presented in Section 2.4.7.

In Definition 3.1 and Definition 3.2, we formally specify concepts of “quantitative” and “qualitative” models which underlie the BGM (see Section 4.4.2)—a metamodel of a web page's layout. An *object* is the main element for both models. For the layout, it is a geometric object

(see Section 3.4 and 4.4.2). In the definitions, we do not take various types of objects and their taxonomy into account.

Definition 3.1 (Quantitative Model). *A quantitative model is a triple $\mathcal{M}^{\text{qnt}} = \langle \Theta, A^{\text{qnt}}, B^{\text{qnt}} \rangle$. Θ is a set of objects in the model \mathcal{M}^{qnt} . $A^{\text{qnt}} = \{\alpha_1^{\text{qnt}}, \dots, \alpha_n^{\text{qnt}}\}$, where $\alpha_i^{\text{qnt}} : \Theta \rightarrow t_i^\alpha$ defines values of the objects' attributes, t_i^α is a range of the attribute α_i^{qnt} . $B^{\text{qnt}} = \{\beta_1^{\text{qnt}}, \dots, \beta_m^{\text{qnt}}\}$, where $\beta_j^{\text{qnt}} : \Theta \times \Theta \rightarrow t_j^\beta$ defines quantitatively expressed relations between objects; t_j^β is a range of the quantitative relation β_j^{qnt} .*

Example of the attribute in \mathcal{M}^{qnt} can be position $\alpha_{\text{pos}}^{\text{qnt}}$ in pixels with range $t_{\text{pos}}^\alpha \in \mathbb{R} \times \mathbb{R}$ (see Section 3.4). Example of the quantitative relation can be direction $\beta_{\text{dir}}^{\text{qnt}}$ defined as an angle in degrees $t_{\text{dir}}^\beta \in [0; 360)$ (see Section 3.7).

In the definition of a qualitative model, we do not mention the ordering of qualitative values as well as the frame of reference.

Definition 3.2 (Qualitative Model). *A qualitative model is a triple $\mathcal{M}^{\text{qlt}} = \langle \Theta, \Phi^{\text{I}}, \Phi^{\text{II}} \rangle$. Θ is a set of objects in the model \mathcal{M}^{qlt} . $\Phi^{\text{I}} = \langle A^{\text{qlt}}, V, \psi^{\text{I}}, \alpha^{\text{qlt}} \rangle$ defines attributes of the objects. $A^{\text{qlt}} = \{a_1, \dots, a_n\}$ is a set of attributes; V is a set of qualitative values of the attributes; $\psi^{\text{I}} : A^{\text{qlt}} \rightarrow 2^V$ defines the range for each attribute a_i ; $\alpha^{\text{qlt}} : \Theta \times A^{\text{qlt}} \rightarrow 2^V$ assigns sets of values for the attributes of the objects. $\Phi^{\text{II}} = \langle B^{\text{qlt}}, R, \psi^{\text{II}}, \beta^{\text{qlt}} \rangle$ defines qualitative relationships on the set Θ . B^{qlt} is a set of various types of qualitative relations; R is a set of binary relations which can be defined between objects; $\psi^{\text{II}} : B^{\text{qlt}} \rightarrow 2^R$ defines a set of relations for each type of relations; $\beta^{\text{qlt}} : \Theta \times \Theta \times B^{\text{qlt}} \rightarrow 2^R$ defines binary relations between objects.*

A necessary properties: Strictly one attribute and only one type of relation should be set in correspondence to every value of an attribute and for every relation respectively, i.e.

$$\begin{aligned} \forall a_i \in A^{\text{qlt}}, a_j \in A^{\text{qlt}} [a_i \neq a_j \rightarrow \psi^{\text{I}}(a_i) \cap \psi^{\text{I}}(a_j) = \emptyset], \\ \forall b_i \in B^{\text{qlt}}, b_j \in B^{\text{qlt}} [b_i \neq b_j \rightarrow \psi^{\text{II}}(b_i) \cap \psi^{\text{II}}(b_j) = \emptyset]. \end{aligned} \quad (3.1)$$

If an object in the qualitative model has more than one value for an attribute (i.e. $\exists \theta \in \Theta, a \in A^{\text{qlt}} [|\alpha^{\text{qlt}}(\theta, a)| > 1]$) or if a pair of objects has more than one relation of the same type (i.e. $\exists \theta_i \in \Theta, \theta_j \in \Theta, b \in B^{\text{qlt}} [|\beta^{\text{qlt}}(\theta_i, \theta_j, b)| > 1]$), then the model has *uncertainty*.

$a \in A^{\text{qlt}}$ and $b \in B^{\text{qlt}}$ can be interpreted as linguistic variables, whereas $\psi^{\text{I}}(a)$ and $\psi^{\text{II}}(b)$ as their values respectively. Example of the attribute can be “width” $a_{\text{width}} \in A^{\text{qlt}}$ of an objects with values such as “small,” “commensurate,” and “big” defined in V and related to a_{width} by ψ^{I} (see Section 3.4). Example of a relation can be relationship such as “alignment” $b_{\text{align}} \in B^{\text{qlt}}$ with values “left aligned,” “right aligned,” “centered vertically,” etc. defined in R and related to b_{align} by ψ^{II} (see Section 3.9). The number of possible values for each attribute and relation is defined by the chosen level of granularity (e.g. for the example with “width” it is 2).

In this chapter, we introduce different qualitative relationships and relate them to the quantitative data. In general, for the system of linguistic terms with a predefined order, such as width (see Section 3.4) or distance (see Section 3.8), we give the following definition:

Definition 3.3. *System of qualitative values is a triple $\langle R^{\text{qlt}}, \mathcal{D}^{\text{qlt}}, \nu^{\text{qlt}} \rangle$, where $R^{\text{qlt}} = \{\rho_1, \rho_2, \dots, \rho_n\}$ is a set of qualitative symbols (or linguistic terms) which form a complete*

lattice with $\rho_1 \prec \rho_2 \prec \dots \prec \rho_n$; $\mathbf{D}^{\text{qlt}} = \{D_{\rho_1}, D_{\rho_2}, \dots, D_{\rho_n}\}$ is a set of fuzzy sets, where, for every ρ_i , $D_{\rho_i} = \{\langle x, \mu_{\rho_i} \rangle \mid x \in [0, +\infty)\}$ with the membership function $\mu_{\rho_i} : [0, +\infty) \rightarrow [0, 1]$; $\nu^{\text{qlt}} : [0, +\infty) \rightarrow 2^{\mathbf{R}^{\text{qlt}}}$ maps quantitative data to the qualitative symbols. $\nu^{\text{qlt}}(x) = \{\rho_i \mid \mu_{\rho_i}(x) > \varepsilon\}$, where $0 < \varepsilon \leq 1$ is a predefined threshold value which used to define a crisp mapping ν^{qlt} .

This system has the following necessary properties:

1. Every qualitative symbol ρ_i has at least one quantitative value which corresponds only to it, i.e. $\forall \rho_i \exists x [\nu^{\text{qlt}}(x) = \{\rho_i\}]$;
2. Every quantitative value corresponds to at least one qualitative symbol, i.e. $\forall x \geq 0 [\max(\mu_{\rho_1}(x), \mu_{\rho_2}(x), \dots, \mu_{\rho_n}(x)) > \varepsilon]$;
3. The order of quantitative values correspond to the order of qualitative symbols, i.e. $\forall x_1, x_2, \rho_i, \rho_j [0 \leq x_1 \leq x_2 \rightarrow (\rho_i \in \nu^{\text{qlt}}(x_1) \setminus \nu^{\text{qlt}}(x_2) \wedge \rho_j \in \nu^{\text{qlt}}(x_2) \setminus \nu^{\text{qlt}}(x_1) \rightarrow \rho_i \prec \rho_j)]$;
4. For every quantitative value $x \geq 0$, there exists one or several qualitative symbols which correspond to one interval without holes in the ordered sequence of qualitative symbols, i.e. $\forall \rho_i \in \nu^{\text{qlt}}(x), \rho_j \in \mathbf{R}^{\text{qlt}} [\rho_j \neq \rho_i \rightarrow (\rho_j \prec \rho_i \vee \rho_j \succ \rho_i)]$

One of the peculiarities of this definition is that it specifies relation between qualitative and quantitative values in terms of fuzzy sets [96]. For μ_{ρ_i} with $i = 2, 3, \dots, n - 1$ a Π -like membership function is recommended. Examples of the Π -like functions are trapezoidal or triangular functions, bell-shaped function, normalized Gaussian function, etc. [131]

3.3 Web Page Canvas and Unit of Measure

Definition 3.4 (Web Page Canvas). A web page canvas (or document canvas, see Section 3.4) is a plane with Euclidean metric space defined on it and pixels as a unit of measure. A canvas lies in the plane of screen with Cartesian coordinate system, abscissa is directed from left to right and ordinate from top to bottom. The top-left corner of the top level page from the page hierarchy specifies the origin of coordinates (see Figure 3.1).

We also differ a page canvas (see Section 3.4) which has similar specification of its coordinate system, however, it is mainly referred to a specific page from the page hierarchy of a web page. The first quadrant of a page canvas is a visible part of a canvas. Our definition of a canvas correlates with the definition provided by the W3C in [260, Sec. 2.3.1] (see Section 2.4.5).

In this thesis, we make a distinction between a pixel as a unit of measure and a pixel as a region which corresponds to the smallest addressable element in a display device. We adopt pixel unit of measure presented by the web browsers according to the W3C recommendation [260, Sec. 4.3.2]. The coordinates of the rendered elements on a web page canvas can be fractional. The task of the web browser layout engine is to map coordinates of elements to particular pixels of a screen according to the chosen level of magnitude. Consideration of this problem is out of the scope of this thesis. We focus on the geometry of objects on the canvas instead of the problem of their visualization on the discrete screen.

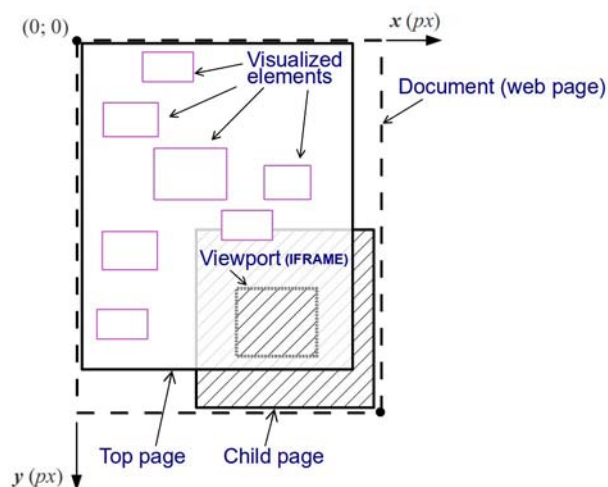


Figure 3.1: Web page structure

3.4 A Web Page Structure, Geometric Object and its Attributes

We distinguish five main types of structural elements of a web page (see Figure 3.1): *document*, *page*, *viewport*, *visualized element*, and *box*.

A **document** is a web page rendered by the web browser's engine. It is formed by the set of X/HTML or XML files connected with each other by means of *inclusion* (e.g. by elements with the names `FRAME`, or `IFRAME`, or `OBJECT`). Width and height of the document correspond to the dimensions of a minimum bounding rectangle wrapping corresponding rendered files—*pages*. A **page** is a single rendered X/HTML or XML file of a document, which can be modeled as a rectangle. It refers to the DOM window in the Browser Object Model (BOM) and CSS Object Model (CSSOM). A set of pages make a *hierarchy of pages* through the inclusion relations. From another point of view a page is a DOM-tree together with computed CSS attributes; it has the counterpart `Window` in the BOM [136]. By the term **viewport**, we mean both viewport of the web browser [260, Sec. 9.1.1] and viewport rendered by the web browser to view child pages (see Figure 3.1).

According to the CSS 2.1 specification (see Section 2.4.5) [260], a CSS box is the minimum visual formatting object which corresponds to the visualized node of the DOM. From the viewpoint of visual perception of the CSS boxes (see Figure 2.8 on page 32, [260, Sec. 8]), the observer can see only the border area of the CSS box model and its inner area which is formed by the conjunction of the padding and content areas. The model of a **visualized element**, the counterpart of the CSS box, is presented in Figure 3.2. It is important to note that CSS box in general is a complex object which can span through several lines forming sequence of client rectangles [264, 286] which are also CSS boxes. In terms of a geometric model of a web page, these client rectangles are referred to as **boxes**. For the analysis, it is reasonable to simplify the box as a geometric object, and represent it as a union of two rectangular areas (which are also geometric objects): *outer* and *inner blocks*. This description of the box is used in the BGM, see

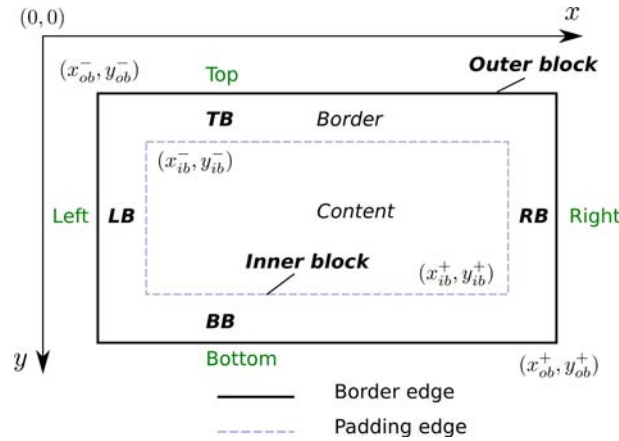


Figure 3.2: Model of the CSS box in the Block-based Geometric Model (BGM) with quantitative positional information

Section 4.4.2).

A geometric (layout) object is an element of the specific geometric structure of a document [228]. In general, a geometric object can be of different shapes [110], such as circle, ellipse, square, rectangle, polygon, but rectangles are more widespread [227]. Thus, different structural elements of a web page, which are geometric objects, we model by minimum bounding rectangle—a *block*.

Definition 3.5 (Block). A block is a rectangular region with the nonzero area which lies on the web page canvas and is defined by its extreme points: top-left (x^-, y^-) and bottom-right (x^+, y^+) corners.

Considering a geometry of the block as part of the BGM, we define the following quantitative attributes (see Section 3.2): 1) *position* ($\alpha_{\text{pos}}^{\text{qnt}}$): (x^-, y^-) , 2) *width* ($\alpha_{\text{width}}^{\text{qnt}}$): $x^+ - x^-$, 3) *height* ($\alpha_{\text{height}}^{\text{qnt}}$): $y^+ - y^-$, and 4) *draw id* ($\alpha_{\text{draw id}}^{\text{qnt}} \in \mathbb{N}$). The draw id is a sequence number which corresponds to the order of painting blocks by the web browser engine on the canvas.

For the qualitative representation (see Section 3.2) of the width ($a_{\text{width}} \in A^{\text{qlt}}$) and height ($a_{\text{height}} \in A^{\text{qlt}}$) of the block, we can use linguistic terms such as “very small,” “small,” “commensurate,” “big,” “very big” with their natural order (e.g. “very small” \prec “small” $\prec \dots \prec$ “very big”) according to the systems of qualitative values (see Definition 3.3 on page 64). According to Definition 3.2 and the necessary property of the system of qualitative values (3.1), symbols referring to corresponding linguistic terms should differ for the width and height. For example, we can use value $v_{\text{big width}}$ for the width and $v_{\text{big height}}$ for the height of the block. The position and draw id can be specified qualitatively relative to other objects via various types of spatial relations.

It is to be noted that a CSS box can have an outline which is drawn separately from it [260, App. E]. This object is perceived visually as a separate CSS box and thus it can also be replaced by the inner and outer blocks in the BGM.

3.5 Spatial Relations

There are several types of spatial relationships, such as *topological* (see Section 3.6), *direction* (see Section 3.7), *distance* (see Section 3.8), *alignment* (see Section 3.9), *interval relations* and the relations based on them (see Section 3.10) widely applied in the fields of spatial cognition [181], Geographical Information Systems (GIS), Computer-Aided Design (CAD) [50], graphical user interfaces design [108, 144, 206] and also the web page layout analysis [94]. Distance and direction can be expressed either quantitatively or qualitatively, while the rest only represents relationships qualitatively. These peculiarities of different types of relationships define their relation to either quantitative or qualitative models (see Section 3.2). This set of relationships provides us with sufficient information about spatial configuration and position of objects on the web page canvas.

When introducing various spatial relationships, it is worth mentioning concepts such as a *primary*, *reference objects*, and a *frame of reference*. These three terms are important for the identification of spatial allocation of the objects. A primary object is an object which spatial location and orientation we would like to identify. A reference object is used to specify position of the primary object relative to it. A frame of reference is used for defining spatial relations. For example, for the quantitative distance, we specify the frame of reference by the coordinate axes of the web page canvas (see Section 3.3).

3.6 Topological Relations

Topology is one of the fundamental aspects of space. It defines qualitative relations between geometric objects and forms a fundamental basis of qualitative spatial reasoning [52]. Topological relations are able to describe all aspects of the scene which are invariant with respect to common linear transformations (translation, rotation, rubber sheeting) [50]. One of the most common topological relations which are suitable for our needs are from the mereotopology of the Region Connection Calculus (RCC), a first-order formalism [53, 203]. It is the region-based approach [3, Sec. 3.1.2]. The fundamental relation which underlie it is a *connection relation*. We give the following definition for this term, which correlates with definition given by B.L. Clarke [49].

Definition 3.6 (Connection). *A relation $C(r, s)$ (read as “ r connects with s ”) is defined on the set of regions (spatial or temporal) between those pairs whose topological closures share a common point.*

Connection relation is reflexive and symmetric, i.e. $\forall r[C(r, r)]$ and $\forall r, s[C(r, s) \rightarrow C(s, r)]$. In addition to Definition 3.5, we define a block taking into account mereotopology.

Definition 3.7 (Block). *A block is a non-empty connected closed region without holes which lies on the web page canvas.*

A region r is *connected* if for every pair of points in r there exists a line (not necessarily a straight one) joining the two points such that all the points on the line are also in r . The property of connectedness prevents a region from having disjoint parts [216]. It is important to note that a

topological space defined by the set of blocks is not a topology (the union and intersection of any closed/open rectangle is not in general a closed/open rectangle).

Considering the mereotopological definition of $C(r, s)$ and the spatial expansion of the blocks r and s in the Euclidean metric space, $C(r, s)$ can be defined as follows¹:

$$C(r, s) = x_r^+ \geq x_s^- \wedge x_r^- \leq x_s^+ \wedge y_r^+ \geq y_s^- \wedge y_r^- \leq y_s^+. \quad (3.2)$$

Concerning the blocks, we use RCC8 which was introduced in the area of GIS [64, 217] and as a decidable subset of Region Connection Calculus (RCC) [203, 204].

3.6.1 RCC8

RCC8 is able to represent relation between pair of regions and it does not distinguish between connected and disconnected regions or regions with and without holes [3, p. 513]. It has eight jointly exhaustive and pairwise disjoint (JEPD) dyadic relations (see Figure 3.3) which are defined purely by means of connection relation $C(r, s)$ (3.3):

- 1) EQUAL(r, s) or $r = s$: r is identical to s ,
- 2) EC(r, s): r is externally connected with s ,
- 3) DC(r, s): r is disconnected from s ,
- 4) PO(r, s): r partially overlaps s ,
- 5) TPP(r, s): r is tangential proper part of s ,
- 6) NTPP(r, s): r is nontangential proper part of s ,
- 7) TPP⁻(r, s): inverse of TPP(r, s),
- 8) NTPP⁻(r, s): inverse of NTPP(r, s).

Listed JEPD relations can be subsumed into more general relations (see Figure 3.3) which are also defined by means of $C(r, s)$ (3.3):

- 1) PP(r, s): r is a proper part of s ,
- 2) P(r, s): r is a part of s ,
- 3) O(r, s): r overlaps s ,
- 4) DR(r, s): r is discrete from s ,
- 5) PP⁻(r, s): inverse of PP(r, s),
- 6) P⁻(r, s): inverse of P(r, s).

For the topological relations, we use notations $*(r, s)$ and $\tau_*(r, s)$ interchangeably, where $*$ is a placeholder; for instance, $C(r, s)$ and $\tau_C(r, s)$.

It is worth mentioning that all the relations except P, PP, TPP, NTPP, and their inverses are symmetric. Moreover, these relations can be embedded in a relational lattice based on the subsumption relation. This is given in Figure 3.3, where the symbol “ \top ” is interpreted as

¹This definition gives one a possibility to relate all qualitative relations based on this to the quantitative values in metric space.

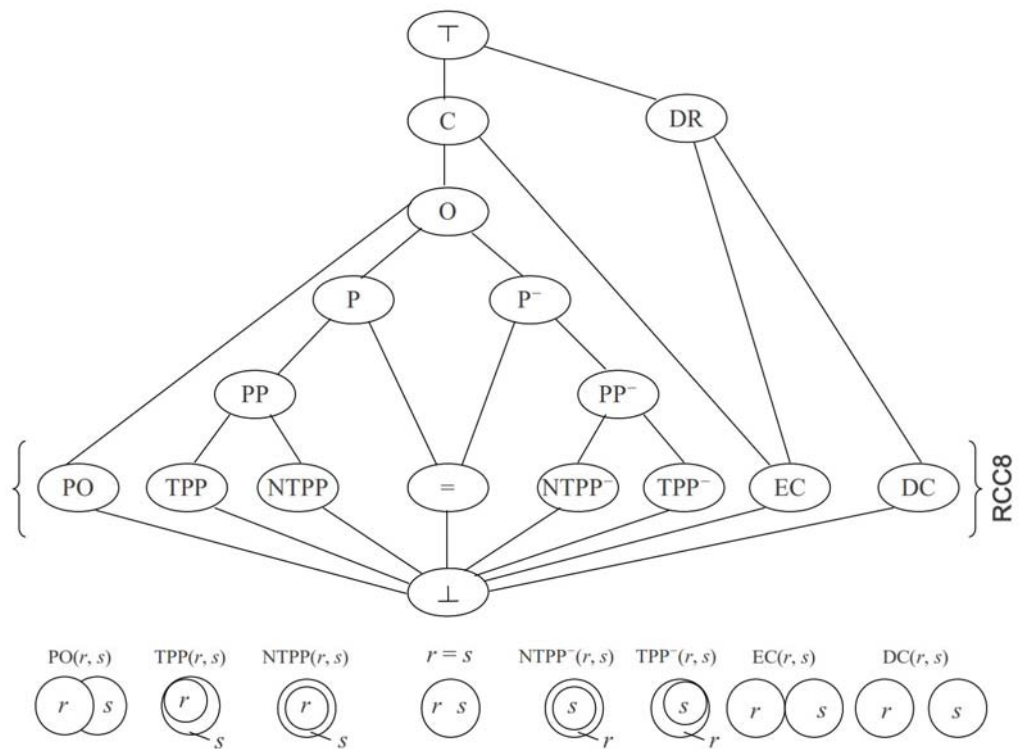


Figure 3.3: A lattice defining the subsumption hierarchy of the dyadic relations defined solely in terms of the basic relation $C(r, s)$

tautology and the symbol “ \perp ” as contradiction. The partial order based on the subsumption (inclusion relation) is set in such a way that a weaker (more general) relation is always above a stronger (more specific) one. The definitions of the set of RCC8 relations is presented in (3.3), where they are expressed by means of $C(r, s)$. The analysis of the RCC8 JEPD applied to the web page layout can be found in Section 3.11.2.

$$\begin{aligned}
\mathbf{DC}(r, s) &\equiv_{def} \neg \mathbf{C}(r, s), \\
\mathbf{P}(r, s) &\equiv_{def} \forall z[\mathbf{C}(z, r) \rightarrow \mathbf{C}(z, s)], \\
\mathbf{PP}(r, s) &\equiv_{def} \mathbf{P}(r, s) \wedge \neg \mathbf{P}(s, r), \\
\mathbf{EQUAL}(r, s) &\equiv_{def} \mathbf{P}(r, s) \wedge \mathbf{P}(s, r), \\
\mathbf{O}(r, s) &\equiv_{def} \exists z[\mathbf{P}(z, r) \wedge \mathbf{P}(z, s)], \\
\mathbf{PO}(r, s) &\equiv_{def} \mathbf{O}(r, s) \wedge \neg \mathbf{P}(r, s) \wedge \neg \mathbf{P}(s, r), \\
\mathbf{DR}(r, s) &\equiv_{def} \neg \mathbf{O}(r, s), \\
\mathbf{EC}(r, s) &\equiv_{def} \mathbf{C}(r, s) \wedge \neg \mathbf{O}(r, s), \\
\mathbf{TPP}(r, s) &\equiv_{def} \mathbf{PP}(r, s) \wedge \exists z[\mathbf{EC}(z, r) \wedge \mathbf{EC}(z, s)], \\
\mathbf{NTPP}(r, s) &\equiv_{def} \mathbf{PP}(r, s) \wedge \neg \exists z[\mathbf{EC}(z, r) \wedge \mathbf{EC}(z, s)], \\
\mathbf{P}^-(r, s) &\equiv_{def} \mathbf{P}(s, r), \\
\mathbf{PP}^-(r, s) &\equiv_{def} \mathbf{PP}(s, r), \\
\mathbf{TPP}^-(r, s) &\equiv_{def} \mathbf{TPP}(s, r), \\
\mathbf{NTPP}^-(r, s) &\equiv_{def} \mathbf{NTPP}(s, r).
\end{aligned} \tag{3.3}$$

3.6.2 Refinement of the RCC for Blocks

Taking into account the peculiarities of the rectangular shape of the block, we can refine RCC8 by introducing additional topological relationships of the higher level of granularity. The new relations are illustrated in Figure 3.4. EC can be further refined by seven corresponding relations, introduced in (3.4). All these relations except EC₃, EC₄, and their inverses are symmetric. DC subsumes eight relations defined in (3.5). All these relations except DC₄, DC₅, and their inverses are symmetric. PO can be split on ten relations specified in (3.6), where all the relations are symmetric except PO₂, PO₃, PO₆, and their inverses. The refinement of TPP is presented in (3.7), where all the eight relations are asymmetric. These 33 specific relations together with NTPP, NTPP⁻, and EQUAL are jointly exhaustive and pairwise disjoint (JEPD). All 36 basic relations enable describing exhaustive set of spatial configurations of blocks merely based on C.

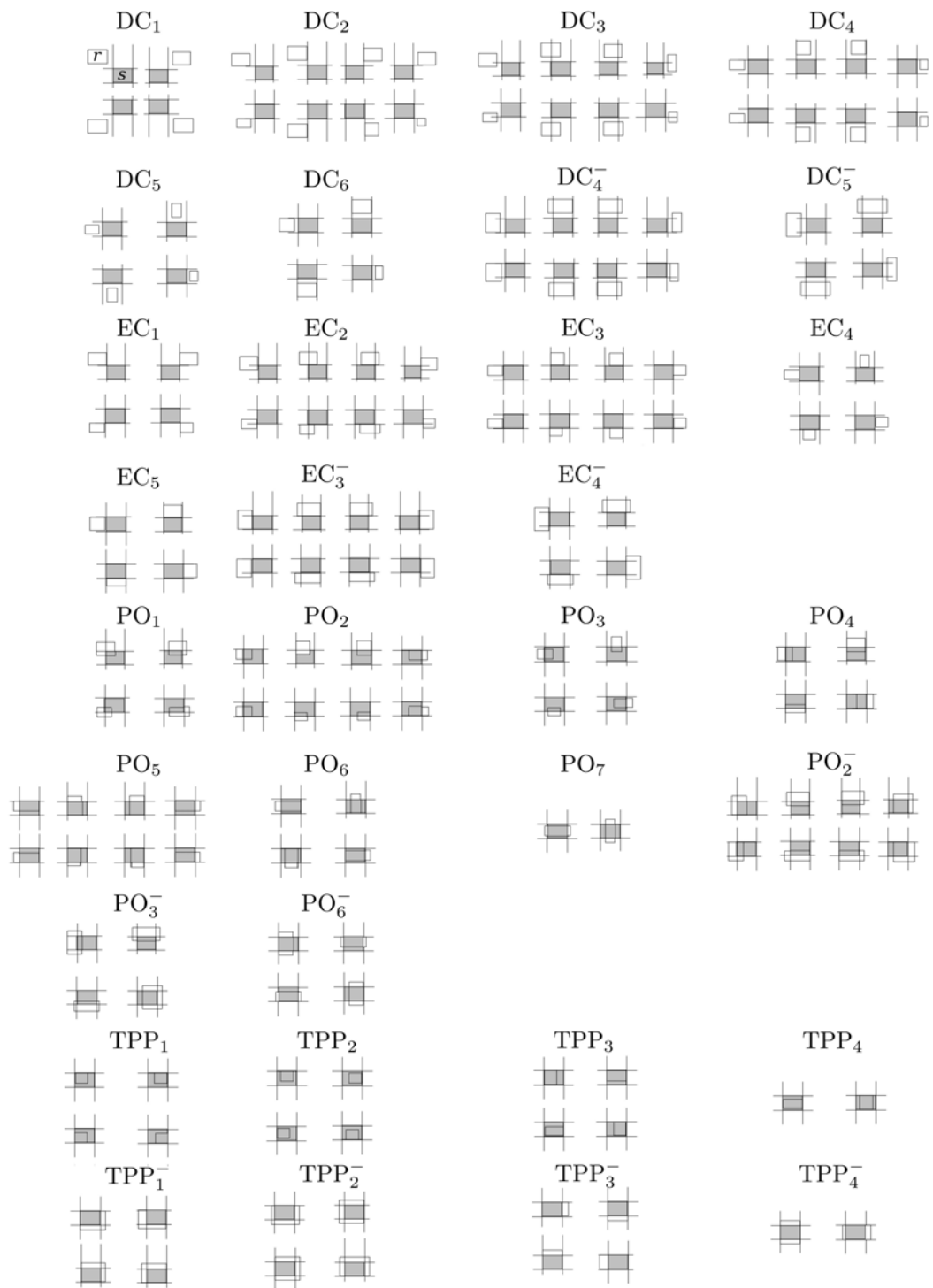


Figure 3.4: Pictorial representation of the additional topological relations between blocks which extend RCC8

$$\begin{aligned}
EC_1(r, s) &\equiv_{def} EC(r, s) \wedge \exists z_1, z_2 [EC(z_1, r) \wedge EC(z_1, s) \\
&\quad \wedge EC(z_2, r) \wedge EC(z_2, s) \wedge EC(z_1, z_2)], \\
EC_2(r, s) &\equiv_{def} EC(r, s) \wedge \forall z_1 \exists z_2 [EC_1(z_2, s) \wedge PO(z_2, r) \\
&\quad \wedge (EC_1(z_1, s) \wedge DC(z_1, z_2) \rightarrow DC(z_1, r))], \\
EC_3(r, s) &\equiv_{def} EC(r, s) \wedge \exists z_1 [EC_1(z_1, s) \wedge EC(z_1, r)] \\
&\quad \wedge \exists z_2 [EC_1(z_2, r) \wedge PO(z_2, s)], \\
EC_4(r, s) &\equiv_{def} EC(r, s) \wedge \forall z [EC_1(z, s) \rightarrow DC(z, r)], \\
EC_5(r, s) &\equiv_{def} EC(r, s) \wedge \exists z_1, z_2 [EC_1(z_1, s) \wedge EC(z_1, r) \\
&\quad \wedge EC_1(z_2, s) \wedge EC(z_2, r) \wedge DC(z_1, z_2)], \\
EC_3^-(r, s) &\equiv_{def} EC_3(s, r), \\
EC_4^-(r, s) &\equiv_{def} EC_4(s, r).
\end{aligned} \tag{3.4}$$

$$\begin{aligned}
DC_1(r, s) &\equiv_{def} DC(r, s) \wedge \forall z [EC_5(z, s) \rightarrow DC(z, r)], \\
DC_2(r, s) &\equiv_{def} DC(r, s) \wedge \exists z [EC_5(z, s) \wedge EC_1(z, r)], \\
DC_3(r, s) &\equiv_{def} DC(r, s) \wedge \exists z [EC_5(z, s) \wedge EC_2(r, z)], \\
DC_4(r, s) &\equiv_{def} DC(r, s) \wedge \exists z [EC_5(z, s) \wedge EC_3(r, z)], \\
DC_5(r, s) &\equiv_{def} DC(r, s) \wedge \exists z [EC_5(z, r) \wedge EC_4(z, s)], \\
DC_6(r, s) &\equiv_{def} DC(r, s) \wedge \exists z [EC_5(z, r) \wedge EC_5(z, s)], \\
DC_4^-(r, s) &\equiv_{def} DC_4(s, r), \\
DC_5^-(r, s) &\equiv_{def} DC_5(s, r).
\end{aligned} \tag{3.5}$$

$$\begin{aligned}
PO_1(r, s) &\equiv_{def} PO(r, s) \wedge \exists z_1 [EC_1(z_1, r) \wedge PO(z_1, s)] \\
&\quad \wedge \exists z_2 [EC_1(z_2, s) \wedge PO(z_2, r)], \\
PO_2(r, s) &\equiv_{def} PO(r, s) \wedge \exists z_1 [EC_1(z_1, s) \wedge EC(z_1, r)] \\
&\quad \wedge \exists z_2 [EC_1(z_2, r) \wedge PO(z_2, s)], \\
PO_3(r, s) &\equiv_{def} PO(r, s) \wedge \exists z_1, z_2 [EC_1(z_1, r) \wedge PO(z_1, s) \\
&\quad \wedge EC_1(z_2, r) \wedge PO(z_2, s) \wedge DC(z_1, z_2)], \\
PO_4(r, s) &\equiv_{def} PO(r, s) \wedge \exists z_1, z_2, z_3 [EC_5(z_1, r) \wedge EC_5(z_2, s) \\
&\quad \wedge EC_5(z_3, z_1) \wedge EC_5(z_3, z_2)], \\
PO_5(r, s) &\equiv_{def} PO(r, s) \wedge \exists z [EC_1(z, r) \wedge EC_1(z, s)], \\
PO_6(r, s) &\equiv_{def} PO(r, s) \wedge \exists z [EC_5(z, r) \wedge EC_4(z, s)], \\
PO_7(r, s) &\equiv_{def} PO(r, s) \wedge \forall z_1 [EC_1(z_1, r) \rightarrow DC(z_1, s)] \\
&\quad \wedge \forall z_2 [EC_1(z_2, s) \rightarrow DC(z_2, r)], \\
PO_2^-(r, s) &\equiv_{def} PO_2(s, r), \\
PO_3^-(r, s) &\equiv_{def} PO_3(s, r), \\
PO_6^-(r, s) &\equiv_{def} PO_6(s, r).
\end{aligned} \tag{3.6}$$

$$\begin{aligned}
\text{TPP}_1(r, s) &\equiv_{def} \text{TPP}(r, s) \wedge \exists z_1, z_2 [\text{EC}_1(z_1, r) \wedge \text{EC}_1(z_1, s) \\
&\quad \wedge \text{EC}_1(z_2, r) \wedge \text{PO}(z_2, s)], \\
\text{TPP}_2(r, s) &\equiv_{def} \text{TPP}(r, s) \wedge \forall z_1 \exists z_2 [\text{EC}_5(z_2, s) \wedge \text{EC}(z_2, r) \\
&\quad \wedge (\text{EC}_5(z_1, s) \wedge \text{DR}(z_1, z_2) \rightarrow \text{DC}(z_1, r))], \\
\text{TPP}_3(r, s) &\equiv_{def} \text{TPP}(r, s) \wedge \exists z_1, z_2 [\text{EC}_1(z_1, r) \wedge \text{EC}_1(z_1, s) \\
&\quad \wedge \text{EC}_1(z_2, r) \wedge \text{EC}_1(z_2, s) \wedge \text{DC}(z_1, z_2)], \\
\text{TPP}_4(r, s) &\equiv_{def} \text{TPP}(r, s) \wedge \forall z_1 [\text{EC}_1(z_1, s) \rightarrow \text{DC}(z_1, r)] \\
&\quad \wedge \forall z_2 [\text{EC}_1(z_2, r) \rightarrow \text{EC}(z_2, s)], \\
\text{TPP}_1^-(r, s) &\equiv_{def} \text{TPP}_1(s, r), \\
\text{TPP}_2^-(r, s) &\equiv_{def} \text{TPP}_2(s, r), \\
\text{TPP}_3^-(r, s) &\equiv_{def} \text{TPP}_3(s, r), \\
\text{TPP}_4^-(r, s) &\equiv_{def} \text{TPP}_4(s, r).
\end{aligned} \tag{3.7}$$

Topological relations also play an important role in Gestalt theory and pertain to different Gestalt laws; we refer interested readers to [143, 291].

3.7 Direction Relations

Direction relation plays an important role in spatial orientation which in turn describes where the objects are placed relative to one another according to the *frame of reference*. Thus orientation of spatial entities is a ternary relationship (with primary and reference objects and frame of reference as parameters), where the *frame of reference* can be specified either by a third object or by a given direction [3, Sec. 4.3.2] (see Section 3.5). Moreover, frame of reference can be extrinsic (external factors impose an orientation on the reference object), intrinsic (the orientation is given by some inherent property of the reference object), or deictic (the orientation is imposed by the point of view from which the reference object is seen) [50, 205]. For the analysis of orientation of the blocks in a web page's layout, it is reasonable to consider directions within the scope of the extrinsic frame of reference. This gives us a possibility to have a contextual factor independent from the particular cases and to be tied to the geometric space defined for the web page canvas rather than concrete elements of a web page's layout.

From the viewpoint of Gestalt theory [143, 291], direction partially refers to the *law of continuity*, *law of closure* and *law of common fate* [151].

For the spatial orientation, we distinguish between *quantitative* and *qualitative directions*. The former refers to the quantitative model, whereas the latter relates to the qualitative one (see Section 3.2).

3.7.1 Quantitative Direction

For the quantitative direction relations, we use abscissa of the web page canvas as a frame of reference and angle to specify direction. This approach has a qualitative counterpart with the

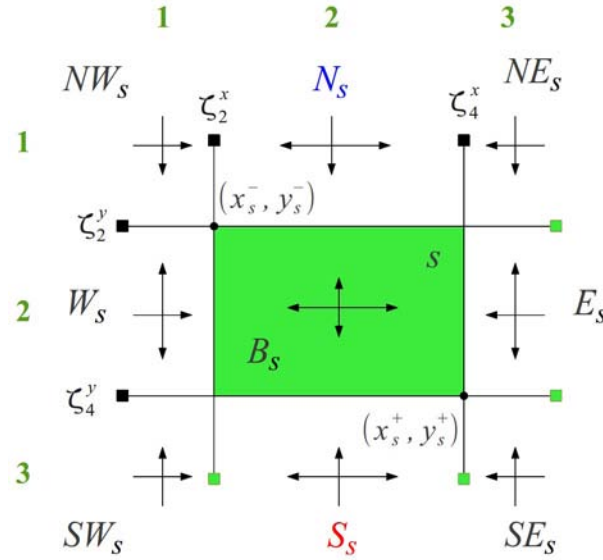


Figure 3.5: Nine main segments (tiles) for the block

cone-based method that divides space at the specific reference point on the set of cones by crossing it with the set of straight lines [89].

Definition 3.8 (Quantitative direction relation). *Quantitative direction relation between primary r and reference s blocks is a binary relation $\gamma^{\text{qnt}}(r, s) \in [0; 360)$ expressed by the clockwise angle in degrees between abscissa and a vector connecting geometric center of the reference block $c(s)$ with geometric center of the primary block $c(r)$.*

The analysis of quantitative directions applied to web pages can be found in Section 3.11.3.

3.7.2 Qualitative Direction

Taking into account peculiarities of the blocks, it makes sense to use *projection-based method* [89] to define qualitative direction relations. Following this method, we define for a block s nine main segments (tiles) such as N_s , NE_s , E_s , SE_s , S_s , SW_s , W_s , NW_s , and B_s which are formed by the orthogonal lines ζ_2^x , ζ_4^x , ζ_2^y , and ζ_4^y bounding the corresponding (reference) block (see Figure 3.5). These tiles are semi-open infinite regions containing points of the corresponding bounding lines ζ_2^x , ζ_4^x , ζ_2^y , and ζ_4^y of the block s .

We distinguish between three types of direction relationships: *O-direction*, *P-direction*, and *center direction relations*. *O-direction* relation is defined based on the types of tails overlapped (see Definition 3.9); *P-direction* requires a containment of the primary block within only one tail of the reference block (see Definition 3.10); a *center direction* specifies the containment of the geometric center of the primary block within one of the tails of the reference block (see Definition 3.11). In their definitions, we use symbols ρ and ρ' as placeholders for directions, for

example, N, NE, or E; and ρ_s and ρ'_s as placeholders for the tiles of the block s , for example, N_s , NE_s , or E_s .

O-direction relation is a type of direction relationship which is defined between primary and reference blocks as follows.

Definition 3.9 (O-direction Relation). *Primary block r has O-direction relation $\rho = N, NE, E, SE, S, SW, W, NW, B$ with the reference block s (denoted as $\rho(r, s)$ or $\gamma_\rho(r, s)$) iff $\tau_O(r, \rho_s)$ where ρ_s is a ρ -th tail of the block s .*

τ_O denotes the RCC8 relation O (see Section 3.6).

Definition 3.10 (P-direction Relation). *Primary block r has P-direction relation $\tilde{\rho}$ with the reference block s (denoted as $\tilde{\rho}(r, s)$ or $\tilde{\gamma}_\rho(r, s)$) where $\rho \in \{N, NE, E, SE, S, SW, W, NW, B\}$ iff $\tau_P(r, \rho_s)$ where ρ_s is a ρ -th tail of the block s .*

τ_P denotes the RCC8 relation P (see Section 3.6).

Definition 3.11 (Center Direction Relation). *Primary block r has center direction relation $\dot{\rho}$ with the reference block s (denoted as $\dot{\rho}(r, s)$ or $\dot{\gamma}_\rho(r, s)$) where $\rho \in \{N, NE, E, SE, S, SW, W, NW, B\}$ iff the following conditions are satisfied:*

- For $\rho' \in \{NE, SE, SW, NW\}$, $\dot{\rho}'(r, s)$ is hold iff $c(r) \in \rho'_s \wedge c(r) \notin \rho_s \setminus \rho'_s$.
- For $\rho' \in \{N, E, S, W\}$, $\dot{\rho}'(r, s)$ is hold iff $c(r) \in \rho'_s \wedge c(r) \notin B_s$.
- $B(r, s)$ is hold iff $c(r) \in B_s$.

($c(r)$ and $c(s)$ denote geometric centers of the blocks r and s respectively.)

O-direction relation is weaker than P-direction and occurs when the primary block overlaps with a certain tail of the reference block. For instance, if a block r is in relation τ_O with the tail N_s of the block s , we say that r is *north of* s and write $N(r, s)$ or $\gamma_N(r, s)$. P-direction is valid if a primary block is equal or contained within the certain tail of the reference block. For instance, if a block r is in relation P with the tail N_s of the block s , we say that r is *strictly north of* s and write $\tilde{N}(r, s)$ or $\tilde{\gamma}_N(r, s)$. Moreover, O-direction relations are jointly exhaustive ($\forall r, s \exists \gamma_i [\gamma_i(r, s)]$) covering all possible spatial configurations of blocks but are not pairwise disjoint ($\exists r [\gamma_i(r, s) \wedge \gamma_j(r, s) \wedge i \neq j]$). In contrast, P-direction relations are pairwise disjoint but are not jointly exhaustive.

For the arbitrary blocks r and s , there can be several O-direction relations defined regarding the tiles being overlapped. Making a *closed world assumption (CWA)*, there are 36 valid combinations of O-direction relations for the blocks, which we denote as $R_{O-dir36}^*$. For example, $\gamma_{NW}(r, s) \wedge \gamma_N(r, s) \wedge \gamma_B(r, s) \wedge \gamma_W(r, s)$ or $\gamma_{NW}(r, s) \wedge \gamma_W(r, s) \wedge \gamma_{NE}(r, s)$. A valid combination of O-direction relations between primary block r and reference block s is formed according to the formula (3.8):

$$\begin{aligned} \gamma_{ij}(r, s) \wedge \gamma_{kl}(r, s) \wedge \min(i, k) \leq p \leq \max(i, k) \\ \wedge \min(j, l) \leq q \leq \max(j, l) \rightarrow \gamma_{pq}(r, s), \end{aligned} \quad (3.8)$$

where $i, j, k, l, p, q \in \mathbb{R}$ denote tails according to Figure 3.5. For instance, γ_{11} corresponds to γ_{NW} , γ_{23} corresponds to γ_E .

It is worth mentioning that in contrast to Rectangular Cardinal Relations (RCR) [216], O-direction and P-direction relations convey necessary minimum of basic orientation relations for the web page's blocks without additional complexity. For example, it is not necessary for us to specify atomic direction relations defined on the unions of tails; it is more suitable to use expression $NW(r, s) \wedge N(r, s)$ instead of defining additional atomic relation $NW : N(r, s)$. This fact is also considered in the development of the corresponding ontological model: the BGM.

In contrast to O-direction and P-direction relations, center direction relations are JEPD. If the geometric center of a block r belongs to the tail N_s and does not belong to any other tail of the reference block s , we say that *the center of the r is north of s* and write $\check{N}(r, s)$ or $\dot{\gamma}_N(r, s)$.

3.8 Distance Relations

Distance and direction are the main types of spatial relationships necessary to specify *positional information*. From the visual perception point of view, distance implicitly reflects the strength of the connection between objects. The less the distance the stronger the relation between the objects. This fact has a direct relation to the *Gestalt* principles [143, 291] and in particular, to the *law of proximity*: we perceive close objects as grouped together [151].

We single out *quantitative* and *qualitative distances* which refer to the quantitative and qualitative models respectively (see Section 3.2).

3.8.1 Quantitative Distance

The Cartesian coordinate system where ordinate is directed from top to bottom (see Definition 3.4) is a frame of reference in Euclidean metric space for the quantitative types of distances. For the points p_1 and p_2 in a metric space, the following axioms hold for a distance function *dist* [50]:

$$dist(p_1, p_1) = 0 \quad (\text{reflexivity}) \quad (3.9)$$

$$dist(p_1, p_2) = dist(p_2, p_1) \quad (\text{symmetry}) \quad (3.10)$$

$$dist(p_1, p_2) + dist(p_2, p_3) \geq dist(p_1, p_3) \quad (\text{triangle inequality}) \quad (3.11)$$

The distance between points p_1 and p_2 , where $p_i = (x_{i1}, x_{i2}, \dots, x_{in})$, of an n -dimensional vector space can be expressed in terms of the Minkowsky L_p -metric:

$$dist_p(p_1, p_2) = \left(\sum_{j=1}^n |x_{1j} - x_{2j}|^p \right)^{1/p}.$$

Conventional Euclidean distance is defined by the L_2 -metric, whereas the city block (or Manhattan) distance is defined by the L_1 -metric.

For the blocks in the BGM, we define three main metrics: 1) distance between borders of blocks (see Definition 3.12), 2) distance between geometric centers (see Definition 3.13), and 3) distance between border projections (see Definition 3.14). These types of relations are illustrated in Figure 3.6.

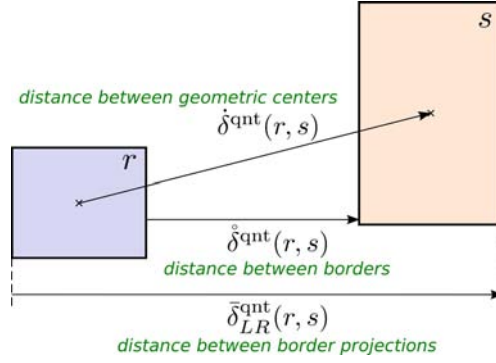


Figure 3.6: Quantitative distances between primary r and reference s blocks

Definition 3.12. *Quantitative distance between borders of the primary r and reference s blocks is a distance $\check{\delta}^{qnt}(r, s)$ with L_2 -metric in two-dimensional Euclidean vector space which equals $\inf(\text{dist}_2(p_r, p_s))$, where $p_r \in r, p_s \in s$.*

According to the definition, for the blocks r and s in RCC8 relation O the distance $\check{\delta}^{qnt}(r, s)$ is equal to 0. $\check{\delta}^{qnt}$ possess a property of reflexivity and symmetry but not a triangle inequality. It is the main type of relations for the blocks in the BGM which allows considering blocks as regions in metric space.

Definition 3.13. *Quantitative distance between geometric centers $c(r)$ and $c(s)$ of primary r and reference s blocks is a distance $\hat{\delta}^{qnt}(r, s)$ with L_2 -metric in two-dimensional vector space.*

In other words, $\hat{\delta}^{qnt}(r, s) = \text{dist}_2\left(\left(\frac{x_r^- + x_r^+}{2}, \frac{y_r^- + y_r^+}{2}\right), \left(\frac{x_s^- + x_s^+}{2}, \frac{y_s^- + y_s^+}{2}\right)\right)$. $\hat{\delta}^{qnt}$ has a property of reflexivity (3.9), symmetry (3.10) and triangle inequality (3.11). This approach can be the case for the complicated non-Manhattan layout (for instance, elements of Scalable Vector Graphics (SVG)).

Definition 3.14. *Quantitative distance between border projections of primary r and reference s blocks is a distance $\bar{\delta}_{ij}^{qnt}(r, s) = p_r - p_s$, where $ij \in \{LL, RR, LR, RL, TT, BB, TB, BT\}$, L corresponds to the left border ($p_k = x_k^-$) of the block $k = r, s$, R to the right border ($p_i = x_i^+$), T to the top ($p_i = y_i^-$), and B to the bottom ($p_i = y_i^+$).*

For instance, quantitative distance between left border projection of the primary block r and right border projection of a reference block s is $\bar{\delta}_{LR}^{qnt}(r, s) = x_r^- - x_s^+$. $\bar{\delta}_{ij}^{qnt}$ possess the property of reflexivity and triangle inequality but not symmetry. It is to be noted that $\bar{\delta}_{ij}^{qnt}(r, s)$ can be negative if the vector connecting certain borders and parallel to the corresponding axis is opposite to this axis. Moreover, there are inversions such as $\bar{\delta}_{LR}^{qnt}(r, s) = -\bar{\delta}_{RL}^{qnt}(s, r)$ and $\bar{\delta}_{TB}^{qnt}(r, s) = -\bar{\delta}_{BT}^{qnt}(s, r)$. This type of spatial relationships allows one to estimate a degree of alignment between blocks and their borders. For instance, for the multicolumn layout right borders of the blocks from the first column has relatively the same distance to the left border of the blocks from the second column.

An evaluation of $\bar{\delta}_{LL}^{\text{qnt}}$ relation on the set of outer and inner blocks of the boxes is presented in Section 3.11.4.

3.8.2 Qualitative Distance

Qualitative expression of different values of distances gives both person and computer the possibility to abstract from the concrete numbers and provides them with concepts which can be used in spatial logical reasoning. For the qualitative distances, there are various levels of granularity and corresponding sets of naming distances (linguistic term). For instance, the first level of granularity distinguishes between “close” and “far.” These two relations divide the plane in two regions centered around the reference object, where the outer region goes to infinity. In a similar way, higher levels of granularity can be defined. For every level of granularity in isotropic space, qualitative distance relations partition the physical space in circular regions of different sizes [50, 120].

For the BGM, we leverage the fourth level of granularity with the terms $R_{\text{dist5}}^* = \{\text{very close}, \text{close}, \text{commensurate}, \text{far}, \text{very far}\}$ (see Section 4.4.2, page 115) with their natural order (e.g. $\text{very close} \prec \text{close} \prec \dots \prec \text{very far}$) according to the *system of qualitative values* presented in Definition 3.3 on page 64 (R_{dist5}^* stands for R^{qnt}). These relations are qualitative representations of the quantitative distance between borders of blocks (see Definition 3.12 on page 78). Based on the definition of the system of qualitative values, we define a function *qualitative distance between blocks* as $\delta^{\text{qnt}} : \mathcal{B} \times \mathcal{B} \rightarrow 2^{R_{\text{dist5}}^*}$, where \mathcal{B} is a set of blocks on a web page canvas; particularly, $\delta^{\text{qnt}}(r, s) = \nu^{\text{qnt}}(\mu(\delta^{\text{qnt}}(r, s)))$, where $\delta^{\text{qnt}}(r, s)$ is a quantitative distance between borders of blocks. $\rho_i \in R_{\text{dist5}}^*$ has a property of symmetry, moreover, all the relations except *very close* are irreflexive. Thus, qualitative distance relations convey a qualitative characteristic of the quantitative data, where the connection between them is realized by the membership functions.

3.9 Alignment Relations

Alignment is a significant supplement to topological, direction and distance relations. This type of spatial relationships plays an important role in laying out objects in physical space [181], documents [2, 129], electronic presentations [108], and graphical user interface (GUI) [144, 180]. Alignment brings out the structure into space of various objects. In the document layout, it can be used to emphasize the border of elements. For instance, it can be used for the main textual content which is distinguished by its left and right alignments and its central position on a document. Alignment is also used for presenting various data structures such as a list, a tree, or a table. According to the Gestalt laws [143, 291], the use of alignment corresponds to the *law of continuity*: oriented parts are integrated into a perceived whole if they are aligned with each other [151].

For the documents, alignment relation is based on the comparison of the projections of geometric objects on abscissa and ordinate. There are two groups of alignment in respect to the projections: horizontal and vertical. Regarding the horizontal alignment of a block r relative to s , we single out relations such as “left aligned” ($\text{LA}(r, s)$ or $\zeta_{\text{LA}}(r, s)$), “right aligned” ($\text{RA}(r, s)$ or $\zeta_{\text{RA}}(r, s)$), “centered vertically” ($\text{CV}(r, s)$ or $\zeta_{\text{CV}}(r, s)$), and “horizontally

not aligned” ($\text{HNA}(r, s)$ or $\zeta_{\text{HNA}}(r, s)$), which are jointly exhaustive for the intervals formed by the vertical projection on abscissa. Regarding the vertical alignment, blocks can be “top aligned” ($\text{TA}(r, s)$ or $\zeta_{\text{TA}}(r, s)$), “bottom aligned” ($\text{BA}(r, s)$ or $\zeta_{\text{BA}}(r, s)$), “centered horizontally” ($\text{CH}(r, s)$ or $\zeta_{\text{CH}}(r, s)$), and “vertically not aligned” ($\text{VNA}(r, s)$ or $\zeta_{\text{VNA}}(r, s)$), which are jointly exhaustive for the intervals formed by the horizontal projection on ordinate. Thus there are 8 jointly exhaustive *basic alignment relations* $R_{\text{align}8} = \{\text{LA}(r, s), \text{RA}(r, s), \text{CV}(r, s), \text{HNA}(r, s), \text{TA}(r, s), \text{BA}(r, s), \text{CH}(r, s), \text{VNA}(r, s)\}$. All the basic relations except HNA and VNA are reflexive, transitive and symmetric. HNA and VNA are symmetric. There are more general relations such as “horizontally aligned” $\text{HA}(r, s)$, “vertically aligned” $\text{VA}(r, s)$, “aligned” $\text{A}(r, s)$, and “not aligned” $\text{NA}(r, s)$ defined in (3.12) which are symmetric; $\text{HA}(r, s)$, $\text{VA}(r, s)$, and $\text{A}(r, s)$ are also reflexive. A and NA are JEPD.

$$\begin{aligned}
\text{HA}(r, s) &\equiv_{\text{def}} \text{LA}(r, s) \vee \text{RA}(r, s) \vee \text{CV}(r, s), \\
\text{VA}(r, s) &\equiv_{\text{def}} \text{TA}(r, s) \vee \text{BA}(r, s) \vee \text{CH}(r, s), \\
\text{A}(r, s) &\equiv_{\text{def}} \text{HA}(r, s) \vee \text{VA}(r, s), \\
\text{NA}(r, s) &\equiv_{\text{def}} \text{HNA}(r, s) \wedge \text{VNA}(r, s).
\end{aligned} \tag{3.12}$$

All the basic alignment relations together with generalized relations can be ordered in the form of a lattice presenting a subsumption hierarchy as it is shown in Figure 3.7. Every block can have more than one basic alignment for every projection (e.g. $\text{LA}(r, s) \wedge \text{RA}(r, s)$). Moreover, the same spatial configuration of the projections can be expressed by different combinations of basic relations, e.g. $\text{LA}(r, s) \wedge \text{RA}(r, s) \leftrightarrow \text{LA}(r, s) \wedge \text{CV}(r, s) \leftrightarrow \text{RA}(r, s) \wedge \text{CV}(r, s)$. To eliminate such cases, we introduce two groups of JEPD alignment relations: for vertical projections $R_{\text{align}5}^h = \{\text{LA}^*(r, s), \text{RA}^*(r, s), \text{LRA}(r, s), \text{CV}^*(r, s), \text{HNA}(r, s)\}$ defined in (3.13) and horizontal projections $R_{\text{align}5}^v = \{\text{TA}^*(r, s), \text{BA}^*(r, s), \text{TBA}(r, s), \text{CH}^*(r, s), \text{VNA}(r, s)\}$ defined in (3.14). The corresponding lattice is presented in Figure 3.8. Relations in $R_{\text{align}5}^h$ and $R_{\text{align}5}^v$ except $\text{HNA}(r, s)$ and $\text{VNA}(r, s)$ are reflexive, symmetric, and transitive. $\text{HNA}(r, s)$ and $\text{VNA}(r, s)$ are symmetric. Leveraging these groups of JEPD alignment relations defined for the blocks’ projections, we introduce a set of 25 JEPD combined alignment relations: $R_{\text{align}25}^* = R_{\text{align}5}^h \times R_{\text{align}5}^v$.

$$\begin{aligned}
\text{LA}^*(r, s) &\equiv_{\text{def}} \text{LA}(r, s) \wedge \neg \text{RA}(r, s), \\
\text{RA}^*(r, s) &\equiv_{\text{def}} \text{RA}(r, s) \wedge \neg \text{LA}(r, s), \\
\text{LRA}(r, s) &\equiv_{\text{def}} \text{LA}(r, s) \wedge \text{RA}(r, s), \\
\text{CV}^*(r, s) &\equiv_{\text{def}} \text{CV}(r, s) \wedge \neg \text{LRA}(r, s).
\end{aligned} \tag{3.13}$$

$$\begin{aligned}
\text{TA}^*(r, s) &\equiv_{\text{def}} \text{TA}(r, s) \wedge \neg \text{BA}(r, s), \\
\text{BA}^*(r, s) &\equiv_{\text{def}} \text{BA}(r, s) \wedge \neg \text{TA}(r, s), \\
\text{TBA}(r, s) &\equiv_{\text{def}} \text{TA}(r, s) \wedge \text{BA}(r, s), \\
\text{CH}^*(r, s) &\equiv_{\text{def}} \text{CH}(r, s) \wedge \neg \text{TBA}(r, s).
\end{aligned} \tag{3.14}$$

The necessary constraint for utilizing $R_{\text{align}25}^*$ is absence of uncertainty in the corresponding qualitative model (see Section 3.2). In contrast to the basic alignment relations, $R_{\text{align}25}^*$ with the open world assumption (OWA) ensures a precise specification of the alignment relation between

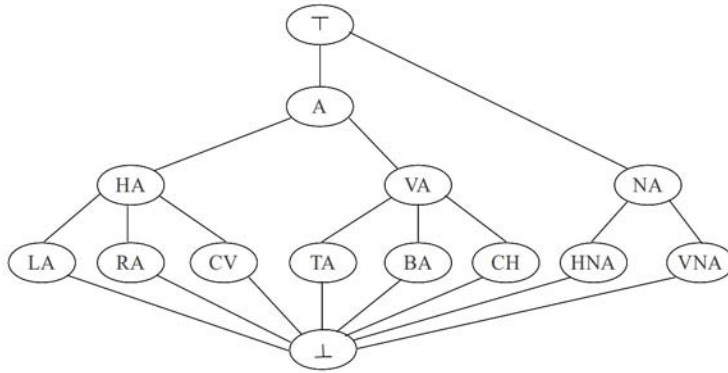


Figure 3.7: A lattice defining the subsumption hierarchy based on the basic alignment relations R_{align8}

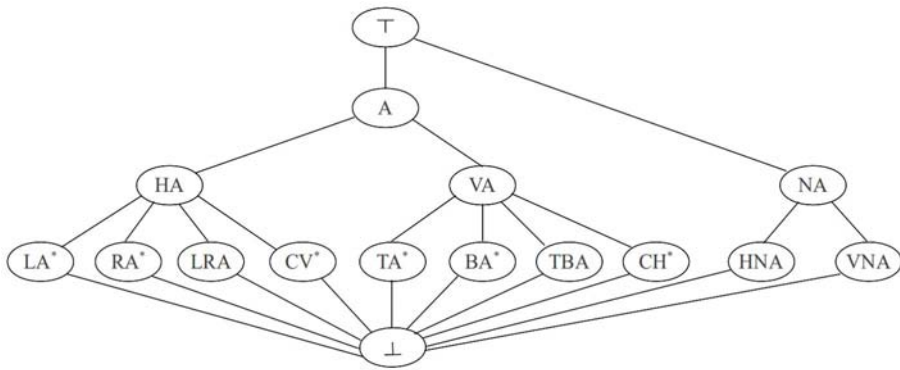


Figure 3.8: A lattice defining the subsumption hierarchy of the dyadic relations based on the alignment relations $R_{align10}$

the blocks. Thus there is always only one relation from R_{align5}^h and R_{align5}^v between any pair of blocks.

An analysis of the alignment relations is presented in Section 3.11.5.

3.10 Interval Relations

In this section, we introduce interval relations (see Section 3.10.1) and express them by means of fuzzy sets and fuzzy relations (see Section 3.10.2) to capture peculiarities of human perception and errors in rendering a web page. Fuzzy interval relations together with centering relation (see Section 3.10.3) are used to express relations in Two-Dimensional Interval Relations with Centering (2DIRC) (see Section 3.10.4). 2DIRC enable us to provide a common framework for expressing spatial relationships such as topological, direction, and alignment, that enables an elimination of corresponding contradictions.

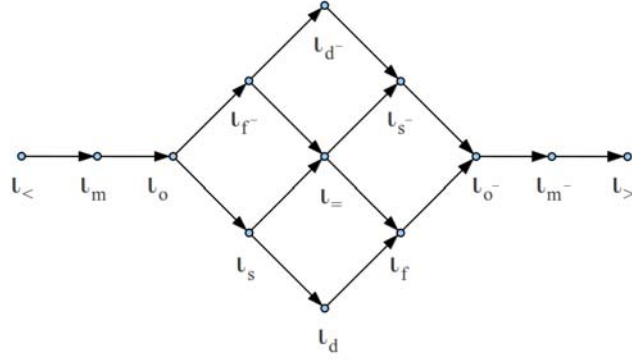


Figure 3.9: Distributive lattice of interval relations [15]

3.10.1 Main Concepts

In [7], Allen defines 13 binary *interval relations* R_{IR} that play an important role in temporal [184] and spatial reasoning [181]. We split the relations in two groups: 1) main, such as *before* $l_{<}$, *meets* l_m , *overlaps* l_o , *starts* l_s , *during* l_d , *finishes* l_f , and *equals* $l_{=}$, and 2) inversions, such as *after* $l_{>}$, *met by* l_{m^-} , *overlapped by* l_{o^-} , *started by* l_{s^-} , *contains* l_{d^-} , and *finished by* l_{f^-} . Table 3.1 gives their detailed description. These relations are JEPD and form a distributive lattice $\langle R_{IR}, \preceq \rangle$ with a partial order \preceq defined on it (see Figure 3.9) [15, 162]. The partial order corresponds to appearance of the interval relations between two intervals during the movement of one relatively to another from left to right, starting with relation $l_{<}$.

3.10.2 Fuzziness in Interval Relations

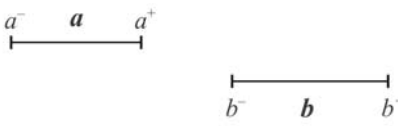
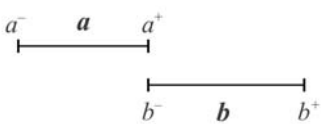
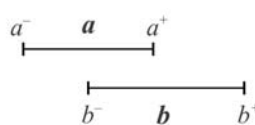
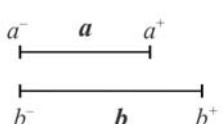
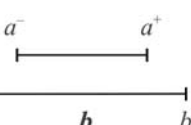
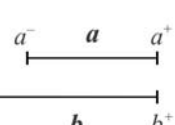
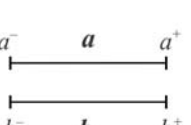
Qualitative information gives a possibility to apply automatic reasoning that tends to be infeasible for a person if the amount of information is big. Thus, definition of qualitative information and its properties should correspond to the peculiarities of human perception. One of these peculiarities is inaccuracy which should be taken into account. Regarding the interval relations, a person cannot definitely say in some cases whether two points are coincide with each other, and as a result sometimes two different interval relations between intervals can be perceived as valid at the same time (for instance, *before* and *meet* relations). Thus, we clarify the concept of equality between endpoints of the intervals.

Setting the origin of coordinates into the necessary endpoint of the reference interval, we specify an equality to the corresponding endpoint of the primary interval in terms of fuzzy sets (a 0-neighborhood) as follows.

Definition 3.15 (0-neighborhood). *A 0-neighborhood E_0 , a fuzzy set of points in the one-dimensional Euclidean space equal to 0, is a pair $\langle x, \mu_0(x) | x \in \mathbb{R} \rangle$, where μ_0 has the following properties:*

1. μ_0 is a function of a Π -like shape, and $\min(\mu_0(x)) = 0$, $\mu_0(0) = 1$;
2. $x_1 \leq x_2 \leq 0 \rightarrow \mu_0(x_1) \leq \mu_0(x_2)$ and $0 \leq x_1 \leq x_2 \rightarrow \mu_0(x_1) \geq \mu_0(x_2)$.

Table 3.1: Interval relations R_{IR} defined by endpoints according to Allen's definition

Name	Denotation	Pictorial example	Equivalent relations on endpoints
<i>before</i>	$\iota_{<}(a, b)$		$a^+ < b^-$
<i>after</i>	$\iota_{>}(b, a)$		
<i>meets</i>	$\iota_m(a, b)$		$a^+ = b^-$
<i>met by</i>	$\iota_{m^-}(b, a)$		
<i>overlaps</i>	$\iota_o(a, b)$		$a^- < b^- \wedge$ $a^+ > b^- \wedge$ $a^+ < b^+$
<i>overlapped by</i>	$\iota_{o^-}(b, a)$		
<i>starts</i>	$\iota_s(a, b)$		$a^- = b^- \wedge$ $a^+ < b^+$
<i>started by</i>	$\iota_{s^-}(b, a)$		
<i>during</i>	$\iota_d(a, b)$		$a^- > b^- \wedge$ $a^+ < b^+$
<i>contains</i>	$\iota_{d^-}(b, a)$		
<i>finishes</i>	$\iota_f(a, b)$		$a^- > b^- \wedge$ $a^- < b^+ \wedge$ $a^+ = b^+$
<i>finished by</i>	$\iota_{f^-}(b, a)$		
<i>equals</i>	$\iota_{=}(a, b)$		$a^- = b^- \wedge$ $a^+ = b^+$

In the following definition, we specify a concept of *system of equality relations* between two endpoints. This system requires the presence of two membership functions ($\mu_0^-(x)$ and $\mu_0^+(x)$) of the 0-neighborhoods (E_0^- and E_0^+ respectively) for the first and second endpoints of the interval. Furthermore, the center $\xi \in (0; 1)$ and the threshold values $\varepsilon^- \in [0; \xi)$ and $\varepsilon^+ \in [0; 1 - \xi)$, which are used to map fuzzy equality relations to a crispy equality relation in terms of ternary logic, should be provided (see Figure 3.10).

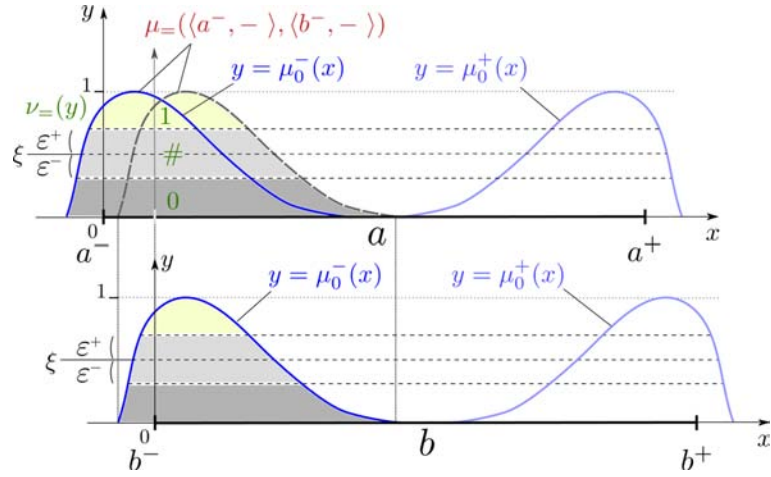


Figure 3.10: Some concepts of the system of equality relations between two endpoints

Definition 3.16. A system of equality relations between two endpoints of intervals is a triple $\langle \mathbb{S}, \mu_{=}, \nu_{=} \rangle$. $\mathbb{S} = \mathbb{R} \times \{+, -\}$ refers to the set of endpoints; the first component determines the set of coordinates, and the second component denotes the type of an interval's endpoint (“-” is the first, and “+” is the second endpoints); membership function $\mu_{=} : \mathbb{S} \times \mathbb{S} \rightarrow [0; 1]$ defines a **fuzzy equality relation** between endpoints. $\mu_{=}(s_1, s_2) = \max(\mu_0^{\pi_2(s_1)}(\pi_1(s_2) - \pi_1(s_1)), \mu_0^{\pi_2(s_2)}(\pi_1(s_1) - \pi_1(s_2)))$, where $\pi_1(s_1)$ and $\pi_1(s_2)$ refer to the first component of s_1 and s_2 respectively, defining their coordinates, whereas $\pi_2(s_1)$ and $\pi_2(s_2)$ refer to the second component of s_1 and s_2 respectively, defining the sign “-” or “+.”

Membership functions $\mu_0^-(x)$ and $\mu_0^+(x)$ belong to the corresponding 0-neighborhoods $E_0^- = \{\langle x, \mu_0^-(x) | x \in \mathbb{R} \rangle\}$ and $E_0^+ = \{\langle x, \mu_0^+(x) | x \in \mathbb{R} \rangle\}$ of the first and second endpoints respectively. $\mu_0^-(x)$ and $\mu_0^+(x)$ have a necessary property $\mu_0^+(x) = \mu_0^-(-x)$, that ensures a symmetry of the membership functions in respect to the geometric center of the interval.

$\nu_{=} : [0, 1] \rightarrow \{0, 1, \#\}$, for the value $\mu_{=}$, determines an equality in terms of ternary logic, where “0” is interpreted as false, “1” is true, and “#” is both.

$$\nu_{=}(x) = \begin{cases} 0, & \text{if } x < \xi - \varepsilon^- \wedge \varepsilon^- > 0 \vee x \leq \xi \wedge \varepsilon^- = 0, \\ \#, & \text{if } (\xi - \varepsilon^- \leq x \leq \xi + \varepsilon^+ \wedge \varepsilon^- > 0) \vee (\xi < x \leq \xi + \varepsilon^+ \wedge \varepsilon^- = 0), \\ 1, & \text{if } x > \xi + \varepsilon^+, \end{cases}$$

where $\xi \in (0; 1)$ defines the required center in the range of $\mu_0^{\pm}(x)$ (thus, preferably $\xi = 0.5$), and $\varepsilon^- \in [0; \xi)$ and $\varepsilon^+ \in [0; 1 - \xi)$ are predefined threshold values.

$\nu_{=}$ has the following properties:

1. There is a partial order defined on the domain of $\nu_{=}$: $0 \prec \# \prec 1$;
2. $y_1 \leq y_2 \rightarrow \nu_{=}(y_1) \preceq \nu_{=}(y_2)$, where $y_1, y_2 \in [0; 1]$;
3. $\exists x \in a[\nu_{=}(\mu_0^-(x)) = \nu_{=}(\mu_0^+(x)) = 0]$, this ensures a discernibility of endpoints.

Thus, the membership functions $\mu_0^\pm(x_1 - x_2)$ defines a closeness of a point x_1 to the corresponding endpoint x_2 of certain intervals. $\nu_=\$ gives a possibility to switch from the fuzzy relation represented by $\mu_=\$ to the crisp relations.

Based on the fuzzy equality relation introduced in Definition 3.16, we specify the following binary predicates:

$$\begin{aligned}
x^s \prec y^t &\equiv_{def} x^s < y^t \wedge \nu_=(\mu_=(\langle x^s, s \rangle, \langle y^t, t \rangle)) = 0, \\
x^s \preceq y^t &\equiv_{def} x^s \leq y^t \vee \nu_=(\mu_=(\langle x^s, s \rangle, \langle y^t, t \rangle)) \neq 0, \\
x^s \approx y^t &\equiv_{def} \nu_=(\mu_=(\langle x^s, s \rangle, \langle y^t, t \rangle)) \neq 0, \\
x^s \succ y^t &\equiv_{def} y^t \prec x^s, \\
x^s \succeq y^t &\equiv_{def} y^t \preceq x^s,
\end{aligned} \tag{3.15}$$

where $s, t = +, -$ and $x^s, y^t \in \mathbb{R}$.

To take into account fuzzy equality relation which reflects the specificity of human perception, we redefine interval relations by means of binary relations introduced in (3.15) as follows:

$$\begin{aligned}
\iota_{<}(a, b) &\equiv_{def} a^+ \prec b^- \\
\iota_{>}(a, b) &\equiv_{def} \iota_{<}(b, a) \\
\iota_m(a, b) &\equiv_{def} a^- \prec b^- \wedge a^+ \approx b^- \\
\iota_{m^-}(a, b) &\equiv_{def} \iota_m(b, a) \\
\iota_o(a, b) &\equiv_{def} a^- \prec b^- \wedge a^+ \succ b^- \wedge a^+ \prec b^+ \\
\iota_{o^-}(a, b) &\equiv_{def} \iota_o(b, a) \\
\iota_s(a, b) &\equiv_{def} a^- \approx b^- \wedge a^+ \prec b^+ \\
\iota_{s^-}(a, b) &\equiv_{def} \iota_s(b, a) \\
\iota_d(a, b) &\equiv_{def} a^- \succ b^- \wedge a^+ \prec b^+ \\
\iota_{d^-}(a, b) &\equiv_{def} \iota_d(b, a) \\
\iota_f(a, b) &\equiv_{def} a^- \succ b^- \wedge a^- \prec b^+ \wedge a^+ \approx b^+ \\
\iota_{f^-}(a, b) &\equiv_{def} \iota_f(b, a) \\
\iota_=(a, b) &\equiv_{def} a^- \approx b^- \wedge a^+ \approx b^+
\end{aligned} \tag{3.16}$$

A definition of the interval relations will be equal to the Allen's definition if we use

$$\mu_0^-(x) = \mu_0^+(x) = \begin{cases} 0, & \text{if } x \neq 0, \\ 1, & \text{if } x = 0. \end{cases}$$

Considering an equality relation between two endpoints as a containment relation within some interval, we define

$$\mu_0^-(x) = \begin{cases} 0, & \text{if } x < \sigma^- \wedge x > \sigma^+, \\ 1, & \text{if } \sigma^- \leq x \leq \sigma^+, \end{cases} \tag{3.17}$$

where $\sigma^- \leq 0$ corresponds to the outer part and $\sigma^+ \geq 0$ to the inner part of an interval. This definition correlates with the definition of Thick Boundary Rectangle Relations (TBRR) introduced in [2].

3.10.3 Centering Relation

Considering the peculiarities of the intervals, we introduce additional interval relation, a centering relation $\iota_c(a, b)$, which refers to $\zeta_{CV}(r, s)$ and $\zeta_{CH}(r, s)$ alignment relations for the blocks r and s (see Section 3.9). $\iota_c(a, b)$ has the following necessary properties:

$$\forall a, b[\iota_c(a, b) \rightarrow \iota_d(a, b) \vee \iota_{d^-}(a, b) \vee \iota_{=} (a, b)], \quad (3.18)$$

$$\forall a, b[\iota_{=} (a, b) \rightarrow \iota_c(a, b)]. \quad (3.19)$$

For the intervals a and b , centering relation $\iota_c(a, b) = \nu_{=}^c(\mu_0^c(|(a^- + a^+)/2 - (b^- + b^+)/2|))$, where $\mu_0^c : [0; +\infty) \rightarrow [0; 1]$ is a membership function of the 0-neighborhood of an interval's geometric center (see Definition 3.15 on page 82).

$\nu_{=}^c : [0, 1] \rightarrow \{0, 1, \#\}$, for the corresponding value $\mu_{=}^c$, determines an equality in terms of ternary logic (similar to Definition 3.16 on page 84), where “0” is interpreted as *false*, “1” is *true*, and “#” is *both*.

$$\nu_{=}^c(x) = \begin{cases} 0, & \text{if } x < \xi^c - \varepsilon^{c-} \wedge \varepsilon^{c-} > 0 \vee x \leq \xi^c \wedge \varepsilon^{c-} = 0, \\ \#, & \text{if } (\xi^c - \varepsilon^{c-} \leq x \leq \xi^c + \varepsilon^{c+} \wedge \varepsilon^{c-} > 0) \vee (\xi^c < x \leq \xi^c + \varepsilon^{c+} \wedge \varepsilon^{c-} = 0), \\ 1, & \text{if } x > \xi^c + \varepsilon^{c+}, \end{cases}$$

where $\xi^c \in (0; 1)$ defines the required center in the range of μ_0^c , and $\varepsilon^{c-} \in [0; \xi^c)$ and $\varepsilon^{c+} \in [0; 1 - \xi^c)$ are predefined threshold values.

$\nu_{=}^c$ has the following properties:

1. There is a partial order defined on the domain of $\nu_{=}^c$: $0 \prec \# \prec 1$;
2. $y_1 \leq y_2 \rightarrow \nu_{=}^c(y_1) \preceq \nu_{=}^c(y_2)$, where $y_1, y_2 \in [0; 1]$;
3. $\forall x[(0 \prec \nu_{=}^c(\mu_0^c(x))) \rightarrow (\nu_{=}^c(\mu_0^-(x)) = \nu_{=}^c(\mu_0^+(x)) = 0)]$, this ensures a discernibility of endpoints from the geometric center.

We denote the set of interval relations together with centering relations as R_{IRC} :

$$R_{IRC} = R_{IR} \cup \{\iota_c\}.$$

3.10.4 Two-Dimensional Interval Relations And Centering

Crisp *Two-Dimensional Interval Relations (2DIR)* (also known as *block relations* [16] or *bidimensional temporal relations* [15]) between rectangular geometric objects are based on the Allen's interval relations [7] for the projections of the blocks on abscissa and ordinate. Crisp 2DIR have 169 JEPD pairs of interval relations (see Figure 3.11). They are used in the automatic document understanding [4, 288], reading order detection [2], and for the spatial reasoning [17]. In Figure 3.12, the *reference block* is depicted; vertical ζ_k^x and horizontal ζ_l^y straight lines define the main qualitatively distinguishable locations for the corner points of a *primary block*. The lines with even i (or j) are fixed whereas the lines with odd i (or j) can be drawn through any

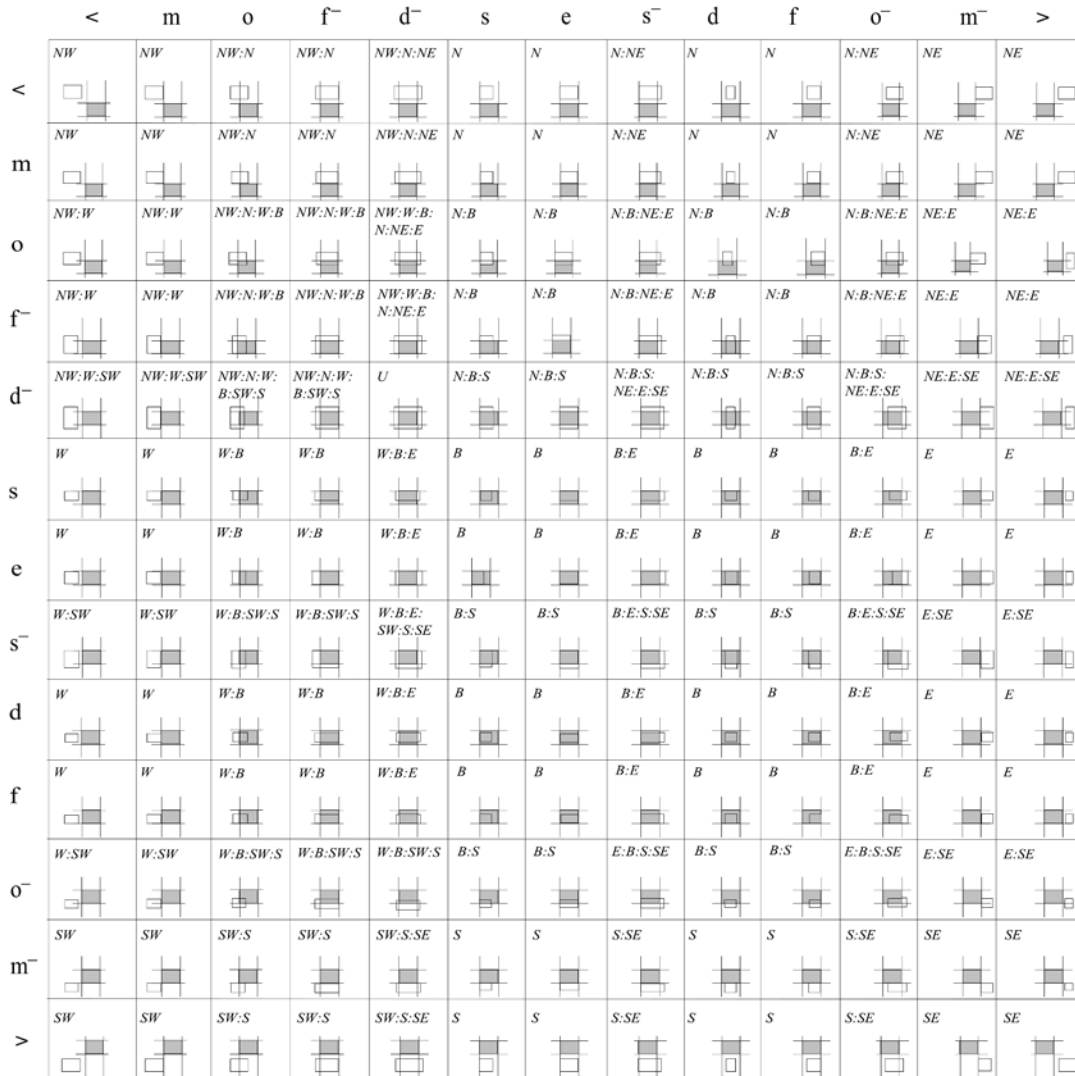


Figure 3.11: Pictorial representation of Two-Dimensional Interval Relations (2DIR) and corresponding Rectangular Cardinal Relations (RCR) [183]

point between neighboring even lines. Depending on the chosen pairs of projection lines ζ_i^x and ζ_j^y for the primary block's corner points, certain 2DIR can be set.

In this thesis, we define 2DIR as follows: $R_{2IR} = \{\langle l_i, l_j \rangle | l_i, l_j \in R_{IR}\}$, taking into account the fuzzy equality relation between endpoints of the blocks' projections (see Definition 3.16 on page 84). This allows uncertainty in the corresponding qualitative model where these relations are defined (presence of more than one relation of the same type between objects, see Section 3.2), that makes R_{2IR} pairwise not disjoint in contrast to the crisp 2DIR.

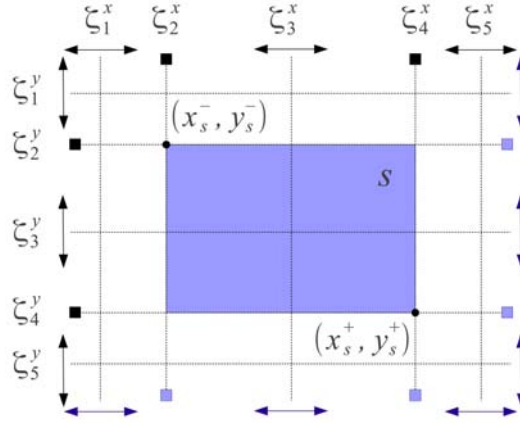


Figure 3.12: Borderlines for the block's corner points

A necessity of introducing fuzziness in the 2DIR is due to the peculiarities of both visual information perception by humans and rendering by the web browser's engine. For instance, in some cases a person cannot give a definite answer regarding an existence of either alignment relation between blocks without a consideration of quantitative information of their positions. On the other hand, according to the Gestalt principles [143, 291], humans tend to pay more attention on those features which correspond to the familiar spatial configuration, for example, be it table or list. Thus, in case when the web author positions elements in a table-like or list-like structure, making some *slight* errors in their quantitative spatial characteristics, the user has the tendency to perceive certain qualitative features correctly. Moreover, the web browser's rendering engine can make some errors that should be taken into account during the analysis of a web page's layout: it can place CSS boxes regardless of the CSS specifications. For example, it happens that the *non-positioned inline-level elements* from the *normal flow* of the same *stacking context* overlap with each other (see Figure 3.13), which is not compliant with [260, Sec. 9.4.2]. Furthermore, an *inline-level element* with borders defined can have the *client rectangles* overlapping with each other (see Section 2.4.5).

Considering various spatial relations between blocks, we discovered the sufficiency of using 2DIR together with centering relations (introduced in Section 3.10.3) for expressing topological (e.g. R_{RCC8} , see Section 3.6), direction (e.g. $R_{O-dir36}^*$, see Section 3.7), and alignment (e.g. $R_{align25}^*$, see Section 3.9) relations [73]. Thus, we introduce Two-Dimensional Interval Relations with Centering (2DIRC) $R_{2IRC} = \{\langle \iota_i, \iota_j \rangle \mid \iota_i, \iota_j \in R_{IRC}\}$ which also have the necessary properties of fuzzy interval relations (see Section 3.10.2) and centering (see Section 3.10.3).

For all possible valid relations R_{RCC8} , $R_{O-dir36}^*$, and $R_{align25}^*$ in case there is no uncertainty, i.e. $\forall x_r^g, y_s^h [\nu = (\mu = (\langle x_r^g, g \rangle, \langle y_s^h, h \rangle)) \neq \#]$, we have $|R_{RCC8, O-dir36, align25}| = |R_{RCC8} \times R_{O-dir36}^* \times R_{align25}^*| = 8 \times 36 \times 25 = 7200$ various jointly exhaustive combinations which are not pairwise disjoint. It is also worth mentioning that there are invalid combinations, such as $\langle \tau_{EQUAL}(r, s), \gamma_E(r, s), \langle \zeta_{LRA}(r, s), \zeta_{TBA}(r, s) \rangle \rangle$ or $\langle \tau_{NTPP}(r, s), \gamma_B(r, s), \langle \zeta_{LA^*}(r, s), \zeta_{CH^*}(r, s) \rangle \rangle$. In contrast, if there is no uncertainty, jointly

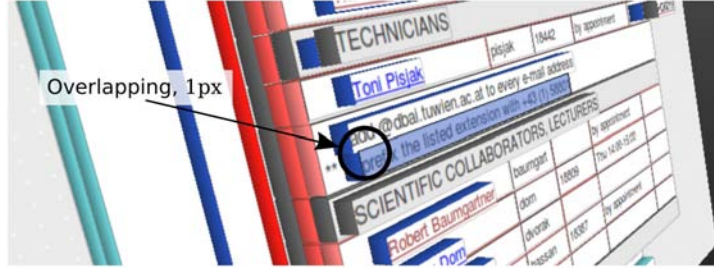


Figure 3.13: Example of overlapping between non-positioned inline-level elements from the normal flow of the same stacking context in Firefox v.19.0.2, <http://www.dbai.tuwien.ac.at/staff/index.html>

exhaustive 2DIRC contains 225 valid relations:

$$\left| \left(\binom{R_{\text{IR}} \setminus \{\iota_{=}, \iota_{\text{d}}, \iota_{\text{d}^{-}}\}}{10} \times \{\neg \iota_{\text{c}}\} \cup \{\iota_{=}\} \times \{\iota_{\text{c}}\} \cup \{\iota_{\text{d}}, \iota_{\text{d}^{-}}\} \times \{\iota_{\text{c}}, \neg \iota_{\text{c}}\} \right)^2 \right| = 225.$$

Relations in 2DIRC are able to express all valid triples in $R_{\text{RCC8}, \text{O-dir36}, \text{align25}}$, that we prove expressing RCC8 R_{RCC8} (3.20), O-directions $R_{\text{O-dir9}}$ (3.21), P-directions $R_{\text{P-dir9}}$ (3.22), and alignment relations R_{align8} (3.23).

$$\begin{aligned}
\tau_{\text{EQUAL}}(r, s) &= \iota_{=}^x(r, s) \wedge \iota_{=}^y(r, s), \\
\tau_{\text{EC}}(r, s) &= (\iota_{\text{m}}^x(r, s) \vee \iota_{\text{m}}^x(s, r)) \wedge \neg \iota_{<}^y(r, s) \wedge \neg \iota_{>}^y(r, s) \\
&\quad \vee (\iota_{\text{m}}^y(r, s) \vee \iota_{\text{m}}^y(s, r)) \wedge \neg \iota_{<}^x(r, s) \wedge \neg \iota_{>}^x(r, s), \\
\tau_{\text{DC}}(r, s) &= \iota_{<}^x(r, s) \vee \iota_{>}^x(r, s) \vee \iota_{<}^y(r, s) \vee \iota_{>}^y(r, s), \\
\tau_{\text{PO}}(r, s) &= (\iota_{\text{o}}^x(r, s) \vee \iota_{\text{o}}^x(s, r)) \\
&\quad \wedge \neg \iota_{<}^y(r, s) \wedge \neg \iota_{\text{m}}^y(r, s) \wedge \neg \iota_{\text{m}}^y(s, r) \wedge \neg \iota_{>}^y(r, s) \\
&\quad \vee (\iota_{\text{o}}^y(r, s) \vee \iota_{\text{o}}^y(s, r)) \\
&\quad \wedge \neg \iota_{<}^x(r, s) \wedge \neg \iota_{\text{m}}^x(r, s) \wedge \neg \iota_{\text{m}}^x(s, r) \wedge \neg \iota_{>}^x(r, s), \\
\tau_{\text{TPP}}(r, s) &= (\iota_{\text{s}}^x(r, s) \vee \iota_{\text{f}}^x(r, s) \vee \iota_{=}^x(r, s)) \\
&\quad \wedge (\iota_{\text{s}}^y(r, s) \vee \iota_{\text{d}}^y(r, s) \vee \iota_{\text{f}}^y(r, s)) \\
&\quad \vee (\iota_{\text{s}}^y(r, s) \vee \iota_{\text{f}}^y(r, s) \vee \iota_{=}^y(r, s)) \\
&\quad \wedge (\iota_{\text{s}}^x(r, s) \vee \iota_{\text{d}}^x(r, s) \vee \iota_{\text{f}}^x(r, s)), \\
\tau_{\text{NTPP}}(r, s) &= \iota_{\text{d}}^x(r, s) \wedge \iota_{\text{d}}^y(r, s).
\end{aligned} \tag{3.20}$$

$$\begin{aligned}
\gamma_N(r, s) &= (\iota_s^x(r, s) \vee \iota_d^x(r, s) \vee \iota_f^x(r, s) \vee \iota_{\underline{}}^x(r, s)) \\
&\quad \wedge (\iota_{<}^y(r, s) \vee \iota_m^y(r, s) \vee \iota_o^y(r, s) \vee \iota_f^y(s, r) \vee \iota_d^y(s, r)), \\
\gamma_{NE}(r, s) &= (\iota_{<}^x(s, r) \vee \iota_m^x(s, r) \vee \iota_o^x(s, r) \vee \iota_s^x(s, r) \vee \iota_d^x(s, r)) \\
&\quad \wedge (\iota_{<}^y(r, s) \vee \iota_m^y(r, s) \vee \iota_o^y(r, s) \vee \iota_f^y(s, r) \vee \iota_d^y(s, r)), \\
\gamma_E(r, s) &= (\iota_{<}^x(s, r) \vee \iota_m^x(s, r) \vee \iota_o^x(s, r) \vee \iota_s^x(s, r) \vee \iota_d^x(s, r)) \\
&\quad \wedge (\iota_s^y(r, s) \vee \iota_d^y(r, s) \vee \iota_f^y(r, s) \vee \iota_{\underline{}}^y(r, s)), \\
\gamma_{SE}(r, s) &= (\iota_{<}^x(s, r) \vee \iota_m^x(s, r) \vee \iota_o^x(s, r) \vee \iota_s^x(s, r) \vee \iota_d^x(s, r)) \\
&\quad \wedge (\iota_{<}^y(s, r) \vee \iota_m^y(s, r) \vee \iota_o^y(s, r) \vee \iota_s^y(s, r) \vee \iota_d^y(s, r)), \\
\gamma_S(r, s) &= (\iota_s^x(r, s) \vee \iota_d^x(r, s) \vee \iota_f^x(r, s) \vee \iota_{\underline{}}^x(r, s)) \\
&\quad \wedge (\iota_{<}^y(s, r) \vee \iota_m^y(s, r) \vee \iota_o^y(s, r) \vee \iota_s^y(s, r) \vee \iota_d^y(s, r)), \\
\gamma_{SW}(r, s) &= (\iota_{<}^x(r, s) \vee \iota_m^x(r, s) \vee \iota_o^x(r, s) \vee \iota_f^x(s, r) \vee \iota_d^x(r, s)) \\
&\quad \wedge (\iota_{<}^y(s, r) \vee \iota_m^y(s, r) \vee \iota_o^y(s, r) \vee \iota_s^y(s, r) \vee \iota_d^y(s, r)), \\
\gamma_W(r, s) &= (\iota_{<}^x(r, s) \vee \iota_m^x(r, s) \vee \iota_o^x(r, s) \vee \iota_f^x(s, r) \vee \iota_d^x(r, s)) \\
&\quad \wedge (\iota_s^y(r, s) \vee \iota_d^y(r, s) \vee \iota_f^y(r, s) \vee \iota_{\underline{}}^y(r, s)), \\
\gamma_{NW}(r, s) &= (\iota_{<}^x(r, s) \vee \iota_m^x(r, s) \vee \iota_o^x(r, s) \vee \iota_f^x(s, r) \vee \iota_d^x(r, s)) \\
&\quad \wedge (\iota_{<}^y(r, s) \vee \iota_m^y(r, s) \vee \iota_o^y(r, s) \vee \iota_f^y(s, r) \vee \iota_d^y(s, r)), \\
\gamma_B(r, s) &= \neg \iota_{<}^x(r, s) \wedge \neg \iota_m^x(r, s) \wedge \neg \iota_{<}^x(s, r) \wedge \neg \iota_m^x(s, r) \\
&\quad \wedge \neg \iota_{<}^y(r, s) \wedge \neg \iota_m^y(r, s) \wedge \neg \iota_{<}^y(s, r) \wedge \neg \iota_m^y(s, r).
\end{aligned} \tag{3.21}$$

$$\begin{aligned}
\tilde{\gamma}_N(r, s) &= (\iota_s^x(r, s) \vee \iota_d^x(r, s) \vee \iota_f^x(r, s) \vee \iota_{\underline{}}^x(r, s)) \\
&\quad \wedge (\iota_{<}^y(r, s) \vee \iota_m^y(r, s)), \\
\tilde{\gamma}_{NE}(r, s) &= (\iota_{<}^x(s, r) \vee \iota_m^x(s, r)) \\
&\quad \wedge (\iota_{<}^y(r, s) \vee \iota_m^y(r, s)), \\
\tilde{\gamma}_E(r, s) &= (\iota_{<}^x(s, r) \vee \iota_m^x(s, r)) \\
&\quad \wedge (\iota_s^y(r, s) \vee \iota_d^y(r, s) \vee \iota_f^y(r, s) \vee \iota_{\underline{}}^y(r, s)), \\
\tilde{\gamma}_{SE}(r, s) &= (\iota_{<}^x(s, r) \vee \iota_m^x(s, r)) \\
&\quad \wedge (\iota_{<}^y(s, r) \vee \iota_m^y(s, r)), \\
\tilde{\gamma}_S(r, s) &= (\iota_s^x(r, s) \vee \iota_d^x(r, s) \vee \iota_f^x(r, s) \vee \iota_{\underline{}}^x(r, s)) \\
&\quad \wedge (\iota_{<}^y(s, r) \vee \iota_m^y(s, r)), \\
\tilde{\gamma}_{SW}(r, s) &= (\iota_{<}^x(r, s) \vee \iota_m^x(r, s)) \\
&\quad \wedge (\iota_{<}^y(s, r) \vee \iota_m^y(s, r)), \\
\tilde{\gamma}_W(r, s) &= (\iota_{<}^x(r, s) \vee \iota_m^x(r, s)) \\
&\quad \wedge (\iota_s^y(r, s) \vee \iota_d^y(r, s) \vee \iota_f^y(r, s) \vee \iota_{\underline{}}^y(r, s)), \\
\tilde{\gamma}_{NW}(r, s) &= (\iota_{<}^x(r, s) \vee \iota_m^x(r, s)) \\
&\quad \wedge (\iota_{<}^y(r, s) \vee \iota_m^y(r, s)), \\
\tilde{\gamma}_B(r, s) &= (\iota_s^x(r, s) \vee \iota_d^x(r, s) \vee \iota_f^x(r, s) \vee \iota_{\underline{}}^x(r, s)) \\
&\quad \wedge (\iota_s^y(r, s) \vee \iota_d^y(r, s) \vee \iota_f^y(r, s) \vee \iota_{\underline{}}^y(r, s)).
\end{aligned} \tag{3.22}$$

$$\begin{aligned}
\zeta_{LA}(r, s) &= \iota_s^x(r, s) \vee \iota_s^x(s, r) \vee \iota_{\underline{=}}^x(r, s), \\
\zeta_{RA}(r, s) &= \iota_f^x(r, s) \vee \iota_f^x(s, r) \vee \iota_{\underline{=}}^x(r, s), \\
\zeta_{CV}(r, s) &= \iota_c^x(r, s), \\
\zeta_{TA}(r, s) &= \iota_s^y(r, s) \vee \iota_s^y(s, r) \vee \iota_{\underline{=}}^y(r, s), \\
\zeta_{BA}(r, s) &= \iota_f^y(r, s) \vee \iota_f^y(s, r) \vee \iota_{\underline{=}}^y(r, s), \\
\zeta_{CH}(r, s) &= \iota_c^y(r, s).
\end{aligned} \tag{3.23}$$

Presented 2DIRC allows us to take into account peculiarities of human perception and web page engine as well as use it as a main framework for topological, direction and alignment relations giving us the possibility to dramatically decrease the amount of various invalid combinations of relations and different ways of expressing equal spatial configurations.

3.11 Analysis of the Spatial Relations and Visibility of CSS Boxes

In this section, we consider a statistical picture of spatial relations, such as topological (see Section 3.6), direction (see Section 3.7), distance (see Section 3.8), and alignment (see Section 3.9), between elements of web pages' layouts [73]. We also investigate the relative amount of visualized elements of the DOM tree (see Section 3.11.6) and relevant statistical characteristics. The analysis was conducted over web pages from the WPPS-HTML-DS1 dataset.

For the WPPS-HTML-DS1 dataset, 650 RDF-based models that contains blocks (outer and inner areas of the CSS boxes corresponding to visualized elements, see Figure 3.2 on page 67) and coordinates of their extreme points were generated. Data was acquired from the Gecko engine (version 1.9.2) with viewport size of 1024×768 pixels. Java along with Jena [10] were used to generate and maintain these experimental geometric models while the SPARQL query language (version 1.1 [282]) was applied for querying the models and providing necessary data for the analysis [100]. The list of SPARQL queries is presented in Appendix A.

3.11.1 WPPS-HTML-DS1 Dataset

A WPPS-HTML-DS1² [72] dataset was built to conduct the analysis of spatial relationships between geometric objects of web pages (in particular, outer and inner blocks of CSS boxes). WPPS-HTML-DS1 contains web pages of CNN and Amazon websites, some subset of the RISE³ corpus which is used for comparing and testing information extraction systems, and a collection of web forums provided by Big Boards⁴.

3.11.2 RCC8

Topological relations provide us with a conceptual description of the spatial configuration of blocks. For instance, location of the block inside another block (PP) means that the former is a

²<http://www.dbai.tuwien.ac.at/staff/fayzrakh/wpps/datasets/WPPS-HTML-DS1.tar.gz>

³<http://www.isi.edu/info-agents/RISE/repository.html>

⁴Web pages of web forums were collected by Wolfgang Holzinger from <http://rankings.big-boards.com/?p=all>.

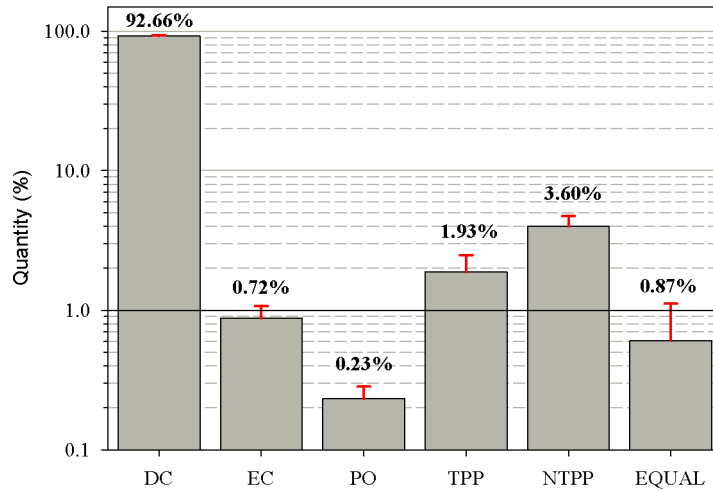


Figure 3.14: RCC8 relations. The ordinate represents average quantity of relations (without symmetric counterparts) of particular type from the set RCC8 defined on web pages from the WPPS-HTML-DS1 dataset, and 95% confidence interval.

part of the latter; if two blocks are externally connected (EC), then an associative relation can be defined between them, e.g. relationship between adjacent cells in a table or between adjacent items in a list.

RCC8 relations are widely used for the connected regions and provide a small set of quite expressive mereotopological relations. In this analysis we consider RCC8 relations, such as DC, EC, PO, TPP, NTPP, and EQUAL to complement Section 3.6.

Evaluation 1. We consider the main *RCC8* relations, such as DC, EC, PO, TPP, NTPP, and EQUAL, between outer blocks of visible CSS boxes (boxes with a nonzero area from the visible parts of page canvases). Symmetric counterparts were omitted, for instance $DC(s, r)$ for $DC(r, s)$, $EQUAL(s, r)$ for $EQUAL(r, s)$. The result of the analysis of the RCC8 topological relations between the blocks from the set of web pages of the WPPS-HTML-DS1 dataset is depicted in Figure 3.14. SPARQL queries used for the analysis can be found in Section A.1.

As we can see, DC relation is significantly predominant (92.66%), giving reason for using additional relations, such as distance and direction, to specify relative position of remote blocks more precisely. NTPP is the second relation by the frequency of appearance (3.60%). For qualifying a relative position of the blocks with NTPP relation, notions such as distance between blocks' borders and centering can be utilized. The former is actively used for modeling a segments' layout of presentations [108]. Presence of TPP relation (1.93%) provides a clue about the existence of alignment between contained and containing blocks, which is a motivation for using the alignment relations supplementary to specify coincident borders. There are 0.87% pairs of blocks which are in equality relation EQUAL with each other. At least one of the blocks in the pair is invisible. This fact gives a possibility for reducing the complexity of the model of web page layout by removing such invisible objects. Blocks with relations TPP and NTPP also

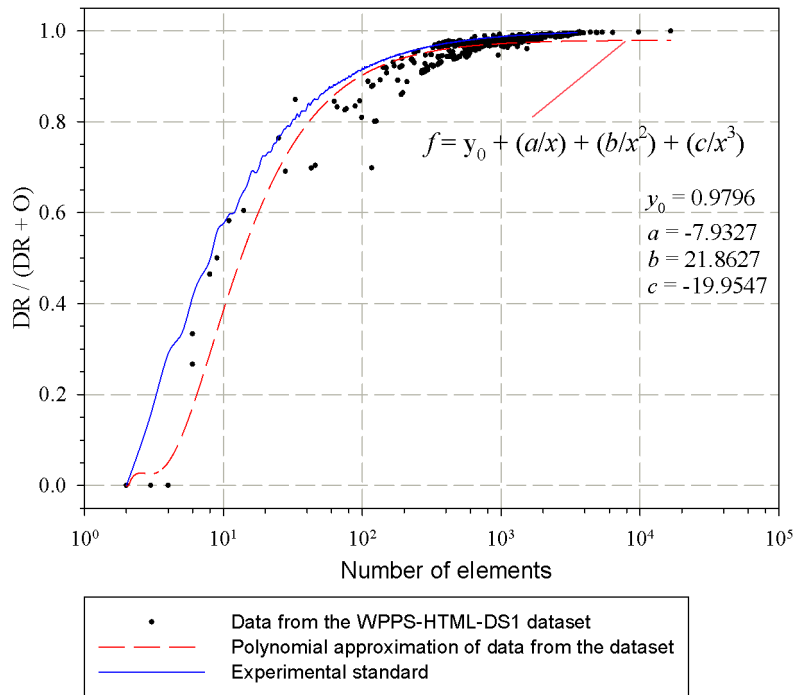


Figure 3.15: Correspondence between number of elements and relative amount of DR relations for web pages from the WPPS-HTML-DS1 dataset and randomly generated trees that serve as a model of a randomly generated web page

can be canceled by the corresponding containing block if it was drawn later [260, Sec. 9.9]. A relative position of the blocks which are in EC relation (0, 72%) can be qualified by the means of direction relation to express the border of contact. Some pairs of blocks are in PO relation (0.23%). Therefore, traditional approaches of document segmentation (e.g. XY-cut [106, 176, 182] and VIPS [40, 41]) cannot be leveraged if overlapped objects should be considered as separate objects. A relative position of the blocks which are in PO relation can be specified with distance relation between their borders, combination of distance and direction, or by the ratio of the intersection area to the area of the intersecting block.

Evaluation 2. This experiment gives us an intuition about average difference between authored web page and randomly generated web page. In this evaluation, we analyzed a relative amount of DR relations (without symmetric counterparts) between visible outer blocks of web pages from the WPPS-HTML-DS1 dataset and randomly generated trees where a relationship between child and parent elements was considered as P. These trees model randomly generated web pages and model relations such as P and DR. Thus, these models neglect PO relations that however do not affect the results of the analysis since we are only interested in DR and O relations.

The result of the analysis is illustrated in Figure 3.15. Polynomial approximation with inverse third order represents averaged functional dependency between number of visualized blocks and relative quantity of DR relations based on the data from the WPPS-HTML-DS1 dataset.

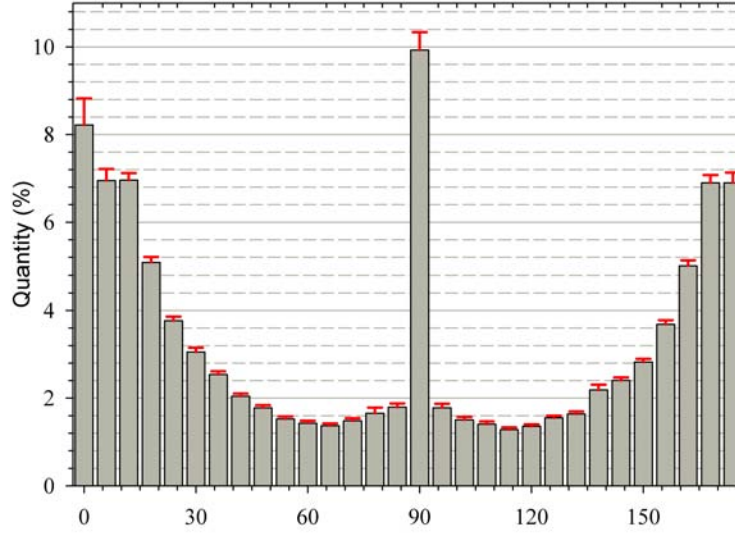


Figure 3.16: Quantitative direction relations. Averaged quantity of directions and 95% confidence within the interval $[-3^\circ; 177^\circ)$ split into groups of the length 6° for the web pages from the WPPS-HTML-DS1 dataset

Experimental standard represents an average relative quantity of elements with DR relation in the random trees. For every number of nodes from interval $[2, 3500]$ 100 generated tree were considered. As we can see, almost all values corresponding to the dataset are below the computed standard that tells us that web authors and authoring tools tend to nest elements that makes segmenting tree more narrow and deeper than average randomly generated tree with the same amount of nodes.

3.11.3 Quantitative Direction

Evaluation. For the quantitative direction relations defined in Section 3.7, we carried an experiment. For the web pages of the WPPS-HTML-DS1 dataset, we computed quantitative directions between outer blocks which are positioned at a distance δ^{qnt} not exceeding 50px from each other. Moreover, we defined 30 groups of quantitative directions within the interval $[-3^\circ; 177^\circ)$, where every group corresponds to the interval of the length 6° . The aggregated data is depicted in Figure 3.16. A SPARQL query used for aggregating specified groups of directions is presented in Section A.2. As was expected most of the directions are either horizontal or vertical. It is due to the fact that most of the web pages follows Manhattan [110] or near-Manhattan layout [116].

This evaluation practically confirms the importance of applying *projection-based method* [89] with cardinal directions introduced in Section 3.7.

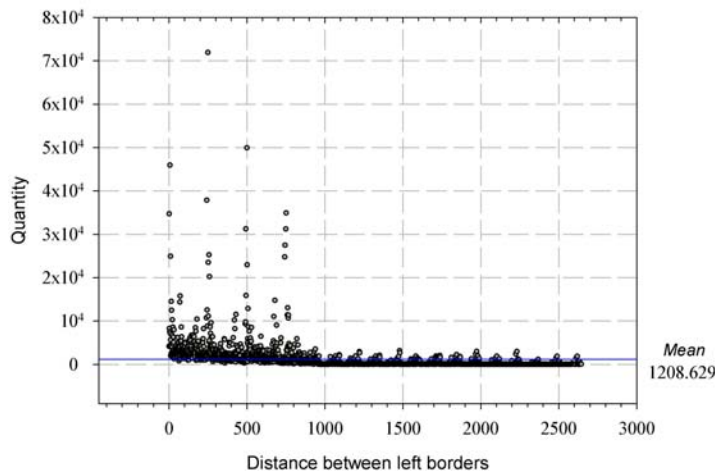


Figure 3.17: Quantitative distance relations. Quantity of various distances between left borders of the blocks for CNN web page <http://edition.cnn.com/> from the WPPs-HTML-DS1 dataset

3.11.4 Quantitative Distance

Considering the quantitative directions between blocks (see Section 3.8), one can find out various regularities which reflect peculiarities of the web page’s layout (some values of distances occur more often). These peculiarities are associated with the presence of various alignments between blocks that is necessary, for instance, for making multicolumn layout, tables and lists.

Evaluation. To confirm an existence of regularities in distances between blocks, we consider CNN web page (<http://edition.cnn.com/>) and positive quantitative distances between left borders both of outer and inner blocks (see Figure 3.17). For the sequence of number of repetitions of various distances between left borders with ascending order, the ninth decile 141.25 times more than the first. This supports the fact that some distances occur much more often than the other. A distance of 7 px from the ninth decile appears 45 924 times and corresponds to the distance between title of the category and list of news titles. The most frequent is distance 250 px, which is met 71 908 times. It corresponds to the distance between left borders of elements of

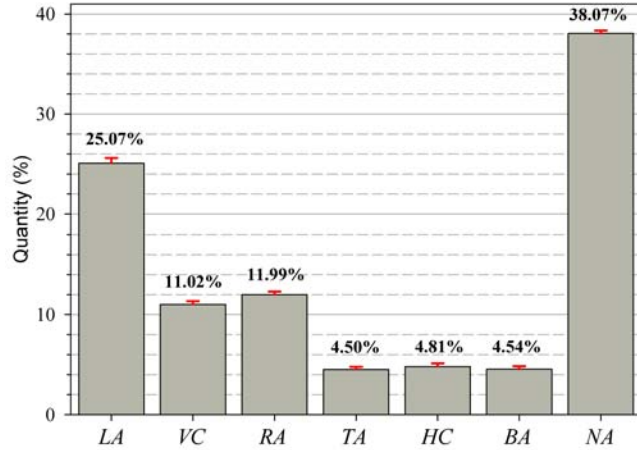


Figure 3.18: Alignment relations. The ordinate represents average quantity of the various alignment relations (without symmetric counterparts) defined on web pages from the WPPS-HTML-DS1 dataset, and 95% confidence interval.

different columns in the multi-column layout of the CNN web page.

Section A.3 presents a SPARQL query used for computing the aggregated values illustrated in Figure 3.17.

3.11.5 Alignment relations

Alignment relations described in Section 3.9 play an important role in web page visual formatting. They are also an important attribute of Manhattan layout [110].

Evaluation. In this experiment, we consider an existence of various alignment relations between elements on a web page canvas. The SPARQL queries which were used for querying the experimental geometric model are presented in Section A.4. As was discovered (see Figure 3.18), more than 60% of blocks are aligned. The left alignment relations (LA = 25.07%) occur more often than any other relations; this is due to the fact that most of the web pages in the WPPS-HTML-DS1 dataset are with languages written from left to right. Horizontal alignment exceeds vertical alignment which is related to the vertical orientation of web pages where enumeration from top to bottom dominates over horizontal enumeration. Among the vertical alignment, all relations occur almost similarly often (4.5% – 4.81%). It is due to the fact that very often blocks are aligned on their top and bottom borders at the same time.

Predominance of alignment relationships reflect an existence of a common practice to follow Manhattan or near-Manhattan layout in web page authoring.

3.11.6 CSS Box Visibility

We define the following concepts:

- A **visible CSS box** is a CSS box, which has nonzero area, computed CSS properties `display≠none` and `visibility=visible`, and overlaps with the first quadrant of the relevant *page* canvas (see Section 3.3).
- An **invisible CSS box** is a CSS box, which has computed CSS properties `display≠none` and `visibility=visible` and does not overlap with the first quadrant of the relevant *page* canvas or has a zero area.
- A **nonempty text node** is a text node of the DOM tree, which is not empty and is not presented by only whitespace symbols.
- A **visible nonempty text node** is a *nonempty text node* which overlaps with the first quadrant of the relevant *page* canvas and its nesting element node is a *visible CSS box*.
- An **invisible nonempty text node** is a *nonempty text node* which does not overlap with the first quadrant of the relevant *page* canvas and its nesting element node is a *visible CSS box*.

We distinguish three types of DOM elements by their visibility from the viewpoint of the CSSOM:

1. **Visible visualized elements**⁵ are *visible CSS boxes* and *visible nonempty text nodes*.
2. **Invisible visualized elements** are *invisible CSS boxes* and *invisible nonempty text nodes*.
3. **Unvisualized elements** are element nodes and *nonempty text nodes* which are not *visible visualized elements* and not *invisible visualized elements*.

Visible visualized elements are the most important elements since they can be perceived by the user.

Evaluation. To evaluate the relation between three types of DOM elements, we considered web pages from the WPPS-HTML-DS1 dataset [72]. The results are depicted in Figures 3.19 and 3.20. As was discovered, among all DOM nodes of the three types presented, 84.15% are visible visualized elements, 9.31% are invisible visualized elements, and 6.54% are unvisualized elements. Therefore, identification and elimination of invisible visualized and unvisualized elements can reduce the amount of objects in the relevant model (i.e. BGM, see Section 4.4.2) in average by 15.85%.

Furthermore, the following statistical characteristics of visible visualized elements in the WPPS-HTML-DS1 dataset were computed: mean 1105.01, standard deviation 1177.87, maximum value 16631, minimum value 2, median 807, first decile 193, ninth decile 2444.8.

3.12 Discussion

In this chapter, we provided an exhaustive analysis of web pages formatting, presenting concepts such as qualitative and quantitative model, introducing a geometric object and its attributes,

⁵We do not single out visualized elements which can be hidden by another element which was rendered later.

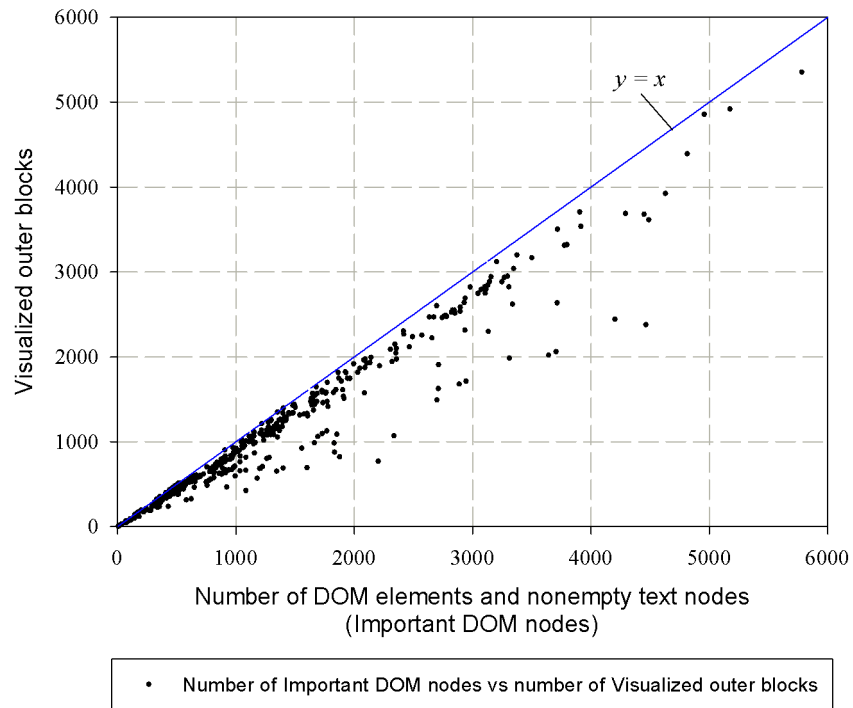


Figure 3.19: Relation between visualized and whole elements of web pages from the WPPS-HTML-DS1 dataset [72]

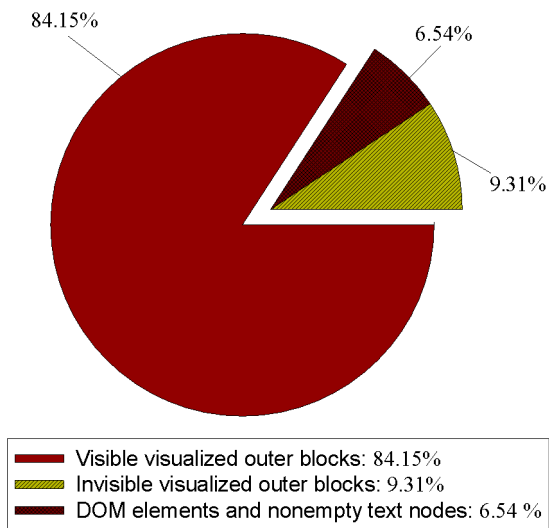


Figure 3.20: Average amount of visible and invisible web page elements from the WPPS-HTML-DS1 dataset [72]

conducting analysis of their spatial relations. Definitions given in this chapter underlie the Block-based Geometric Model (BGM) (see Section 4.4.2) and considered in the process of building the Physical Model (PM) of a web page (see Chapter 5).

The main idea of this chapter is tackling a problem with peculiarities of human perception and web browser engine by means of fuzziness which relate quantitative data to the qualitative information. A Two-Dimensional Interval Relations with Centering (2DIRC) introduced in this chapter enable us to unite relationships such as topological, direction and alignment under one framework. This allows us to eliminate contradictory combinations such as $\langle \tau_{\text{EQUAL}}(r, s), \gamma_{\text{E}}(r, s), \langle \zeta_{\text{LRA}}(r, s), \zeta_{\text{TBA}}(r, s) \rangle \rangle$. If the qualitative model forbids an uncertainty, all triples of relations number of combinations of relations R_{RCC8} , $R_{\text{O-dir36}}^*$, and R_{align25}^* equals 7200, whereas 2DIRC contains 225 relations which are able to express all the valid combinations $|R_{\text{RCC8}} \times R_{\text{O-dir36}}^* \times R_{\text{align25}}^*|$. An evaluation presented in Section 3.11 demonstrates various statistical characteristics regarding their presence on various web page canvas.

The Unified Ontological Model of a Web Page

A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities.

— Tim Berners-Lee, *The Semantic Web*, 2001

This chapter is dedicated to the development of the Unified Ontological Model (UOM) [80] whose purpose is to represent a web page in a form convenient for automatic processing (i.e. web page understanding and web information extraction) and seamless integration with technologies of the Semantic Web. The UOM underlies the Web Page Processing System (WPPS) introduced in Chapter 5 and intended for the development of web page processing methods. It is also used in Chapter 6 for building the Multi-Axial Navigation Model (MANM) to provide effective access to web page content for blind users. In this chapter, we first consider the process of web authoring and reading web pages by the sighted user (see Section 4.1) to demonstrate the importance of visual representation in the understanding of web pages. Such representation is the main modeled object in this thesis. As was discussed in Section 2.4.8, methods based on visual representation are more robust and effective. In Section 4.2, we represent the UOM as a conjunction of the Physical Model (PM) (see Section 4.4) and the Logical Model (LM) (see Section 4.5). It is modeled by means of OWL 2 DL (with RDF-based semantics [281]), and serialized using RDF/XML syntax; some inference rules are additionally applied (e.g. for *irreflexive transitivity*, see Section 4.3). We focus on modeling a web page's layout that can be realized by the Block-based Geometric Model (BGM), which is a sub-model of the PM. The BGM is compliant with the concepts introduced in Chapter 3, as it enables an analysis of various spatial configurations of web page elements. In Section 4.3, we introduce the main properties of relations in ontological models while Section 4.6 concludes the chapter.

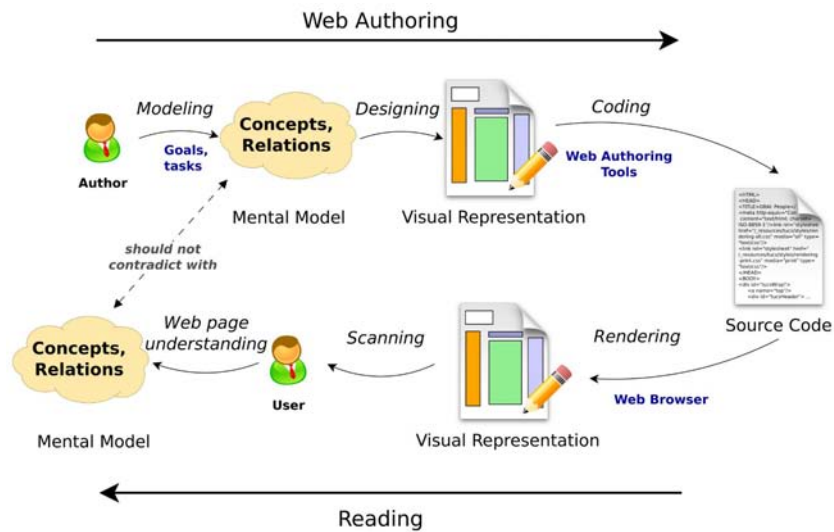


Figure 4.1: Web authoring and reading processes: author's and user's viewpoints

4.1 Web Page Authoring and Visual Representation

Web pages are oriented on sighted users (X/HTML was designed as a visual formatting language). Thus, web pages' visual representation along with the textual content carry essential information about their logical structure and functional roles of elements [111]. To provide access to a web page's content for blind users, we need to analyze processes of authoring and reading web pages by sighted users. These processes are in some sense inverse: If the task of the web page author is to represent all necessary information on a web page's canvas, reflecting functional roles and semantic relationships by means of graphical elements (various shapes, colors, relative position, size, etc.), then the task of the user is to analyze this representation in order to obtain interesting information.

The web authoring process can be represented by the following stages (see Figure 4.1):

1. Modeling: According to her aim, the author forms a *mental model* of what she wants to place on a web page together with the main concepts, features, and relationships between them.

2. Designing: At this stage, the author projects the mental model on a web page's canvas. The model is reflected by the design patterns chosen including page formatting, color palette, and font styles. Some web objects can have conventional representations¹, be it web forum post, navigation menu, or flight search form. This fact is leveraged in different works focused on challenges within the field of Web Page Understanding (WPU), for example, [91, 189, 305] (see also Sections 2.3.2, 2.4.6, and 2.4.7), and in the recent works of the author of this thesis: [121, 145]. Some web page objects can also be mapped into particular data structures and have corresponding representations, such as a table or tree (e.g., hierarchy of sections and subsections in the news article). Peculiarities of information visualization and its perception are studied in Gestalt theory [291] in detail.

¹Examples of design patterns can be found at <http://www.welie.com> and <http://patterntap.com>.

3. Coding: It is a realization of the designed web page by means of X/HTML, CSS, JavaScript and other web languages and web technologies (see Sections 2.4.1 and 2.4.2). Various web authoring tools can be used at this stage (e.g. Dreamweaver, Amaya, HTML-Kit). In contrast to the visual representation, source code has incomparably more various way of representing the same information (see Section 2.4.8). For example, a table can be encoded using HTML tags such as `TABLE`, `TR`, `TD`, or only by `DIV` tags [150].

The process of reading a web page by the sighted user can be represented as follows:

1. Rendering: It is the visualization of a web page performed by the web browser according to the URL specified (by the user). The web browser engine visualizes a web page according to its source code provided in the *coding phase*.

2. Scanning: It includes navigating, reading, scrolling and interacting with the web page leveraging visual cues.

3. Web page understanding: It is a mental process of building the mental model adequately to the scanned web page. It is performed jointly with the *scanning procedure* and involves the analysis of a web page layout, spatial configuration of a web page's objects, color, font, and other visual characteristics engaged in the *designing phase*. A mental model of a user can include an understanding of a web page's logical structure, its web objects (e.g. navigation menu, main content, article, list of products) and textual content. The model should be coherent with the mental model of the web author. It is worth mentioning the vision of a web page publication as a communication process from the author to the user presented in [150].

The spatial configuration of the objects on a web page's canvas and their spatial expansion play an important role in the process of web page understanding. For instance, a horizontal navigation menu is very likely to be presented as a horizontally oriented list of textual elements which are also links on the top of a web page's canvas, and as such, this is how the navigation menu can be identified. There are different types of spatial relationships which form various spatial configurations perceived by the user and recognized as web objects: topology ("overlaps," "part of," "externally connected with;" see Section 3.6), direction (see Section 3.7), relative distance (see Section 3.8), and alignment (see Section 3.9). For example, alignment is used to represent data structures such as lists and tables, to make multicolumn layout, and mold a group of elements, such as vertical list of radio buttons related to the same group of controls. Furthermore, features of web page elements may also influence their perception as a group. For example, elements with the same color or font type may be perceived as a group [291].

In contrast to the sighted user, the blind user cannot reap the benefits of the visual representation and has to use assistive technology to get access to the web page content. Due to the fact that common screen reading technologies (e.g., JAWS, NVDA, Windows-Eyes) do not take into account peculiarities of visual characteristics and do not apply analysis on the visual level, blind users encounter difficulty in navigating and understanding the web pages [21] (see Section 2.2.3). Thus, their mental model is often is not compliant with the mental model both of the author and sighted user. This issue is attentively studied in Chapter 6.

In this chapter, by modeling web page visual representation, we mainly focus on aspects such as web page formatting (layout) and functional roles of elements established by the DOM and CSSOM (e.g., button, link, and table). Once again we shall note that an enhancement of a web page's accessibility is considered in this work from the viewpoint of WPU and Web Information

Extraction (WIE).

4.2 Overview of the Unified Ontological Model for the Web Page Processing

As was discussed in Section 2.4.8, the source code of the web pages and DOM trees are not enough for developing effective methods for web page understanding and web information extraction. Furthermore, these forms of web page representation do not carry substantial information that can be perceived and analyzed by the user. In addition, the conceptual gap between the source code and visual representation is getting even bigger [190] due to the new technologies (e.g. AJAX, Google Web Toolkit, ASP.NET), and new web authoring techniques supported by the web authoring tools (e.g. Dreamweaver, Amaya, HTML-Kit) that have emerged. The need for a thorough analysis of rendered web pages, their layout and visual characteristics for developing more effective and robust methods is highlighted by different researchers [6, 189, 306]. However, there is no model of the rendered web pages suitable for WIE and WPU. Indeed, there are three main groups of standards proposed by W3C for web page representation: 1) source code, specified by X/HTML, 2) DOM tree, which is built based on the source code and JavaScript applied, and 3) CSS, which defines rules for visualizing DOM elements (CSS boxes) on a web page's canvas. CSS does not describe the visual appearance of a web page itself, but rather defines the rules that influence the rendering process. Moreover, current web pages do not provide semantics of the content understandable by computers, which is an essential issue in the development of the Semantic Web and providing an accessible content for both humans and computers.

The goal of developing the Unified Ontological Model (UOM) is to provide visual representation and content semantics in a form convenient for the automatic processing of web pages. Thus, the UOM's purpose is to make web pages accessible and understandable for computers. The UOM is built upon so called *Gestalt Ontology* (see Figure 4.2) for web pages, proposed by W. Holzinger and B. Krüpl-Sypien in 2009 [127]. The Gestalt Ontology is intended for applying the concepts and features as well as principles of visual perception introduced in the Gestalt theory [143, 291] for WIE. The model is also briefly mentioned in [82, 151]. The main concepts of the Gestalt Ontology are *ding* (object, entity), which refers to all perceivable elements and features on a web page (be it image, text, or action, such as page load), and principle of figure and ground, which enables it to make a distinction between elements of the foreground (e.g. text, web form elements, and pop-up window) and elements of the background (e.g. background image and background of the CSS boxes). Gestalt Ontology distinguishes some web form elements, enables specification of the size and the absolute quantitative position of dings as well as alignment and direction relationships between adjacent dings. N. Prangnawarat [197] used this ontology in the analysis of functional elements of the grocery web sites, making the latter more accessible for blind users.

The UOM is a *formalization* of the web page *conceptualization* (i.e., *web page conceptual model*²), suitable for automatic processing. Figure 4.3 illustrates current generic conceptual

²The conceptual model as a sequence of layers of web page representation and understanding is mentioned in [151] for the first time.

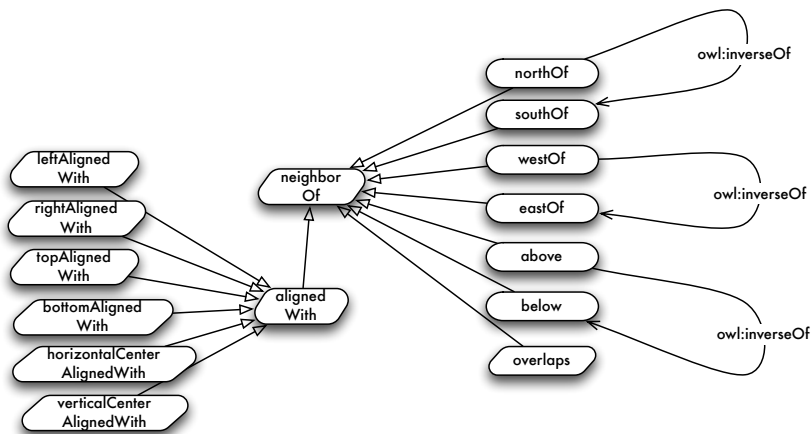
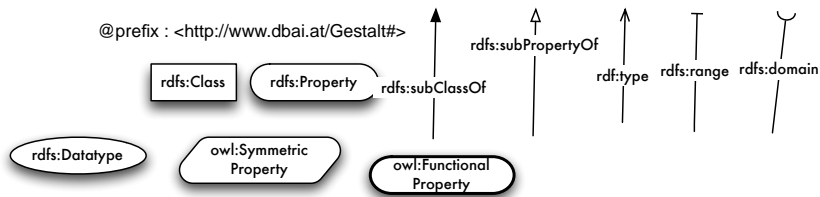
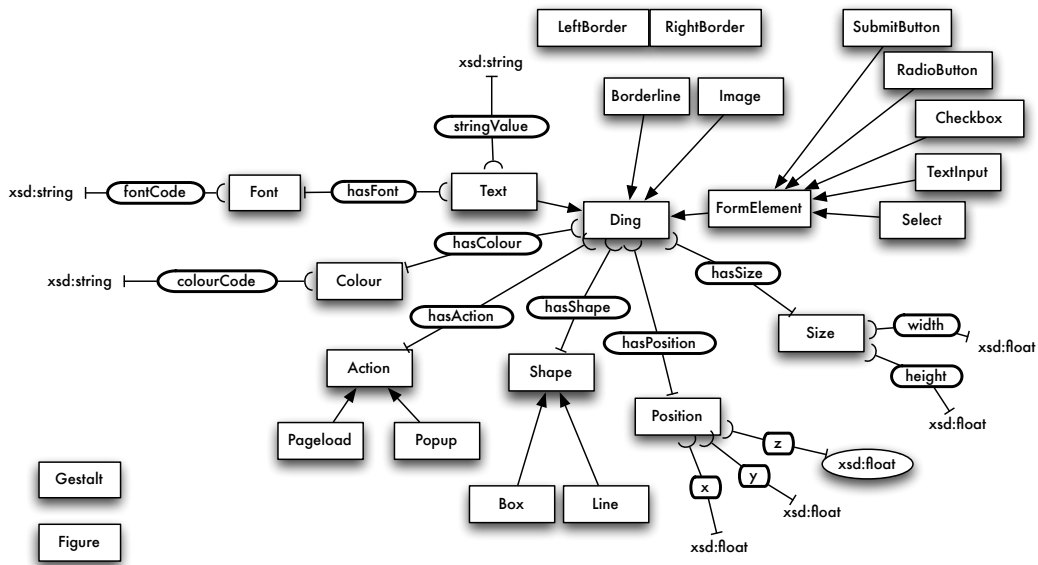


Figure 4.2: Main concepts and features of the Gestalt Ontology for web pages [127]

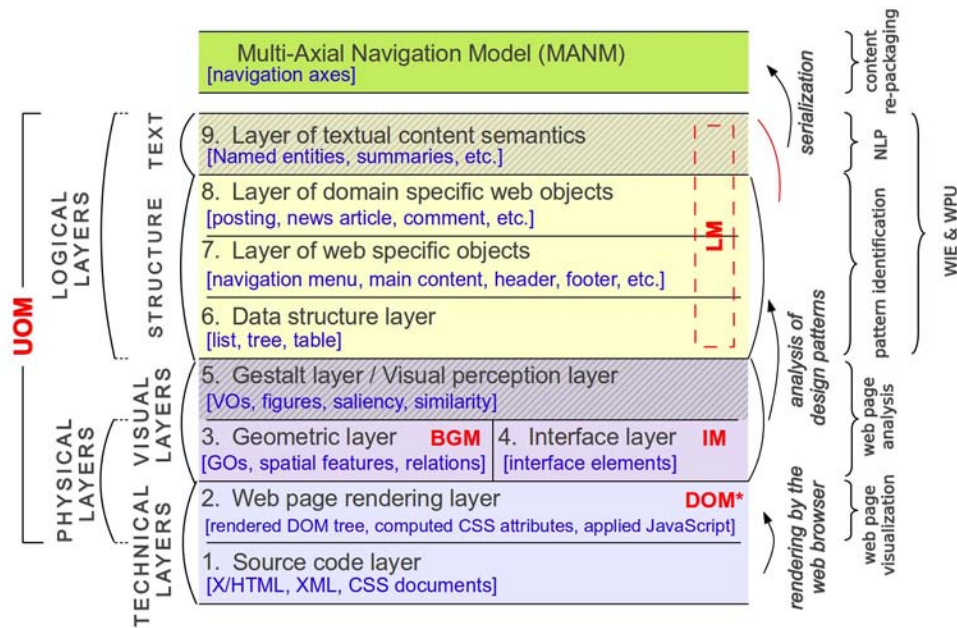


Figure 4.3: Layered conceptual model of the web page

model of the web page. We represent *web page conceptual model* consisting of nine abstraction layers, where every layer has its intention and can be used for specific tasks:

1. Source code layer: This layer is presented by the HTML, XHTML, or XML source code in conjunction with JavaScript code, CSS style sheets, and additional objects such as Java applets, Flash, Silverlight applications, etc. The source code is suitable for transferring in the Web and standardized by the W3C. There are methods which work on this level, such as parsers, textual web browsers, and WIE tools, mainly based on the regular expressions (see Sections 2.4.2 and 2.4.3).

2. Web page rendering layer: This layer consists of the DOM tree generated by the web browser engine, with CSS rules and JavaScript code applied. This layer refers to the web page visualized by the web browser. The existing methods which work on this level make use of DOM tree and computed CSS attributes to localize the visualized DOM elements (see Sections 2.4.4 and 2.4.6). Most of the existing wrappers operate on this level by mainly leveraging XPath.

3. Geometric Layer: This is the result of an analysis of the web page layout and, in particular, spatial characteristics of the DOM elements visualized by the web browser engine. This layer describes a geometry of the web page layout expressing information mainly in a qualitative form (see Section 2.4.7). It plays an important role in the analysis of the spatial configurations of the web page and detecting the objects corresponding to particular spatial pattern. This layer also enables the application of methods from Document Understanding field, such as document segmentation and table recognition, to a certain extent.

4. Interface Layer: This layer provides information regarding the functional role of the elements on a web page, defining functionality of the graphical user interface (GUI). The elements

are web forms elements, such as buttons, text fields, selection lists, check boxes, and radio buttons, as well as links, images, and HTML5 elements, such as `article`, `section`, `canvas`, and `data`.

5. Gestalt layer: This is intended for reflecting the process of human visual perception according to Gestalt theory [143, 291]. This layer allows the possibility to investigate such a process and its use in the problem of web page understanding. A Gestalt layer is based on the geometric and interface layers also taking into account different visual features. The ongoing development of the Gestalt Ontology is primarily oriented on this layer.

6. Data structure layer: It uses data structures to model various logical objects on a web page. For instance, a navigation menu can be represented as a list or tree and article with sections and subsections can be represented as a tree while data visually arranged in the form of a table can be mapped into their logical counterpart, i.e. a table.

7. Layer of web specific objects: Web specific objects correspond to those which are commonly used on web pages regardless of their genre, e.g. navigation menu, header, footer, and main content.

8. Layer of domain specific objects: Domain specific objects refer to a particular genre of a web page. For example, for the web forum genre, these are forum thread, topic, post, and reply. A news web page can have a news article with its title, sections, and comments. This layer represents a set of domain specific ontologies which can be used to model web pages of a certain genre.

9. Layer of textual content semantics: This layer is dedicated to representing meaning of the textual content, its linguistic characteristics and logical (semantic) structure. The layer is formed as the result of applying techniques from the field of Natural Language Processing (NLP).

We call the first two layers *technical layers* (see Figure 4.3), as they represent a web page in forms suitable for transferring through the Web and refer to the inner model of the web page rendered by the web browser engine. Most of the languages and models that correspond to these layers are standardized by W3C. Layers 3–5 are *visual layers* and reflect information acquired due to the analysis of the rendered state of the web page. The analysis of design patterns, spatial configurations, and other visually perceivable features (Gestalt concepts) can be performed on these layers. A variety of existing methods from Document Understanding field can be naturally or with certain adaptation applied for these layers. We name layers 2–4 as *physical layers*; they possess basic information regarding geometric characteristics and interface elements as well as the DOM and the CSSOM. Layers 6–9 are *logical layers* and reflect logical structure and semantics of the web page. They are a result of information extraction or web page understanding together with linguistic analysis of the textual content. Layers 6–8 represent the *structural logical model* of the web page. This *structural logical model* is closely linked with layout, the functional role of interface elements, and their visual features. We associate structural layers with the result of pattern identification. Layer 9 represents semantics of the textual content, which is formed due to the application of methods from the field of NLP. While each layer addresses a different level of abstraction, all of them serve the same purpose in the spirit of the Semantic Web: a more precise and machine-understandable semantic description of a web page. As mentioned in Section 6.2, this conceptualization is leveraged for building the MANM that provides an accessible representation for blind users.

In this work, we formalize the conceptualizations of layers 2–4 and 6 and represent them in the form of the UOM. Thus, we primarily consider the semantics of rendered web pages, established by its layout, structural characteristics of the visualized elements, and their functional roles as well as DOM trees.

Definition 4.1 (Unified Ontological Model). *A Unified Ontological Model (UOM) \mathcal{Y} is a meta-model of web pages, it consists of the Physical Model (PM) \mathcal{P} (see Section 4.4) and the Logical Model (LM) \mathcal{L} (see Section 4.5): $\mathcal{Y} = \langle \mathcal{P}, \mathcal{L} \rangle$.*

Thus, the domain ontology UOM provides the necessary modeling primitives for reflecting physical and logical aspects of web pages. The UOM can be seen as a metamodel for modeling web pages for automatic processing.

Definition 4.2 (Physical Model). *A Physical Model (PM) \mathcal{P} is a set of domain ontologies which correspond to the physical layers of the conceptual representation of the web page; it is a conjunction of the Extended DOM \mathcal{D} (or DOM^* ; see Section 4.4.1), corresponding to the web page rendering layer, the Block-based Geometric Model (BGM) \mathcal{G} (see Section 4.4.2), corresponding to the geometric layer, and the Interface Model \mathcal{I} (see Section 4.4.3) of the interface layer: $\mathcal{P} = \langle \mathcal{D}, \mathcal{G}, \mathcal{I} \rangle$.*

Definition 4.3 (Logical Model). *A Logical Model (LM) is a set of domain ontologies which model certain aspects of logical characteristics of the web page, such as data structures, web specific and domain specific objects, reflecting its layout and functionality, as well as semantics of the textual content (see Section 4.5). The LM corresponds to the layers 6–9.*

Regardless of the definition given, in this thesis, we only formalize data structure layer and consider other logical layers as external ontologies which can be utilized and integrated into the UOM if necessary. Thus, the LM is used in conjunction with PM as an annotation of certain objects of the PM and serves as an interpretation of the physical objects. The understanding of the ontology is compliant with Definition 2.1 on page 48. For modeling the UOM, we use OWL 2 DL (with RDF-based semantics [281]), which provides us with necessary expressive power.

4.3 Properties of Relations

Ontological relations defined on the set of individuals (object properties in OWL, see Section 2.6) can possess different properties.

1. Being defined on the set X a relation $R \subseteq X \times X$ can have properties of the following types:

- *reflexive* $\forall x \in X[\rho(x, x)]$.
- *irreflexive* $\forall x \in X[\neg\rho(x, x)]$.
- *symmetric* $\forall x, y \in X[\rho(x, y) \rightarrow \rho(y, x)]$.
- *antisymmetric* $\forall x, y \in X[\rho(x, y) \wedge x \neq y \rightarrow \neg\rho(y, x)]$

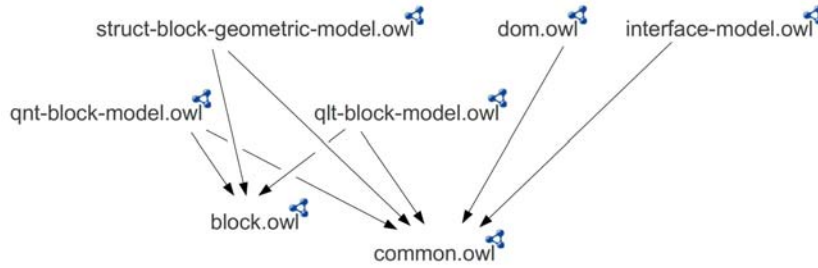


Figure 4.4: Ontology files of the PM [80]

- *asymmetric* $\forall x, y \in X [\rho(x, y) \rightarrow \neg\rho(y, x)]$; this property is stronger than antisymmetry.
- *transitive* $\forall x, y, z \in X [\rho(x, y) \wedge \rho(y, z) \rightarrow \rho(x, z)]$.
- *intransitive* $\neg\forall x, y, z \in X [\rho(x, y) \wedge \rho(y, z) \rightarrow \rho(x, z)]$.
- *antitransitive* $\forall x, y, z \in X [\rho(x, y) \wedge \rho(y, z) \rightarrow \neg\rho(x, z)]$; this property is stronger than intransitivity.

Additionally, we introduce *irreflexive transitivity* which is defined as $\forall x, y, z \in X [\rho(x, y) \wedge \rho(y, z) \wedge x \neq z \rightarrow \rho(x, z)]$.

2. Being defined on the distinct sets X and Y ($X \neq Y$) a relation (*correspondence*) $R \subseteq X \times Y$ can have properties of the following types:

- *functional* $\forall x \in X, y, z \in Y [\rho(x, y) \wedge \rho(x, z) \rightarrow y = z]$.
- *injective (inverse functional)* $\forall x, z \in X, y \in Y [\rho(x, y) \wedge \rho(z, y) \rightarrow x = z]$.

For object properties, assuming an incompleteness in the ontology, we do not consider relations' properties such as *total* and correspondences' totality properties, such as *left-total* and *surjective (right-total)*.

4.4 The Physical Model of a Web Page

The Physical Model (PM) \mathcal{P} (see Definition 4.2 on page 108) is an ontological model which corresponds to the physical layers of the web page's conceptualization introduced in Section 4.2 and consists of the Block-based Geometric Model (BGM) (see Section 4.4.2), Interface Model (see Section 4.4.3) and Extended DOM (see Section 4.4.1). Files with schemata of the sub-models of the PM and their dependencies are depicted in Figure 4.4. Thus, the BGM is mainly represented by `struct-block-geometric-model.owl`, `qnt-block-model.owl`, `qlt-block-model.owl`, and `block.owl`. The Interface Model is mainly based on `interface-model.owl`, and the Extended DOM is defined in `dom.owl`. An instance of the PM is a result of the analysis of a web page's DOM trees and its CSS Object Models (CS-SOMs) (see Section 5.1) which correspond to the technical layers of the web page's conceptual model.

The PM is realized in OWL 2 DL for the seamless integration with the Semantic Web and serialized by means of RDF/XML syntax [277]. Also, it is important to note that OWL 2 has some restrictions, for instance, it is only possible to set constraints for the object properties such as functional, inverse functional (injective), transitive, symmetric, asymmetric, reflexive, and irreflexive (see Section 4.3). Furthermore, transitivity cannot be used together with asymmetry and irreflexivity, and it is forbidden to define a transitivity together with disjointness between properties. The PM is optimized for needs of the web page processing, and due to this fact we distinguish between *intrinsic properties* («I»), which the relation naturally has and properties introduced in the ontology («O»). For instance, we omit the reflexivity, and use irreflexive transitivity in stead of transitivity to avoid reflexivity in the ontology. The irreflexive transitivity is realized by applying inference rule («R»). Due to the fact that the information necessary for particular tasks should be generated, we also try to avoid use of disjointness and cardinality in the ontologies. Ontologies of different sub-models of the PM are represented by means of Class Diagrams. Antisymmetric and intransitive properties are sometimes omitted in the diagrams when their value is obvious; for instance, if an object property is asymmetric then it is also antisymmetric.

Furthermore, in the DOM* and BGM, we allow realization of the *one-to-many* relations both as a set of assertions in the form of RDF triples (e.g. `VisualizedElement hasClientRectangle Box`, see Figure 4.8 on page 114) and via RDF containers, such as sequence (e.g. `VisualizedElement hasClientRectangles RDF:Seq`, see Figure 4.8). RDF sequence gives a possibility to take into account sequential order of elements which was formed during the model generation. In particular, when using sequences, *boxes* (i.e. CSS client rectangles) in a *visualized element* of the PM are ordered according to the sequence of CSS client rectangles in corresponding CSS box, and the sequence of *visualized elements* on a *page* of the PM is ordered according to the depth-first traversal over the DOM tree.

The *object* is the most generic type of the element in the PM which subsumes all others, such as `Node` of the DOM*, `Block` of the BGM, and `WPBIElement` of the Interface Model.

4.4.1 Extended DOM

The Extended DOM \mathcal{D} (or DOM*) conveys necessary vocabulary to describe the DOM trees of a web page in the form of one consistent model. It also enables representing computed CSS attributes (datatype property `hasComputedCssCode`). DOM* corresponds to the *SHIF* Description Logics (DL) and introduces all the node elements (see Figure 4.5) and necessary relations realized as data (see Figures 4.6) and object properties (4.7) according to the W3C specifications [247]. In contrast to the DOM, the DOM* models the whole web page with its *frame hierarchy* as one single DOM* tree, where a *document node* (`Document`) of the *frame* is connected with its parent frame via `descendantOf`, `childOf` and their inverses (`childOf` relates instance of `Document` to the corresponding framing element). Based on the DOM specifications [247], we define *traversal* and *non-traversal nodes*, where the former represents vertexes of the DOM* tree connected by the `parentOf` relation. The non-traversal nodes do not represent vertexes of the DOM* tree and refer to the corresponding traversal nodes, for instance, attribute nodes correspond to the elements, and notation nodes correspond to the document type nodes [244, Sec. 1.2]. According to [240, Sec. 1.1.1], every node type can have certain

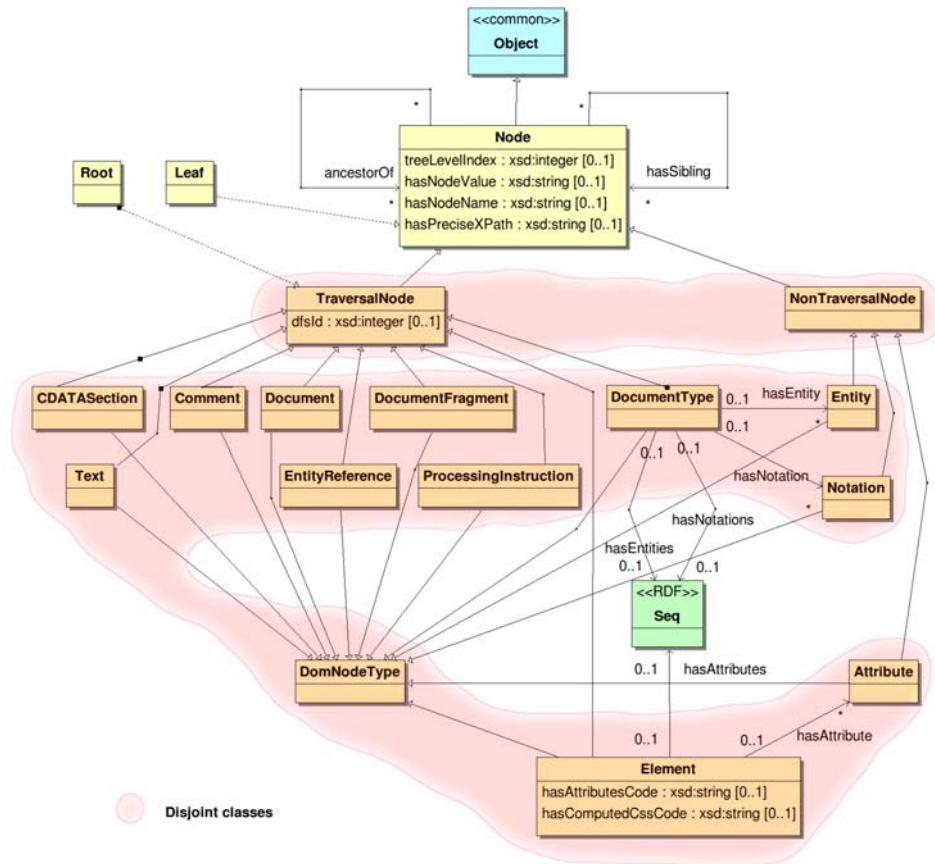


Figure 4.5: Classes of the Extended DOM

set of node types as children. For example, Document can have ProcessingInstruction, Comment, DocumentType, and at most one Element. This fact is properly reflected in the ontology (e.g. $\text{Document} \sqsubseteq \forall \text{parentOf}.\text{ProcessingInstruction} \sqcup \forall \text{parentOf}.\text{Comment} \sqcup \forall \text{parentOf}.\text{DocumentType} \sqcup (\leq 1 \text{parentOf}.\text{Element})$).

Additionally, we define a root element of the DOM* tree as follows: $\text{Root} \sqsubseteq \text{Document} \sqcap \neg \exists \text{descendantOf}.\text{TraversalNode}$; and leaf node $\text{Leaf} \sqsubseteq \text{Node} \sqcap \neg \exists \text{ancestorOf}.\text{Node}$. Several roots are allowed in the instance of the DOM*; it can happen, for instance, when the DOM tree of the parent frame has not been materialized in the ontology while the DOM trees of its child frames have been materialized. The subsumption is used instead of the equality to void inconsistency in case more assertions are added later into the ontology during the process of web page analysis. Roots and leafs can be identified after a completion of the model generation process. By definition, a leaf concept subsumes non-traversal nodes such as CDATASection, Comment, DocumentType, Notation, ProcessingInstruction, Text and thus restricts their use. Every node is identified by its XPath (hasPreciseXPath), and additionally every traversal node is specified by its sequential number (dfsId) in the depth-first traversal over a tree of the

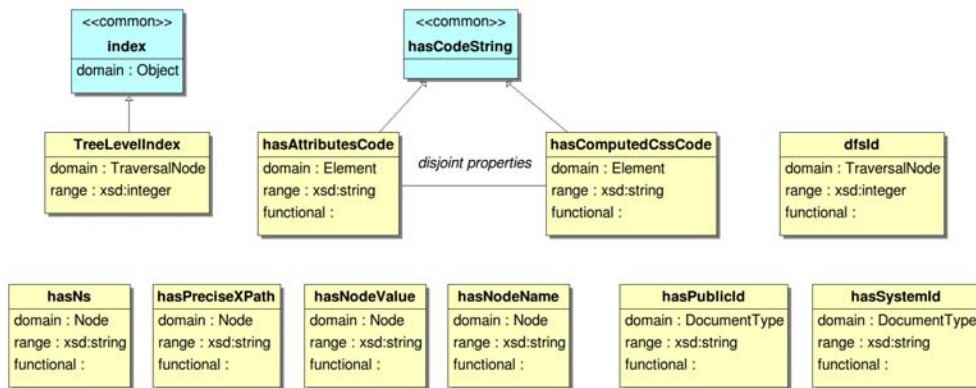


Figure 4.6: Datatype properties of the Extended DOM

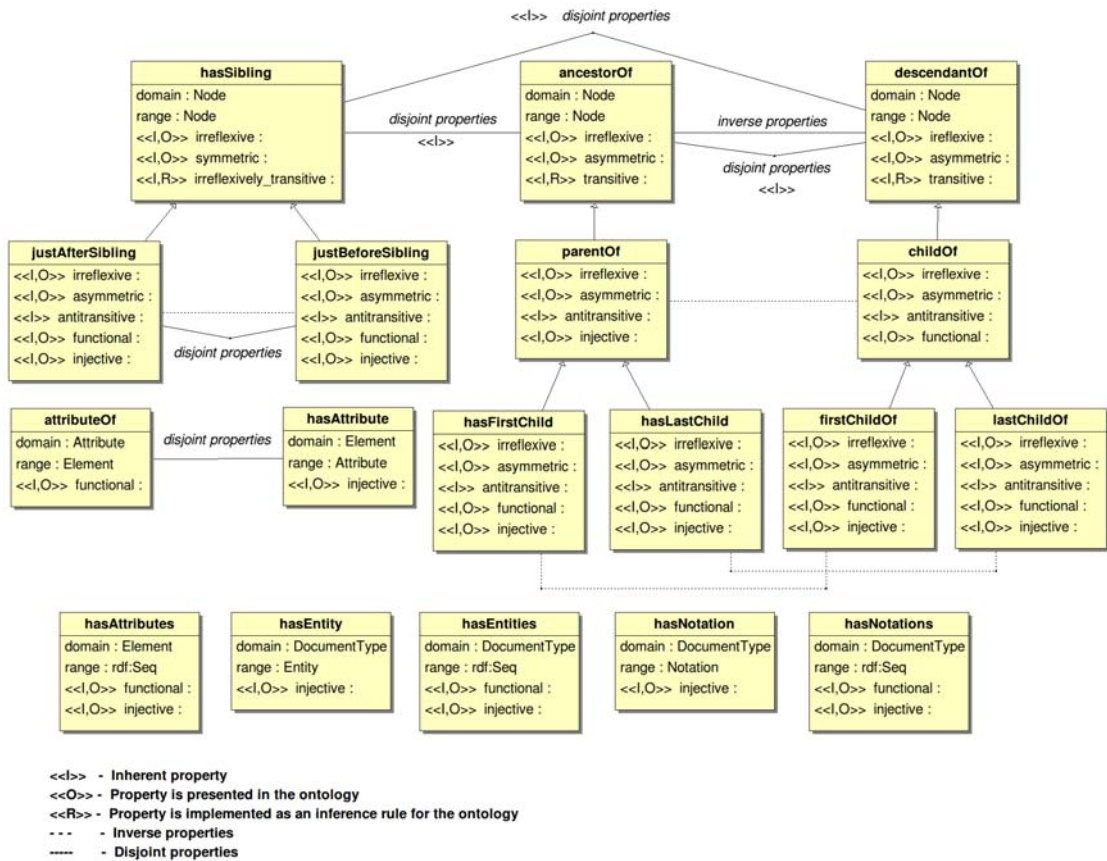


Figure 4.7: Object properties of the Extended DOM

DOM trees of a web page.

A disjointness of the concepts is illustrated in Figure 4.5.

4.4.2 Block-based Geometric Model

The Block-based Geometric Model (BGM) [73], which is the main sub-model of the PM, describes the layout and geometric structure of a web page, and it is based on the work presented in Chapter 3. It plays an important role in the analysis of various spatial configurations of objects for web data extraction and web page understanding. The BGM comprises three main sub-models: the Structural Block-based Geometric Model (StrBGM), Quantitative Block-based Geometric Model (QntBGM), and Qualitative Block-based Geometric Model (QltBGM).

Structural BGM

The Structural Block-based Geometric Model (StrBGM) represents the main geometric and constituent objects of the web page (see Figures 4.8 and 4.9). Some of these objects are described in Section 3.4. The StrBGM corresponds to $\mathcal{ALCLFH}(D)$ DL. The most abstract object in this model is *block* that conforms with Definition 3.5 on page 67. There are two main types of blocks: *basic block* and *composite block*. A *basic block* corresponds to a rectangular area on the web page canvas (see Definition 3.4 on page 65) and can relate to some particular visual element (e.g. a CSS box or a set of CSS boxes). If the visual element cannot be represented by a block (e.g. in case of rotated CSS box, or some non-rectangular glyph of an image), its outline can be considered, which serves as its substitution. The *composite block* corresponds to a rectangular area on the web page canvas and contains either one or several basic or composite blocks.

There are six main types of a composite block in the current version of the BGM: *document block*, *page block*, *viewport block*, *visualized element*, *box*, and *bounding block* (see Figure 4.8). A *document block* represents a whole web page rendered by the web browser's engine together with its frames—a *document*. A document is formed by the set of X/HTML or XML files connected with each other by means of their inclusion (e.g. by elements with the names `FRAME`, or `IFRAME`, or `OBJECT`). A *page block* corresponds to a single rendered X/HTML or XML file of a document—*page*—and relates to the document element of the DOM*. A *page block* contains all the visualized elements of the corresponding DOM tree through the `containsBlock` relation. A set of pages make a *hierarchy of pages* by means of `hasChildPage` relation. (A page has the counterpart `Window` in the Browser Object Model (BOM) [136]). A *viewport block* represents a viewport of a web browser [260, Sec. 9] (see Section 2.4.5) as well as a viewport corresponding to the embedded web page (frame). A *viewport block* uses `containsBlock` relations for the visualized DOM tree elements that topologically are inside or equal (P). The *visualized element* corresponds to the visualized DOM element and is compliant with *box model* and *visual formatting model* specified by the W3C (see Section 2.4.5) [260]. A visualized element consists of the *client rectangles* which are part of the CSS *line box* [260, Sec. 9.4.2], whereas every client rectangle is a *box* (see Section 3.4). As a composite block, a box consists of components such as *outer* and *inner blocks*, where the latter is drawn on top of the former. A *bounding block* is used as a container to wrap other blocks.

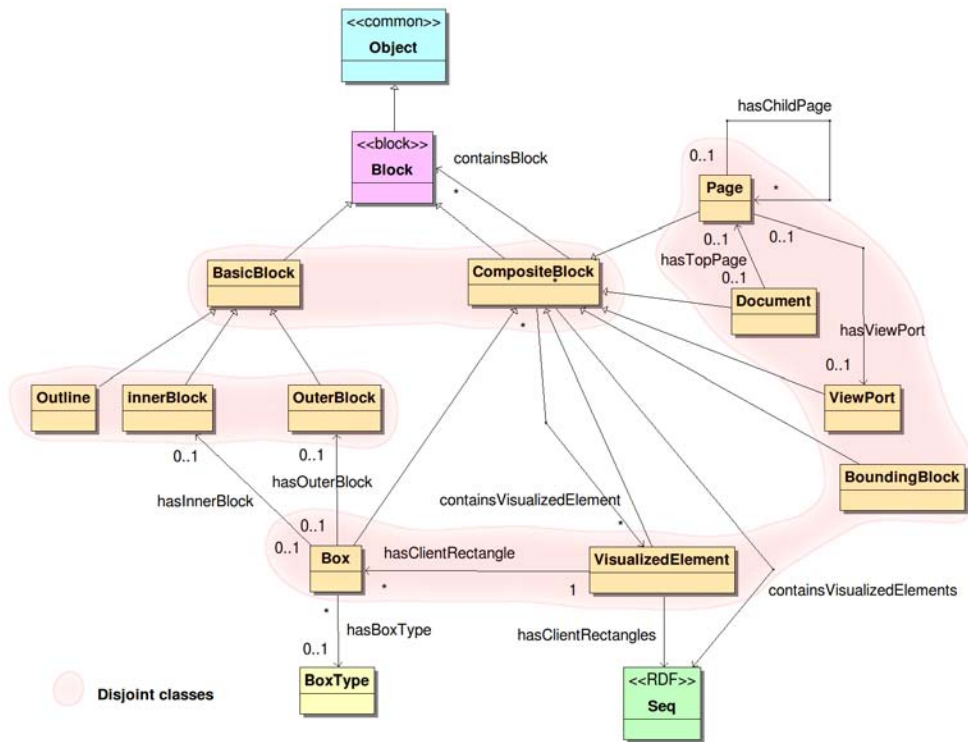


Figure 4.8: Classes of the Structural Block-based Geometric Model (StrBGM)

An *inner block*, *outer block* and *outline* are basic blocks. Inner block corresponds to the content of a box, while outer block includes the entire box (content and border). *Outline* wraps some arbitrary part of a web page and can have no correspondence to any DOM elements.

The containment of blocks can be realized as a set of RDF triples (e.g. utilizing object properties `containsBlock`, `hasChildPage`, `containsVisualizedElement`, and `hasClientRectangle`) or as a triple with RDF sequence for *object* (e.g. utilizing object properties `containsBlocks`, `hasChildPages`, `containsVisualizedElements`, and `hasClientRectangles`, see Figure 4.9).

Quantitative BGM

The Quantitative Block-based Geometric Model (QntBGM) refers to the definition of the *quantitative model* introduced in Section 3.2 and corresponds to $\mathcal{ALCFH}(D)$ DL. The QntBGM enriches the StrBGM with quantitative information describing the location, spatial expansion (see Section 3.4), direction (see Section 3.7.1) and distance (see Section 3.8.1) between blocks. The QntBGM is illustrated in Figure 4.10.

For simplicity, reification was not used and all the quantitative relations are presented by means of Relation class, which has corresponding object properties `hasPrimaryObject` and `hasReferenceObject` and datatype property `hasFloatValue` with `xsd:float` data type.

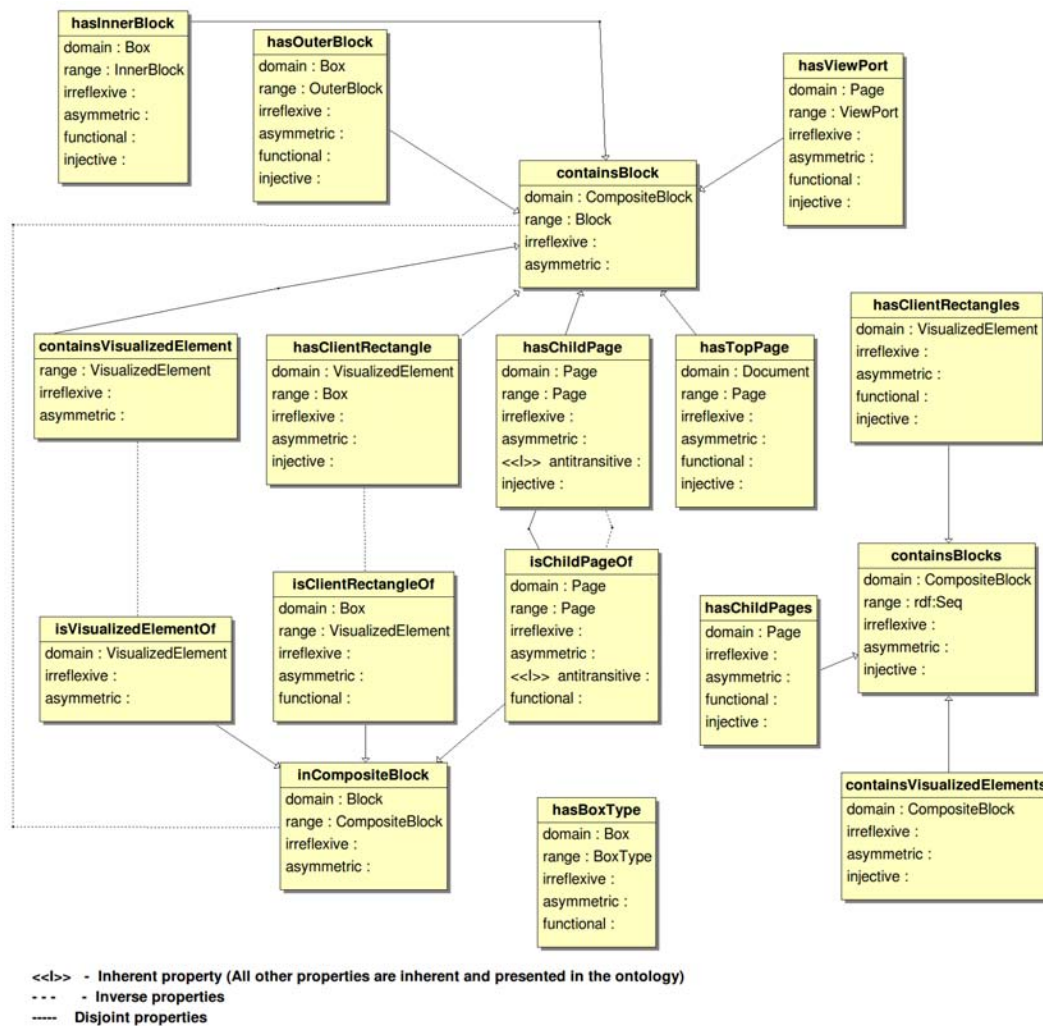


Figure 4.9: Object properties of the Structural Block-based Geometric Model (StrBGM)

Qualitative BGM

The Qualitative Block-based Geometric Model (QltBGM) enriches the StrBGM with qualitative information, such as the topology (see Figure 4.11 and Section 3.6), direction (see Figures 4.12, 4.13, 4.14, and Section 3.7.2), distance (see Figure 4.18 and Section 3.8.2), alignment (Figure 4.19 and Section 3.9), and interval relations which form Two-Dimensional Interval Relations with Centering (2DIRC) (Figure 4.20 and Section 3.10). The QltBGM also enables developers to express relations such as neighborhood (see Figure 4.15) and orthogonal visibility [84, p. 12] (see Figure 4.16 and 4.17). The QltBGM is compliant with the definition of qualitative model presented in Section 3.2 and corresponds to $\mathcal{ALIFH}(D)$ DL.

We do not use disjointness for the qualitative relationships, even for those which originally are pairwise disjoint, for example, topological relations RCC8. This fact is due to the use of 2DIRC

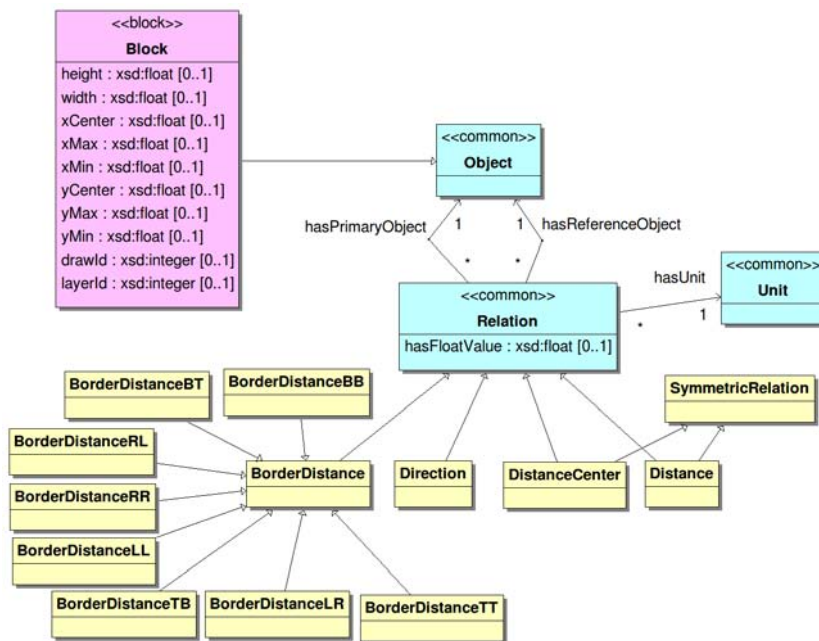


Figure 4.10: Classes and datatype properties of the Quantitative Block-based Geometric Model (QntBGM)

for representing most of the qualitative relations according to Section 3.10.4, and the definition of 2DIRC in terms of fuzzy sets and fuzzy relations. Thus, in the corresponding qualitative model, the presence of more than one relation of the same type between certain pair of objects is allowed.

In the QltBGM we do not leverage a refinement of RCC8 introduced in Section 3.6 for the blocks. The main goal of this model is to convey the developer with the most useful and simple spatial relations with minimal complexity. Thus, RCC8 with eight topological relations provides one with the most important relationships, in contrast to 36 more specialized relations. Moreover, the former can be expressed by means of Two-Dimensional Interval Relations (2DIR) (see Section 3.10.4).

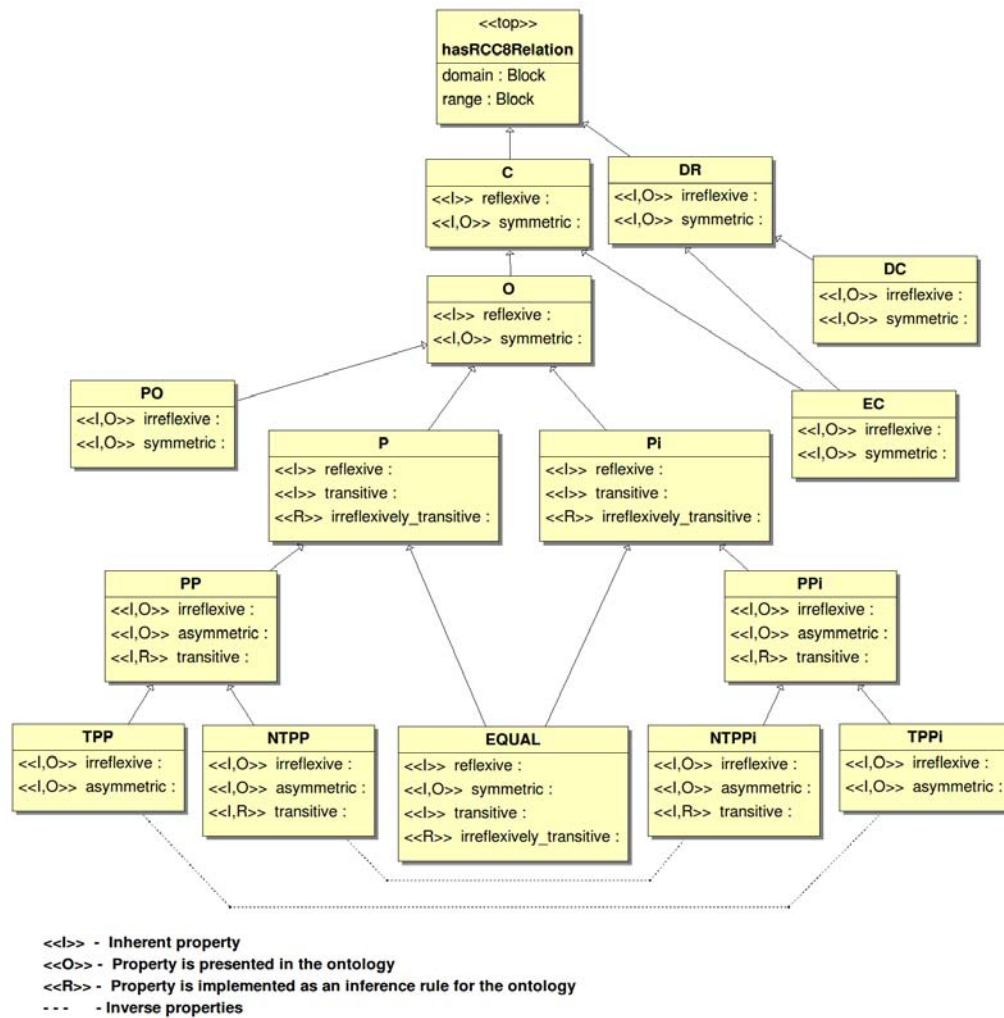


Figure 4.11: RCC8 relations in the Qualitative Block-based Geometric Model (QltBGM)

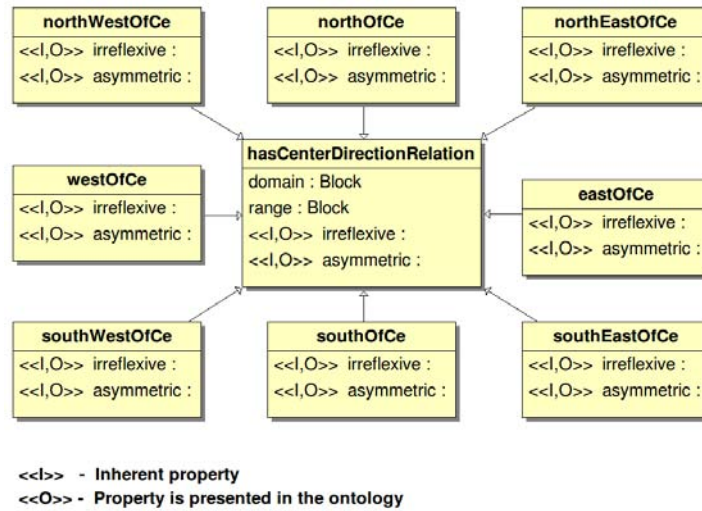


Figure 4.12: Center direction relations in the Qualitative Block-based Geometric Model (Qlt-BGM)

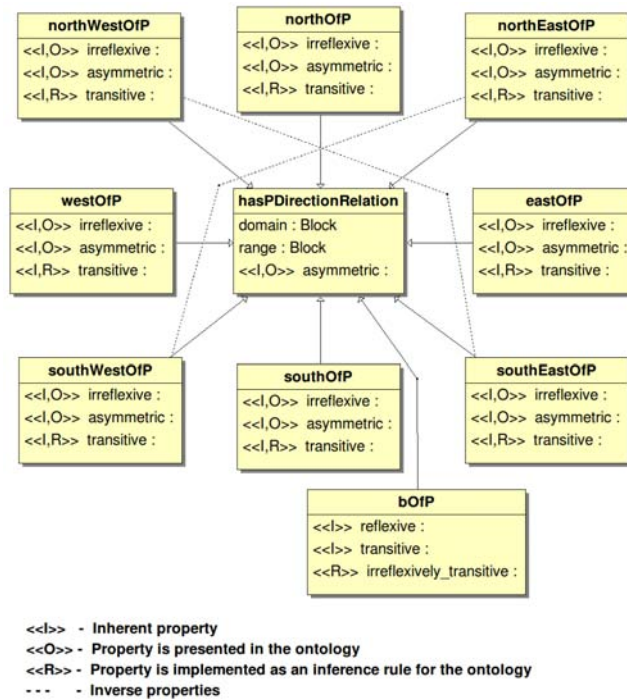


Figure 4.13: P-direction relations in the Qualitative Block-based Geometric Model (QltBGM)

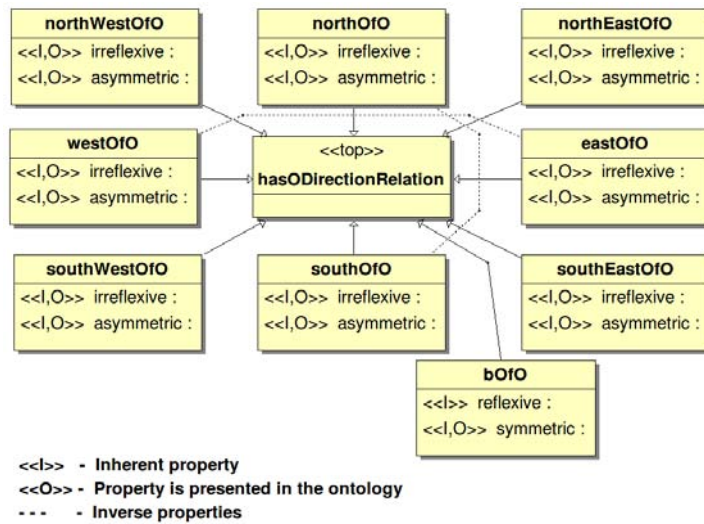


Figure 4.14: O-direction relations in the Qualitative Block-based Geometric Model (QltBGM)

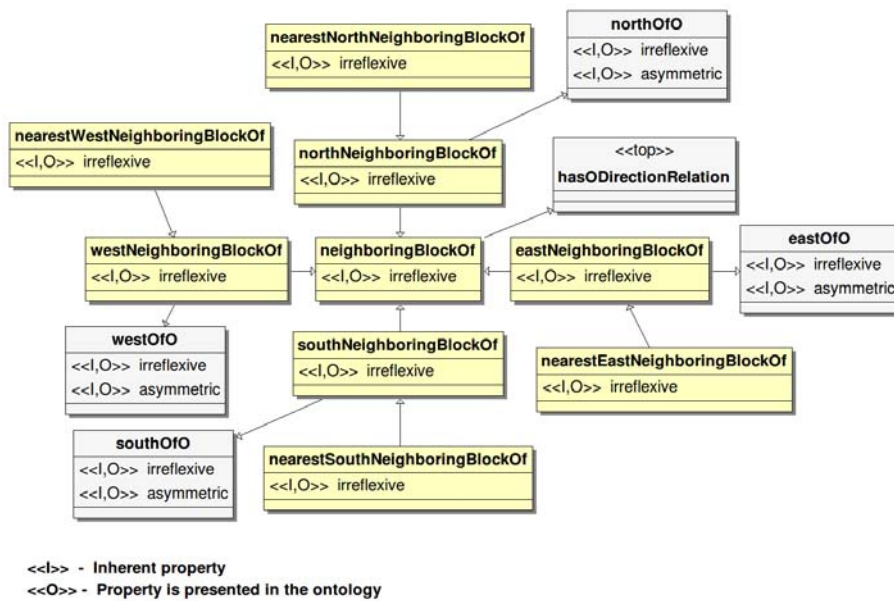


Figure 4.15: Neighborhood relations in the Qualitative Block-based Geometric Model (QltBGM)

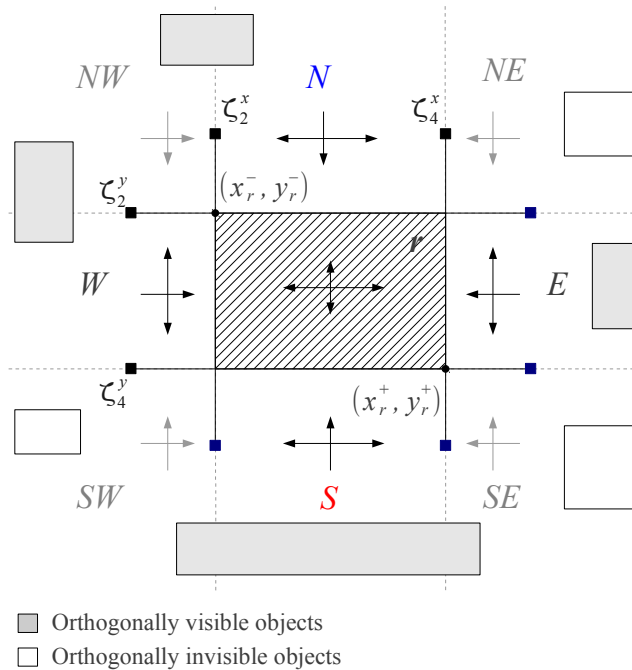


Figure 4.16: Orthogonal visibility

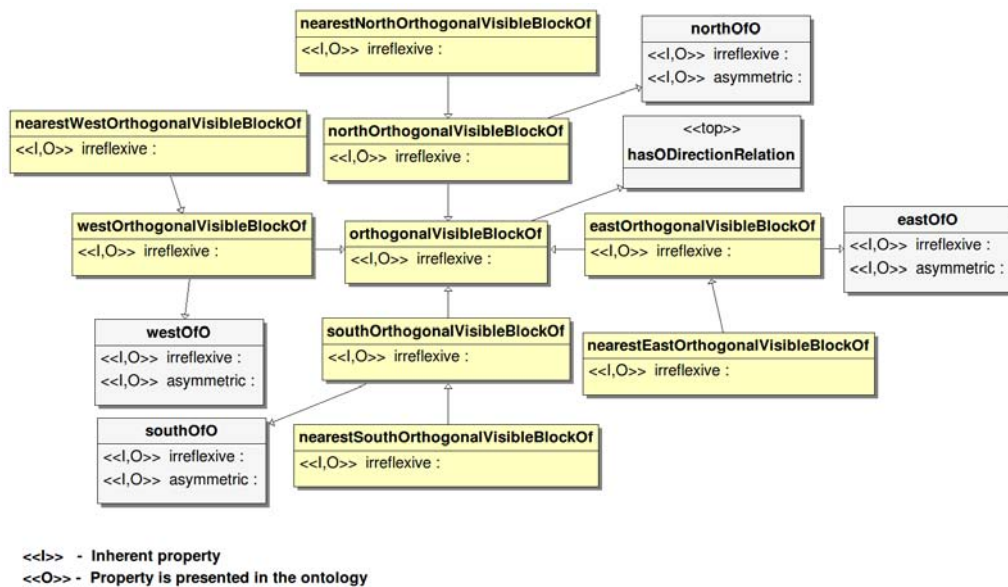
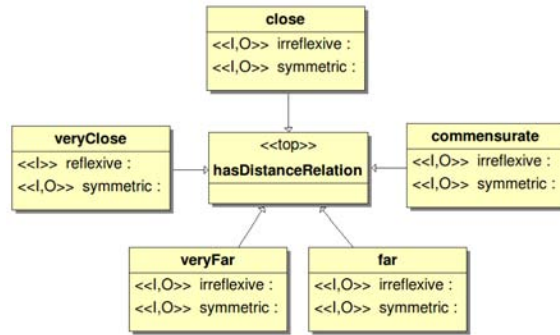
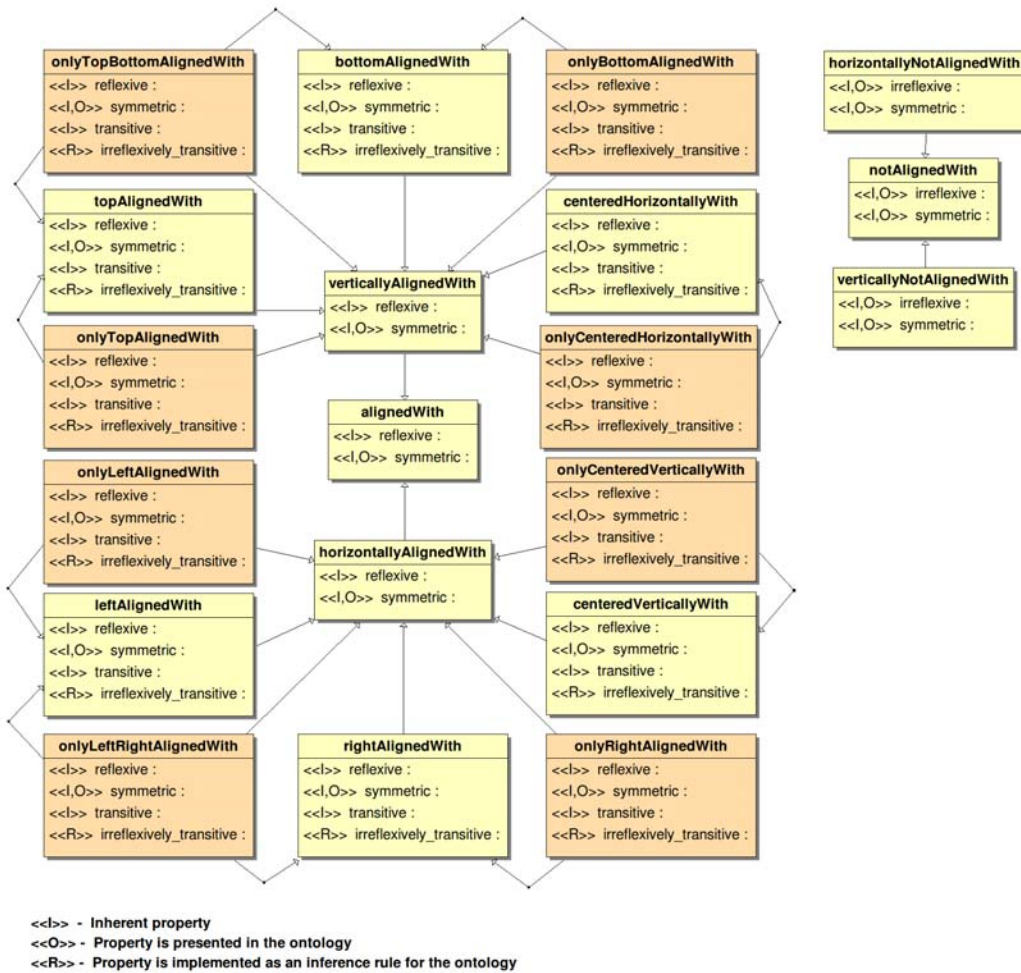


Figure 4.17: Orthogonal visibility relations in the Qualitative Block-based Geometric Model (QltBGM)



<<l>> - Inherent property
 <<O>> - Property is presented in the ontology

Figure 4.18: Distance relations in the Qualitative Block-based Geometric Model (QltBGM)



<<l>> - Inherent property
 <<O>> - Property is presented in the ontology
 <<R>> - Property is implemented as an inference rule for the ontology

Figure 4.19: Alignment relations in the Qualitative Block-based Geometric Model (QltBGM)

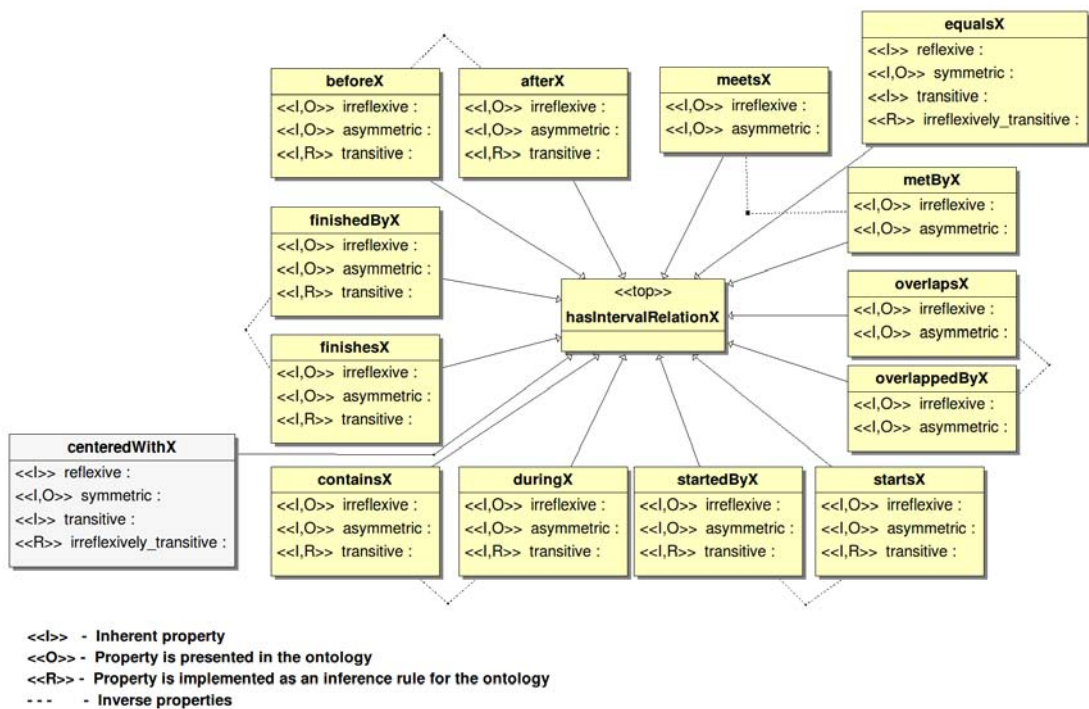


Figure 4.20: Interval relations in the Qualitative Block-based Geometric Model (QltBGM) for the x -projection (on abscissa)

4.4.3 Interface Model

The Interface Model (see Figures 4.21, 4.22, and 4.23) represents the functional elements of a web page (e.g. links, buttons, images) and basic structures (e.g. lists, tables), with complexity comparable to $\mathcal{ALCCINH}(D)$ DL. This model can be formed mainly based on the DOM* model (see Section 4.4.1) taking into account elements' names, values, and CSS attributes. The main element of the model is `WPBIElement` which is a subclass of `Object`. This class is an abstract element of a web page's interface, and together with other subclasses, whose name starts with "WPBI" (see Figure 4.21), represents elements of an interface independent from the source code and mostly defined by the computed CSS attributes. Thus, some elements independent from the source code reflect the layout of a web page, for example, `WPBITable` and `WPBIList`.

Considering a web page encoded in X/HTML, `HtmlElement` is the most abstract element (see Figure 4.22), which is a subclass of `WPBIElement` and defines the X/HTML specific objects. These objects are mainly defined by the information acquired from the DOM trees regardless of the CSSOM. Thus, source code dependent elements mostly define different types of DOM tree elements, default representation and spatial allocation which can be changed by the CSS style sheets [150]. Due to this fact, these elements should be considered with caution in terms of information extraction and web page understanding methods.

Based on information considered in the DOM*, we distinguish three main groups of elements:

- Elements defined by the name of the DOM* element (datatype property `hasNodeName`). For instance, `HtmlLink` is defined by the element with name `A`, `HtmlImage` corresponds to the element's name `IMG`, `HtmlTable` corresponds to `TABLE`.
- Elements defined by the value of an attribute of the DOM* element (class `Attribute`). For instance, elements such as `WPBICheckBoxGroup`, `WPBIRadioButtonGroup` are identified by the `hasNodeName` attribute of the corresponding check boxes and radio buttons in the DOM*; elements such as `HtmlTextInput`, `HtmlSubmitButton`, `HtmlPasswordInput`, `HtmlRadioButton` are defined based on the value of the attribute `type` of the element `input`.
- Elements defined by the value of a CSS attribute (datatype property `hasComputedCssCode`). For instance, `WPBIImage` can be defined by the CSS attribute `background-image`, `WPBITable` is defined by `display` with values `table` or `inline-table`, `WPBIListItem` is defined by `display` with value `list-item`.

The Interface Model provides us with necessary information regarding the functional components provided by the web page.

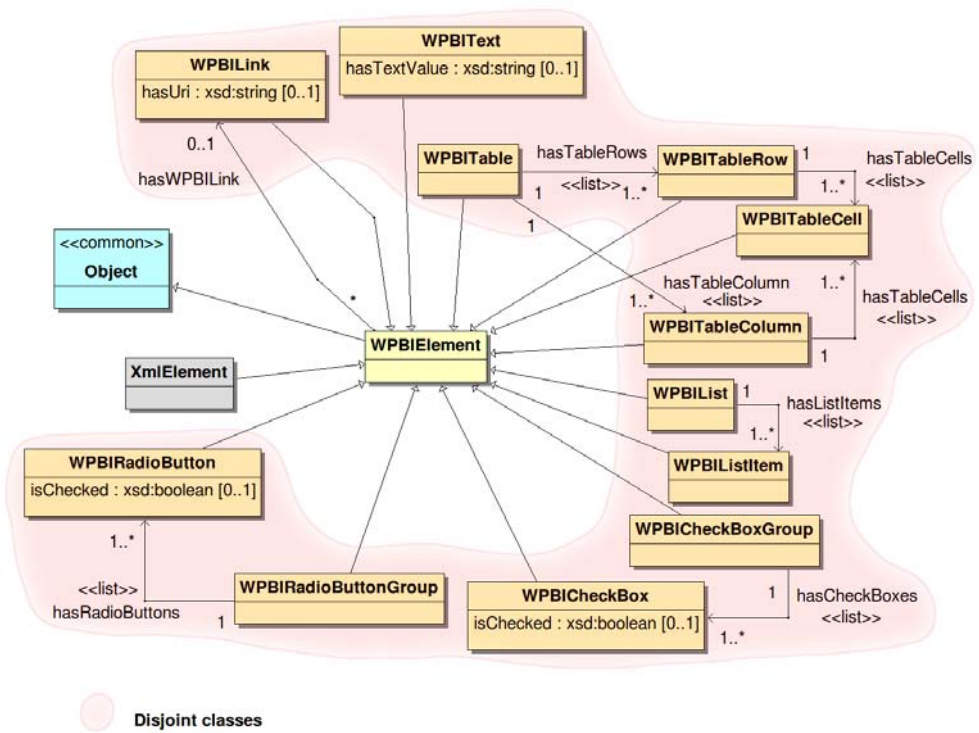


Figure 4.21: Main classes of the Interface Model representing abstract interface elements

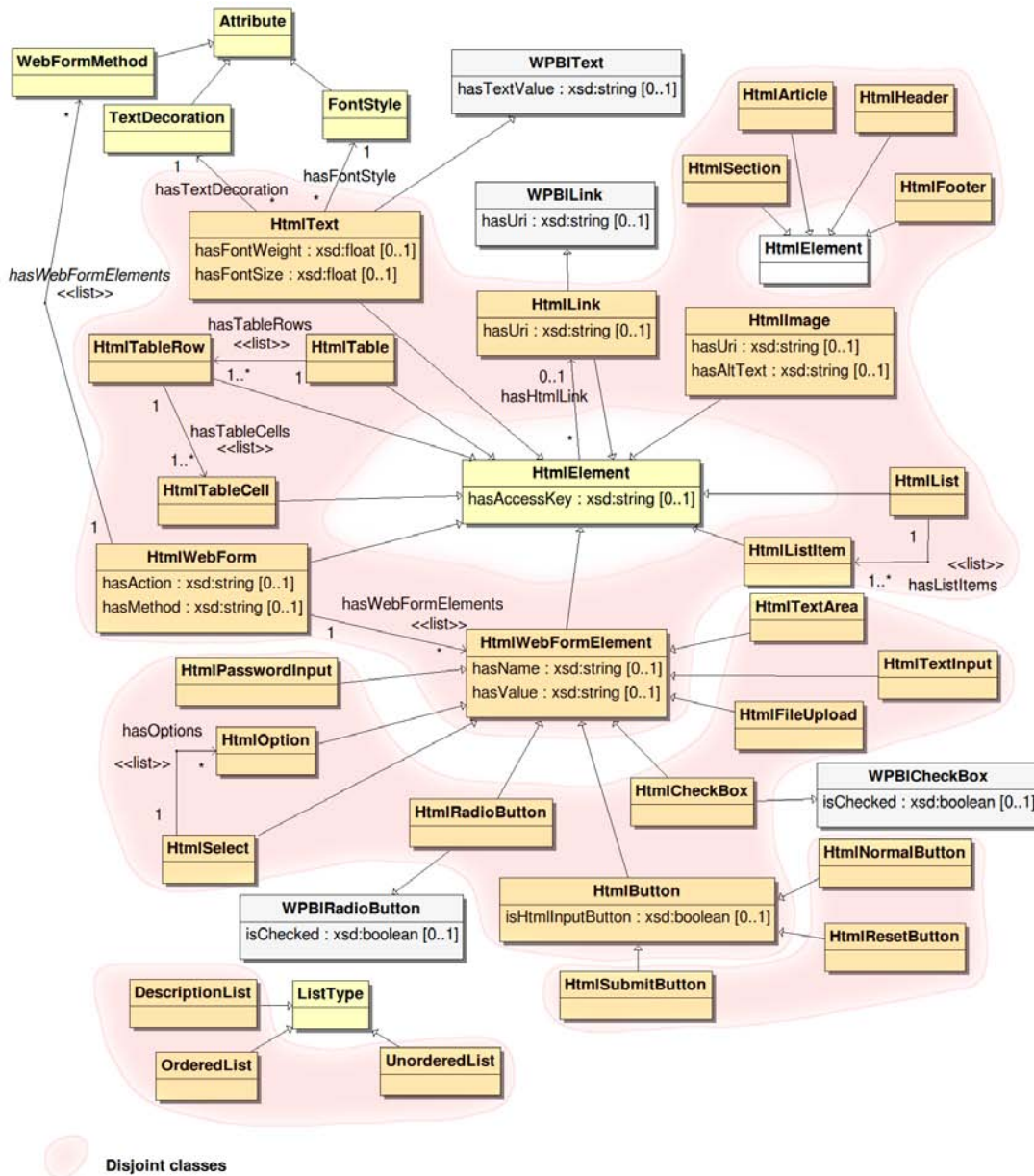


Figure 4.22: Classes of the Interface Model representing HTML-based interface elements

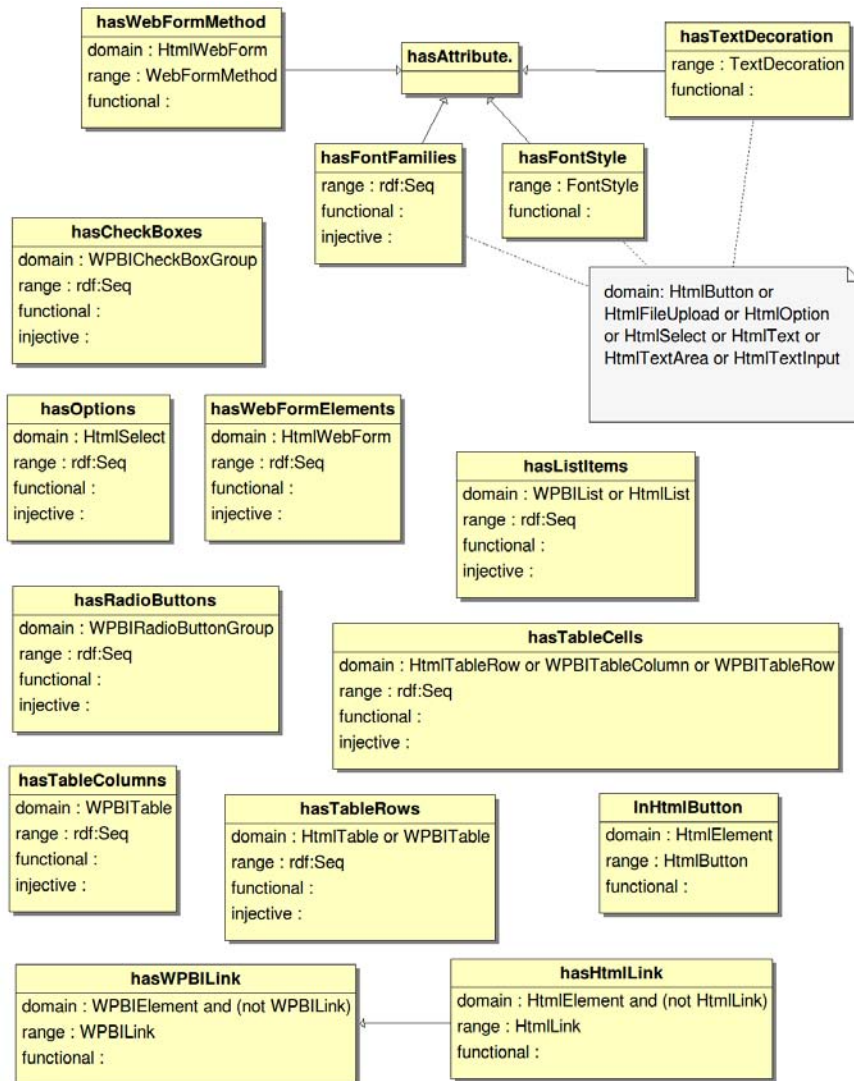


Figure 4.23: Object properties of the Interface Model

4.5 The Logical Model of a Web Page

The Logical Model (LM) corresponds to the logical layers of the conceptual model of a web page (see Section 4.2) and is used to model semantics of a web page. This model is intended for integration with the Semantic Web and providing human- and machine-readable representation for integration with external systems. Using various vocabularies and domain-specific ontologies, one can model different web specific (e.g., navigation menu, main content, header, and footer) and domain specific objects (e.g., web forum post, news article and corresponding users' comments). Objects instantiated in the LM can be also integrated into datasets such as DBpedia³, UMBEL⁴, GeoNames⁵, YAGO⁶, and WordNet⁷ by means of the Linked Data and thereby extending the semantics of the entities recognized on a web page as well as their seamless integration with the Semantic Web technologies. It is important to note that the LM can be represented by other formalisms different from OWL, for example, Datalog \pm [43] and $H\mu L\epsilon X$ [173] used in the problems of WIE.

In this thesis, we do not focus on modeling various web page objects and only represent the main data structures that relate to the data structure layer of the conceptual representation of a web page (see Section 4.2). ABox (schema) of the ontology is presented in the `logical-model.owl` file (see Figure 4.24) and corresponds to $\mathcal{ALUIFH}(D)$ DL. The ontology has RDF-based-semantics [281] and is serialized using RDF/XML syntax. There are four main data structures in the LM: node, sequence, tree, and grid (see Figure 4.25). A node is basic data structure which can be used to represent simple objects on a web page such as author name, price, and date. A sequence can be used to represent lists, such as a simple navigation menu as a sequence of items, product list, search results, etc. A tree can be used for hierarchical navigation menus, to represent a tree of replies in the web forum, or a tree of comments for the article. A grid is used to model basic tables on a web page.

Data structures, mainly treelike structures, are commonly used in representing logical structure of a document in the field of Document Understanding [161, 228] and Web Page Understanding [41, 104, 300]. These data structures are used in this thesis for building the MANM which requires such strict definitions of the logical objects (see Chapter 6).

4.6 Discussion

In this chapter, we have introduced the Unified Ontological Model (UOM) and its main component, Physical Model (PM) and Logical Model (LM), represented by means of OWL 2 DL (mainly with RDF-based semantics [281]), and thus it is serialized using RDF/XML syntax. The corresponding ontologies of the UOM are illustrated with Class Diagrams. The UOM formalizes some layers of the conceptual model of a web page that represent various levels of a web page's abstractions. The PM is the result of *web page analysis* (see Chapter 5). It models a web page's layout in the form

³<http://dbpedia.org>

⁴<http://umbel.org/>

⁵<http://www.geonames.org/>

⁶<http://www.mpi-inf.mpg.de/yago-naga/yago/>

⁷<http://wordnet.princeton.edu/>

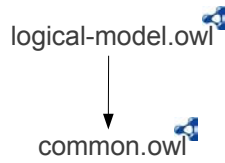


Figure 4.24: Ontology files of the LM [80]

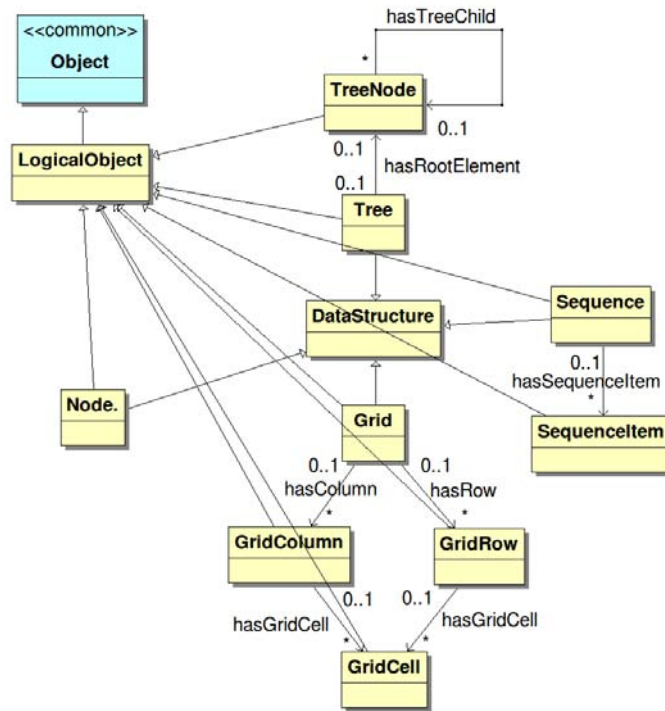


Figure 4.25: Main data structures of the Logical Model (LM)

of Block-based Geometric Model (BGM), Interface Model, and DOM* (a structure established by the DOM trees). The information necessary to represent the PM can be acquired from the analysis of DOM trees and CSSOM of a web page. The LM is the result of analysis of a web page's PM (see Chapter 5), and corresponds to the higher layers of a web page's conceptualization. The LM can be used to provide the user and computer with semantically rich information that describes logical structures and meaning of a web page's content. Furthermore, users can utilize Linked Data approach to enrich the LM.

Web Page Processing

A new kind of document often means a complete development of a new recognition system. This is a real lost of energy which can be avoided by defining a generic system: a generator of recognition systems for structured documents.

— Bertrand Coüasnon, *International Journal of Document Analysis* 8(2), 2006

In Section 5.1 we discuss the process of web page processing (WPP), presented in Section 2.3, from the viewpoint of the application of the Unified Ontological Model (UOM) (see Chapters 3 and 4). As presented in Section 5.2, this vision, as well as the presence of declarative and imperative approaches for web page understanding and web information extraction, motivated us to design and implement an abstraction method which makes the application of both paradigms possible. This ontology abstraction, the UOM, and the proposed concept of WPP were realized in the Web Page Processing System (WPPS) presented in detail in Section 5.3. Taking into account different aspects of web page representation, the WPPS is a basis for developing effective and robust methods. Principles and examples of realizing WPP methods by means of WPPS are discussed in Section 5.4. An evaluation of the efficiency of WPPS is conducted in Section 5.5. In Section 5.6, we consider the challenge of web object identification as well as the use of WPPS as a framework for rapid application development. Section 5.7 concludes the chapter.

5.1 A Principle of Web Page Processing based on the Unified Ontological Model

Within the scope of this thesis, web page processing (see Section 2.3) is referred to different methods which are leveraged within the fields of Web Page Understanding (WPU) and Web Information Extraction (WIE). As applied to the UOM (see Chapter 4), we present WPU and

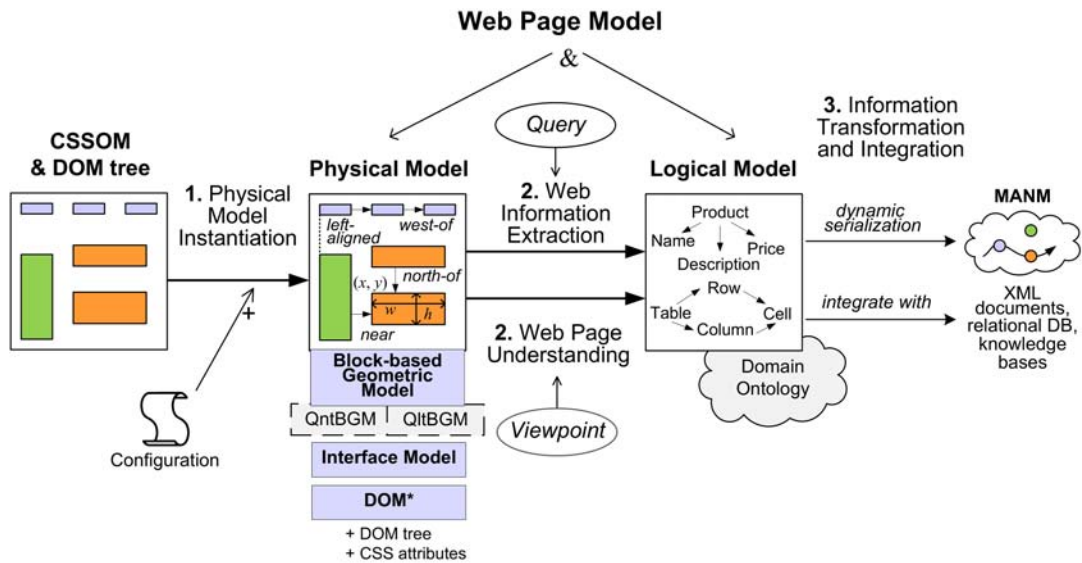


Figure 5.1: Data flow diagram of the process of WIE and WPU

WIE processes consisting of three main phases (see Figure 5.1): 1) Physical Model instantiation (analysis of a web page’s technical representation), 2) web page understanding and direct web information extraction, 3) information transformation and integration.

Physical Model instantiation is based on the data models (i.e., DOM trees and CSSOMs) provided by the web browser engine (see Section 5.3.3). This process can be controlled by the specified configuration which defines necessary constraints (i.e., what should be modeled and materialized in the certain instance of the Physical Model (PM)). In turn, WPU and direct WIE are performed leveraging the PM. The aim of these two processes from the viewpoint of the UOM is to provide an interpretation in the form of the Logical Model (LM) for the concepts of the PM by means of domain ontologies and Linked Data technology (see Section 4.5). Thus, we represent WIE as a process of the PM analysis performed according to the *query* specified. A query can be represented by the wrapper which defines the necessary information to be extracted as well as its main characteristics. It can be realized as an algorithm, template or a query over ontology, for instance, SPARQL query. In contrast, from the viewpoint of the UOM, methods of WPU operate over the PM mainly targeting the whole web page. The main goal of WPU methods is to derive semantics of a web page hidden in the materialized PM according to the *viewpoint* indicated. The viewpoint defines the necessary granularity which is required for describing the web page semantics and realizing it in the materialized LM. For example, WPU can be limited to the web page segmentation and building a segmentation tree in the LM, or it can consider more web specific objects such as a navigation menu, main content, header, or footer. Information transformation is a process aimed at providing information in the form appropriate for external applications. It can be represented as XML document, tuples in the relational database, or assertions in the knowledge base. In this work, we transform the UOM into the Multi-Axial Navigation Model (MANM) (see Chapter 6), an ontological model that provides a navigation

means for blind users and ultimately making the Web more accessible.

5.2 An Object-Oriented Abstraction for the Unified Ontological Model

An application of the UOM in the web page processing requires a presence of mechanisms which enable leveraging both declarative and imperative (procedural) approaches. This necessity is related to the active development of various methods of web page processing which are mainly implemented using imperative languages (see Section 2.3 and 2.4). In this section, we place emphasis on both approaches. However, we discuss the challenge of using imperative methods on the UOM, particularly the object-oriented, in more detail.

5.2.1 Declarative and Imperative Approaches

Ontology provides great opportunities of applying various declarative approaches which enable automatic reasoning and logical deduction as well as querying (see Section 2.6). Declarative languages provide an elegant way of writing programs by describing *what* should be accomplished rather than specifying *how* it should be realized. This allows the development of compact programs. Declarative languages also have a correspondence to formal logic. In addition, a declarative program is considered as a theory whereas computation is a deduction from this theory. The disadvantages of the declarative approach includes limited expressive power and limited means for program optimization. Examples of such languages, which are used for Web Data Extraction (WDE) from the ontology, are Datalog \pm [43], H $\mathcal{L}\mathcal{E}\mathcal{X}$ [173], and simplified web pattern matching language (SWPML) [151].

Datalog \pm [43] is a family of languages which are extensions of Datalog. It allows features such as existential quantifiers, the equality predicate, and the truth constant *false* to appear in rule heads. Thus, Datalog \pm , in contrast to Datalog, are richer languages in modeling TBox. However, these languages are syntactically restricted, so as to achieve decidability and in some cases even tractability. Datalog \pm can also be applied for the WDE tasks from the UOM, which in turn should be spelled into Datalog \pm program consisting of the intensional and extensional components. The former represents TBox of the materialized UOM (i.e., statements regarding the domain of web pages) according to its own expressive power, while the latter models ABox of the materialized UOM (i.e., facts regarding the relevant web page).

H $\mathcal{L}\mathcal{E}\mathcal{X}$ [173] language is dedicated to information extraction from the non-structured documents which are represented as the OntoDLP ontology. This ontology is dedicated to describing certain linguistic and structural characteristics (e.g., tabular structures) of a web page. In general, the ontology can be formed as a result of applying NLP techniques, such as part of speech tagging and entity recognition as well as a result of a certain analysis of the layout (e.g., table recognition). Thus, H $\mathcal{L}\mathcal{E}\mathcal{X}$ ontology can be used to represent a layer of textual content semantics and data structure layer of the LM of the UOM respectively (see Section 4.2). In addition, H $\mathcal{L}\mathcal{E}\mathcal{X}$ query language represented by the semantic rules can be applied for querying the model.

Simplified web pattern matching language (SWPML) [151] is a domain specific language based on Java for information extraction from the ontology. The language was designed and

approved on one of the earlier versions of the UOM. It provides the ability to describe a pattern which is evaluated over RDF graphs. The concept of SWPML is loosely inspired by regular expressions. Thus, patterns can be repeated, marked optional and grouped to more complex patterns. In contrast to regular expressions, patterns are evaluated over two-dimensional space, and therefore SWPML has relevant constructs for connecting different sub-patterns which are evaluated over different spatial dimensions. Implementation of the SWPML works in conjunction with SPARQL for evaluating simple queries over ontology for the objects required within the pattern.

In contrast to the declarative approach, within an imperative approach, a computation process is described in terms of instructions or commands which change the state of the program. Among imperative languages, structured (e.g., C, Cobol, Basic) and object-oriented (e.g., Java, C#) are distinguished. There are also procedural languages which realize object-oriented paradigm, such as C++, Perl, and Python. Imperative languages are leveraged for implementing algorithms, for instance, those mentioned in Sections 2.3 and 2.4 for WIE and WPU. However, object-oriented paradigm is the most suitable for representing certain aspects of the ontology. For instance, it introduces concepts of class and object (as *instance* of certain class), its attributes (*data fields*) and associated procedures (*methods*). Thus, a class can be associated with the class in the OWL ontology and the concept in Description Logics (DL) while an object can be associated with the object in OWL and the instance in DL and data fields can be represented by the properties in OWL and roles in DL. Also in contrast to the declarative approach, the imperative allows ample possibilities for program optimization. However, disadvantages include the fact that programs are more cumbersome and usually not as intuitive as declarative programs.

Therefore, in terms of developing efficient and effective methods for WIE and WPU for the UOM, both paradigms should be considered.

5.2.2 A Required Abstraction

To provide an access to the instances of the UOM for both declarative and object-oriented approaches, we need to develop a certain level of abstraction (see Figure 5.2).

For declarative languages such as SPARQL [257] and SWPML [151], this abstraction is not necessary. This is due to the fact that the UOM and, in particular, the PM, can be easily spelled into the RDF syntax which is a target ontology representation for leveraging these languages. In contrast, Datalog \pm [43] and HiL \mathcal{E} X [173] require certain transformations which enable their straightforward applications. For example, Datalog \pm [43] requires representation of the UOM in Datalog \pm style with intensional and extensional databases for ABox and TBox respectively. For HiL \mathcal{E} X, the ontology should be represented in terms of OntoDLP. This transformation is out of the scope of this thesis. Thus, when considering declarative approaches in this thesis, we focus primarily on SPARQL-based queries and Jena-based inference rules and reasoning [10].

In turn, object-oriented languages are related to object-oriented domain models. Such models are much less expressive than ontology languages (e.g., RDF(S), OWL or DL). In addition, they have methods implementing certain functionality and algorithms with side effects (i.e., dependent on any hidden or external information which can change the behavior of the program). Thus, a difference in the nature of ontological and object-oriented languages demands a creation of an object oriented abstraction over the UOM.

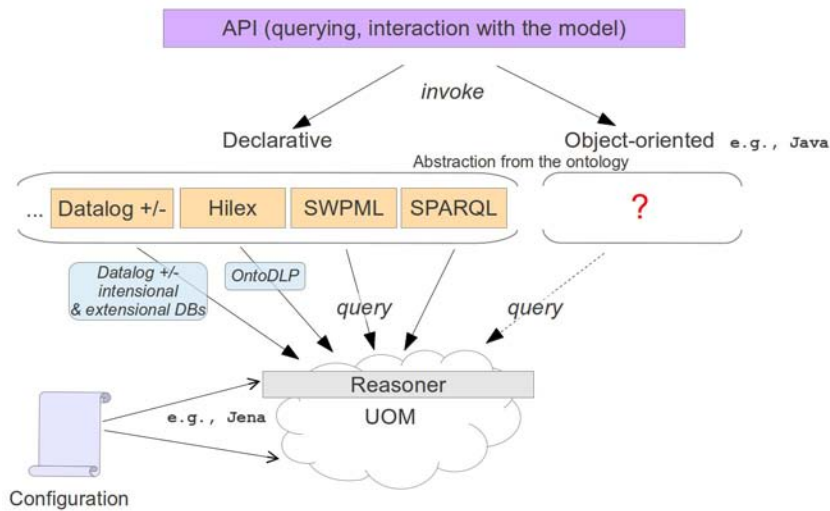


Figure 5.2: A required abstraction from the ontology for applying methods based on the object-oriented paradigm

We specify four main requirements:

- Req. 1** There should be an option to choose a necessary reasoner for the UOM from the pre-defined set. Furthermore, different instances of the UOM (i.e., ABox) can have different requirements for the reasoner. Reasoners can only differ on the totality of considered axioms in TBox of the UOM and the set of considered inference rules.
- Req. 2** A set of available classes and properties to be instantiated for the UOM should be specified by the developer. Thus, various instances of the UOM generated for the same web page can have different types of instances and properties instantiated. Moreover, different properties can be used for specifying one-to-many relation between individuals. For example, in the Structural Block-based Geometric Model (StrBGM) (see Section 4.4.2), relation between a page and a collection of the visualized elements can be specified either by the set of assertions with property `containsVisualizedElement` or by means of RDF container, carrying a collection of the visualized elements and is related to the page via property `containsVisualizedElements` (see Figure 4.8 on page 114).
- Req. 3** There should exist the possibility to specify which properties and class taxonomies, based on the information contained in the ontology, should be computed on the object-oriented level by request (i.e., “on-the-fly”), as well as which of them should be computed within the ontology by the indicated reasoner (according to Req. 1).
- Req. 4** Modifications which were done on the ontology level in ABox and the results of reasoning should be available in the object-oriented level and vice versa.

Thus, when an abstraction satisfies the specified requirements it provides the developer of web page processing methods the possibility of working on the object-oriented level to

leverage the same application programming interface (API) regardless of various valid substantial modifications within the instances of the UOM. Further more, these requirements allow the developer to leverage the benefits of declarative (e.g., with SPARQL querying and automatic logical reasoning) and object-oriented approaches and have the possibility to investigate various features of a web page to develop effective algorithms and approaches. For example, Req. 1 enables the developer to choose a necessary reasoner for the UOM. The reasoning can be performed over subsumptions of classes (class-subclass) and properties (property-subproperty) according to the RDF or OWL semantics. Also, various property and class constraints can be taken into account. In addition, the developer should have the option to select a set of inference rules from the ones available for the UOM to be applied for the ontology; for example, inference rules which derive topological RCC8 relations on the set of blocks from their quantitative position and spatial extension (see formulas (3.2) on page 69 and (3.3)). Req. 2 allows the developer to choose relevant classes and properties which should be instantiated in the ontology. It allows her to avoid an exhaustive instantiation of the whole ontological model and the ability to concentrate on the relevant aspects of the model. Req. 3 provides the developer with the possibility to control which computations should be done on the object-oriented level. It allows taking into account fuzzy relations with the inaccuracy specified in the configuration. Collections can be utilized if the order of elements is important. Req. 4 ensures the connection of ontology with the object-oriented abstraction.

5.2.3 Ontology in Object-Oriented Applications

In this section, we consider only approaches which provide high-level API reflecting relevant application domains. Therefore, Semantic Web frameworks such as Jena [10] and OWL API¹ with low-level API for managing the ontology are omitted.

One of the main challenges considered in model-driven engineering includes the problem of providing object-oriented representation of the application domain for a given ontology. We only mention the approaches which aim at tackling this problem that are in accordance with our objectives and requirements (see Section 5.2.2).

A. Kalyanpur et al. [133] introduce a method of representing certain ontological ABox assertions of OWL in terms of object-oriented paradigm by example of Java. In particular, the authors introduce guidelines of how various class definitions in OWL (i.e., sub-classes, enumerations, unions, intersections, disjoint classes, etc.) can be modeled by means of interfaces and classes of Java. The authors also describe how various OWL property constraints (i.e., symmetry, transitivity, equivalence, cardinality, etc.) can be ensured in Java. P. Bartalos et al. [18] extend this solution with algorithms for providing mapping between ontology and Java objects. Thus, these approaches separate object-oriented representation from the ontology. The authors provide limited methods of keeping the two models synchronized. As such, the issue of collisions, when both models were modified, is not considered. It partially satisfy Req. 4. Furthermore, Req. 1–3 are not considered in these approaches.

À gogo [191] is a domain specific language which enables automatic generation of an API intended for instantiating object-oriented model and mapping it back into the ontology.

¹<http://owlapi.sourceforge.net/>

Unfortunately, there are no full specifications of the *à gogo* syntax and semantics at the time this thesis was written. Therefore, we cannot check this approach. *À gogo* language is to be implemented in TwoUse² toolkit. TwoUse provides graphical user interface (GUI) for modeling ontology leveraging UML notations. It is also worth mentioning the work [221], in which the authors of *à gogo* and TwoUse give a detailed survey of the use of ontology technologies for software modeling in model-driven development.

The work of P. Kremen et al. [147] is the most relevant to the specified requirements. The authors propose a methodology for designing ontology-backed software applications that ensures the evolution of the ontology while being exploited by one or more applications at the same time. The methodology relies on integrity constraints defined between the ontology and the application. Violation of such constraints explicitly signal that the current modification cannot be used in the operable applications. The solutions introduced in the paper [147] are implemented in a Java library JOPA³. They are compliant with Req. 2 and Req. 4. Req. 1 also can be relatively simple to realize in the architecture proposed by P. Kremen et al. However, the approach introduced by the authors do not take into account Req. 3. Object-oriented model, automatically generated based on the ontological model, contains Java beans modeling OWL classes. This enables directly accessing and modifying fields of Java beans without additional computations, which complicates synchronization between ontological and object-oriented models. The synchronization is to be explicitly invoked by the developer whenever it is required. As such, this certainly complicates the mutual application of declarative and imperative approaches.

5.2.4 A Bridged Adapter

Regarding the requirements on the object-oriented ontology abstraction specified in Section 5.2.2, we decided to consider this issue from a practical point of view. Thus, we have developed a design pattern which incorporates the introduced requirements.

We propose an object-oriented abstraction to be represented by three main components: *implementations*, *adapters* and *ontology generator* (see Figure 5.3). The adapters represent classes of the ontology and properties related to them. Thus, this component represents the main elements of the ontology on the object-oriented level. The ontology generator provides the main functionality for instantiating the ontology. In cases of the UOM, it is the PM generated within the process of *web page analysis*. The implementations provide an API for interacting with the objects of the ontology and their properties according to the ontology configuration. This component is used both by the adapters and the ontology generator. The implementations ensure certain independence from the concrete realizations of the instances of the UOM. The adapters and implementations play an important role in providing the developer with object-oriented abstraction and API independent from the ontology configurations. For the realization of these components, we introduce *bridged adapter*.

A *bridged adapter* software design pattern [76, 77] is based on patterns such as *adapter*, *bridge*, and *factory method*. This pattern is recommended if we need to provide access to a certain adapted object (*adaptee*) which has an interface or structure different from what is required, and

²<http://code.google.com/p/twouse/>

³<http://krizik.felk.cvut.cz/km/jopa/>

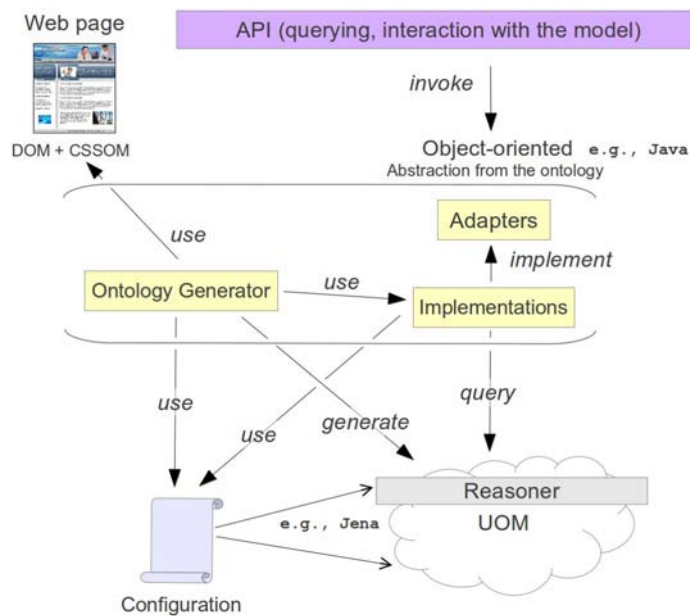


Figure 5.3: Main components of the object-oriented abstraction

where the adaptee either does not have a strictly defined interface (*behavior*) or its structure is not permanent. For instance, an individual of the OWL ontology representing a block in the Block-based Geometric Model (BGM) can have some subset of relevant datatype and object properties materialized, while some subset of other properties is taken to be computed upon request with parameters of fuzziness specified, and which can also be changed. In this case, an interaction with such an object and its use in algorithms can be problematic. Furthermore, it can lead to the so-called problem of boilerplate code. As a solution to this issue, the pattern proposed allows the developer to have the same *adapter* for the ontological object (individual) while *implementations*, which map different interfaces, are selected according to the specification of the object (*ontology configuration* corresponding to class `Configuration`).

In Figure 5.4, a class diagram depicts the proposed design pattern. `Adaptee` represents an adapted object (in our case, an individual of the ontology) and has a predefined configuration represented by the class `Configuration`. An interface `AbstractAdapter` is instantiated by the `AdapterFactory` and is a representation of the `Adaptee` required by the `Client`. An `Adapter` class implements the interface `AbstractAdapter` mapping it into the `Adaptee` by means of the `AbstractImplementor` provided by the `ImplementorFactory` during the instantiation of the `Adapter`. The `ImplementorFactory` dynamically creates an implementation of the `AbstractImplementor` interface, which corresponds to the object wrapped (*adaptee*) and its configuration. `Library` contains all necessary implementations for various valid configuration parameters of the adapted object. In terms of the UOM, a library should have an implementation of all necessary basic queries.

Thus, a bridged adapter pattern can be applied to certain classes in the ontology providing the developer with required adapters whose implementations correspond to the ontology con-

5.3 WPPS: A System for Web Page Processing

In this section we present WPPS [75–77], a Web Page Processing System designed and realized by the author of this thesis for the purpose of developing effective, efficient, and robust web page processing methods based on the UOM introduced in Chapter 4 and definitions given in Chapter 3.

WPPS is intended for: 1) the rapid development of new methods for web page understanding and information extraction tasks; 2) leveraging benefits of declarative and object-oriented approaches in accordance with the *bridged adapter* design pattern introduced in Section 5.2.4; 3) investigating the abundant forms of web page representations, relations and features formalized with the UOM for detecting those most appropriate for solving specified problems.

The framework provides various parameters for configuring ontological models and modes of their generation (see Section 5.3.2). Thus, the developer can specify a set of models, attributes, and relations the WPPS framework should instantiate in the UOM as well as methods for their computation (e.g., whether to store attributes and relations in the ontology or compute them “on-the-fly” based on the quantitative or basic qualitative relations). WPPS makes it possible to control a level of fuzziness for computing attributes and relations and provides a unified access interface via a WPPS API independent from a particular configuration of the UOM. Moreover, an integrated R-tree which also takes into account fuzziness specified makes the queries to the geometric space of a web page (i.e., the BGM introduced in Section 4.4.2) more efficient (with the complexity of search between $O(\log_m N)$ and $O(N)$ in contrast to the exhaustive search provided by the abstraction mechanisms with the time complexity $O(N)$). The application of the R-tree shows good results in practice, see Section 5.5.4. All these factors contribute to the novelty of the WPPS framework and its effectiveness in developing new web page processing methods.

5.3.1 Architecture

The WPPS framework is an Eclipse (Indigo) RCP based cross-platform application implemented in the Java language (JDK 1.7.0). It was successfully tested on different operating systems, such as Ubuntu, Mac OS X, Windows XP, and Windows 7. WPPS has XULRunner (version 1.9.2 corresponding to Firefox of the version 3.6) integrated for rendering web pages. XULRunner is a platform introduced by Mozilla for building rich Internet applications and has Gecko layout engine as its basis. It provides us with rich functionality which Firefox possesses and can be integrated into Java applications by means of the SWT libraries and plug-ins. WPPS also utilizes the ATF project plug-ins of the version 0.3.0, which enables seamless integration of XULRunner within the Eclipse RCP platform and conveys additional graphical components and widgets for interacting with the web browser (i.e., XULRunner).

The general architecture of the Blindzilla prototypes is illustrated in Figure 5.5. It consists of several components:

1. **UOM manager:** realizes the UOM by means of the Jena ontology framework [10] and applies required reasoners. It also provides access to the ontologies by means of API and SPARQL engines mainly implemented by Jena. It is allowed to apply external ontologies for describing individuals within the LM.

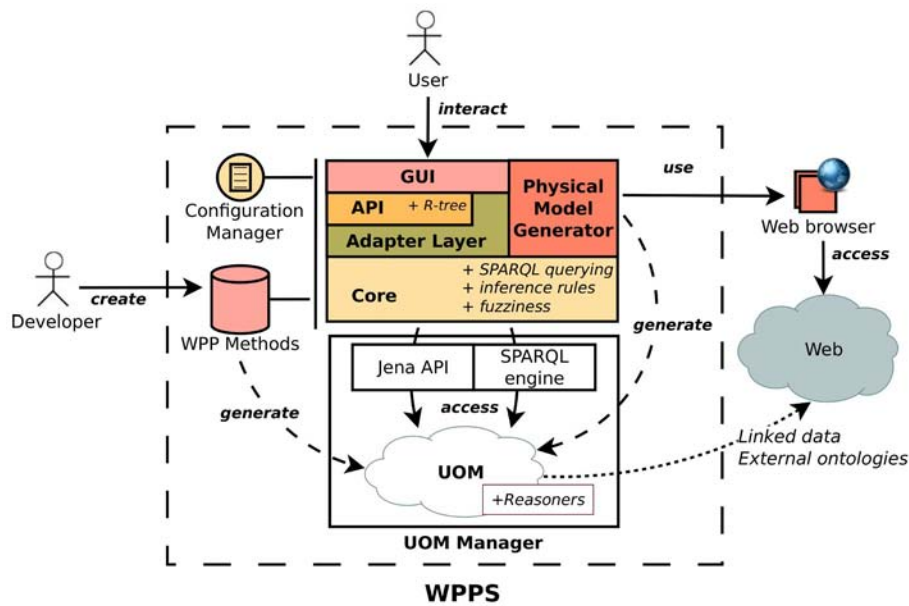


Figure 5.5: Architecture of the WPPS framework

2. **Configuration manager:** is responsible for configuring the WPPS framework; it controls settings of an instance of the UOM and modes of computing features and relations (see Section 5.3.2).
3. **Core:** provides the basic functionality to interact with an instance of the UOM via the UOM manager. As such, it possesses a collection of different implementations relevant to various valid configurations of the WPPS and allows the application of SPARQL queries and logical inference rules, which are handled by Jena. The core is also responsible for processing different inaccuracies (fuzziness) while computing qualitative attributes and relations, for example, as it is defined in Section 3.10.2 for spatial relationships.
4. **Adapter layer:** implements the bridged adapter software design pattern providing an object-oriented abstraction for classes and individuals of the ontologies according to their configurations. It enables the application of heuristics over the ontologies. Thus, the use of the adapter layer enables leveraging both declarative and object-oriented paradigms.
5. **Physical model generator:** is responsible for generating the PM of the UOM relevant to a certain web page and a configuration provided. It leverages Firefox 3.6 (XULRunner v.1.9.2) web browser integrated into the WPPS framework to obtain all necessary information (presented in the DOM trees and CSSOMs) for building the physical model (see Section 5.3.3). This component is easily extensible for other web browsers and sources (for example, PDF document).
6. **WPPS API:** is based on the adapter layer and provides the main functionality necessary for developing web page processing methods (see Section 5.3.4). It is designed for querying

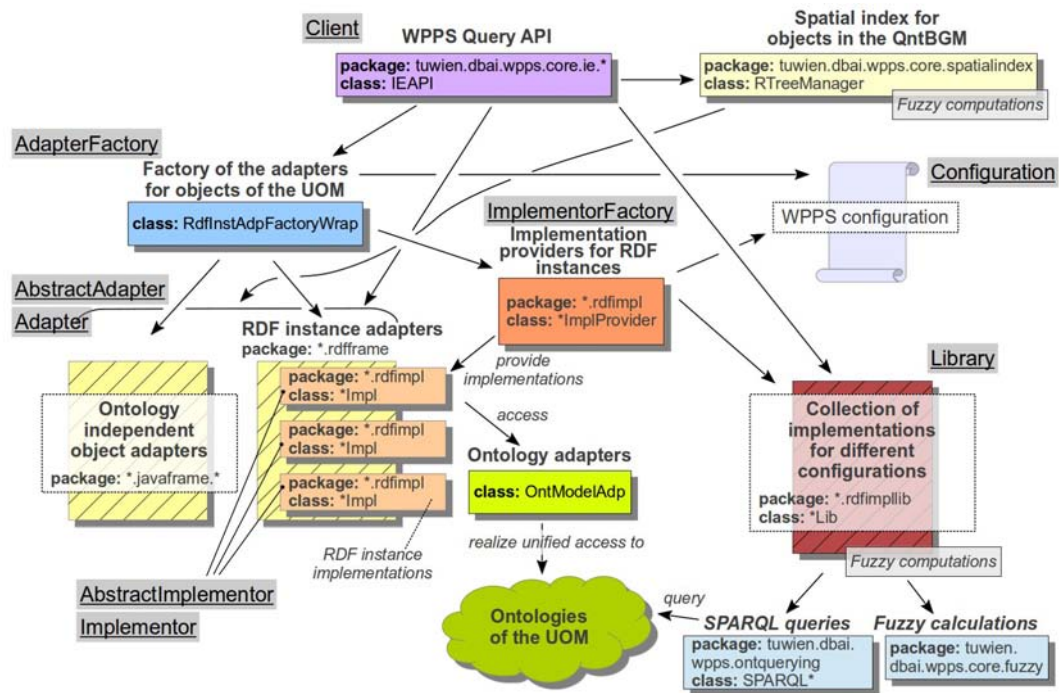


Figure 5.6: Implementation of the bridged adapter software design pattern in WPPS

the UOM, processing the information acquired, building the LM, and further integration of the LM with external systems (e.g., leveraging Linked Data and external ontologies). A WPPS API uses R-tree [105] for indexing two-dimensional objects (i.e., blocks) and performing efficient spatial querying.

7. **WPPS GUI:** a convenient interface for invoking developed methods, applying different configurations by the user and investigating various aspects of web page representation (see Section 5.3.5).
8. **Web page processing methods:** a set of methods with predefined configurations designed for solving specific problems (see Section 5.4.2). The collection is managed by the developer. Each method is primarily represented as Eclipse’s plug-in fragment.

Different components of the WPPS platform are also illustrated in Figure 5.6.

5.3.2 WPPS Configuration

Configuration of the WPPS framework is represented by an XML file which allows the developer of a new method to control the process of model generation including the computation of attributes and features, speeding up the process of building an instance of the PM and ensuring efficient interaction via the WPPS API provided. Although it is out of the scope of this thesis, WPPS configuration can be translated into the Datalog program and checked for the correctness. All

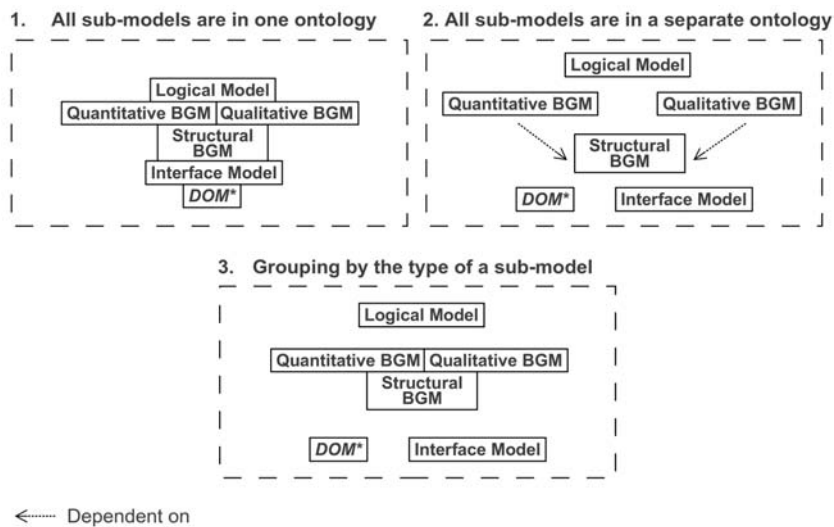


Figure 5.7: Main topological configurations of sub-models in a web page's Unified Ontological Model

the configuration parameters can be grouped into several categories: 1) models configuration, 2) objects configuration, 3) properties configuration, 4) fuzziness, 5) relevant web page area, and 6) simplification.

Models Configuration

When configuring the ontological models, the developer defines parameters such as:

- A) topology of the ontological models,
- B) URI of ontologies' name spaces,
- C) realization of the ontology in Jena either based on the RDF graph or a set of statements in Jena OWL model,
- D) location of the relevant schemata of sub-models which mainly contain TBox,
- E) the use of the schemata of sub-models in corresponding ontology instances,
- F) reasoners to be utilized,
- G) inference rules which should be applied over the relevant ontology.

A) Topology of the ontological models defines which sub-models of the UOM are to be created and which of them should share the same name space and therefore the reasoner and a set of logical inference rules to be applied. The sub-model is an instance of certain schemata corresponding to particular ontologies of the UOM and mainly conveying TBox. Of all possible topological configurations for an instance of the UOM, we would like to highlight the following examples (see Figure 5.7):

1. *All the sub-models are in the same ontology* sharing the same ontological name space. This configuration is suitable in cases when there is no reasoner applied or when a set of inference rules is generic enough to be applicable over different sub-models.
2. *All sub-models in a separate ontology* configuration is recommended when using different sets of inference rules for different sub-models. This topological configuration allows the reasoner to only consider statements of the relevant sub-models, which decreases computation time.
3. *Grouping by the type of a sub-model* enables instantiation of four separate ontological models corresponding to the LM, BGM, Interface Model, and Extended DOM. This topological configuration is efficient for the application of varying methods oriented to different sub-models. For example, the BGM can be used for layout analysis (table identification or main content discovery) and the Interface Model can be used for investigating a functionality provided by a web page. This grouping can also correspond to the groups of inference rules applied to different types of sub-models.

A topology can be specified by the element `ontology`, which denotes an ontological model created in Jena by the WPPS framework. As well, it is specified by its sub-elements with the name `sub-model`, which specifies a set of sub-models to be materialized in the ontology. An example of grouping sub-models by their type is presented in Listing 5.1. Each sub-model can be instantiated exactly in one ontological model.

Listing 5.1: Example of grouping sub-models by their type in the WPPS configuration file

```

1 <wpps-config>
2   <ontology>
3     <type>OWL</type>
4     <load-schemata>>true</load-schemata>
5     <reasoner-type>JENA_REASONER</reasoner-type>
6     <jena-reasoner>OWL_MEM</jena-reasoner>
7     <sub-model>STRUCT_BLOCK_GEOM_MODEL</sub-model>
8     <sub-model>QNT_BLOCK_MODEL</sub-model>
9     <sub-model>QLT_BLOCK_MODEL</sub-model>
10    <rules>
11      <relative-uri>config/symInverse.rul</relative-uri>
12      <ruleMode>FORWARD</ruleMode>
13    </rules>
14  </ontology>
15  <ontology>
16    <type>OWL</type>
17    <load-schemata>>false</load-schemata>
18    <reasoner-type>JENA_REASONER</reasoner-type>
19    <jena-reasoner>OWL_MEM</jena-reasoner>
20    <sub-model>INTERFACE_MODEL</sub-model>
21  </ontology>
22  <ontology>
23    <type>OWL</type>
24    <load-schemata>>true</load-schemata>
25    <reasoner-type>JENA_REASONER</reasoner-type>
26    <jena-reasoner>OWL_DL_MEM_RDFS_INF</jena-reasoner>
27    <sub-model>DOM</sub-model>
28  </ontology>
29 </ontology>

```

```

30 <type>RDF</type>
31 <load-schemata>true</load-schemata>
32 <reasoner-type>JENA_REASONER</reasoner-type>
33 <jena-reasoner>RDFS_DEFAULT</jena-reasoner>
34 <sub-model>LOGICAL_MODEL</sub-model>
35 </ontology>
36 ...
37 </wpps-config>

```

B) A Uniform Resource Identifier (URI) of the instantiated ontological model can be specified in one of two ways: with the element `uri` (a child element of `ontology`) or with the element `ontology-instance-ns-gen-base` (a child element of the root element `wpps-config`). For example:

```

<ontology-instance-ns-gen-base>http://www.dbai.tuwien.ac.at/proj/wpps/ontologies/
  inst-</ontology-instance-ns-gen-base>

```

or

```

<uri>http://www.dbai.tuwien.ac.at/proj/wpps/ontologies/bgminst1#</uri>

```

In the former case, the developer should specify a prefix which is supplemented with the random UUID (a universally unique identifier) and the symbol “#” at the end by WPPS. Thus, each ontological model will have a name space with a unique URI.

C) The developer should also specify the type of ontology instantiated in Jena: RDF or OWL. In spite of the fact that Jena is fundamentally an RDF platform [231], it provides numerous possibilities in realizing various ontology formalisms including OWL sublanguages (i.e., OWL Lite, OWL DL, and OWL Full to some extent). Representations of ontologies in Jena either as RDF graph or as OWL statements supported by the RDF graph differ from each other mainly by the API which Jena conveys for these types of ontologies and a set of applicable standard Jena reasoners. A type of the Jena ontology model is specified by the element `type` which can have either the value `OWL` or `RDF`.

D) The location of the schemata of the ontologies can be specified by the use of either elements: `alt-uri` or `alt-relative-uri`. For example:

```

1 <wpps-config>
2 <schemata>
3 <alt-load>
4 <schemata-name>STRUCT_BLOCK_GEOMETRIC_MODEL_ONT</schemata-name>
5 <alt-relative-uri>/ont/struct-block-geometric-model.owl</alt-relative-uri>
6 </alt-load>
7 </schemata>
8 ...
9 </wpps-config>

```

An `alt-relative-uri` parameter is defined relative to the root directory of the `tuwien.dbai.wpps.core` plug-in of WPPS. A `schemata` element is a child element of `wpps-config` and wraps all definitions of alternative locations of the ontology schemata (i.e., TBox of the ontologies) defined by parameter `alt-load` which in turn contains `schemata-name` and an alternative location.

E) A parameter `load-schemata` (see Listing 5.1), nested by the `ontology` element, contains the value of boolean type and defines a necessity to load relevant schemata (i.e., TBox) into memory and consider them in reasoning procedures.

F) In order to define the reasoner that should be applied to the ontology, the developer should specify a tool providing reasoning mechanisms using the tag `reasoner-type`. Presently, the WPPS framework only supports Jena reasoners. Therefore, the developer is required to assign

```
<reasoner-type>JENA_REASONER</reasoner-type>
```

and specify one of the standard reasoners Jena possesses by the parameter `jena-reasoner` (see Listing 5.1). Jena has two different sets of standard reasoners relevant to OWL- and RDF-based ontological models⁴. For example, there are reasoners such as `RDFS_SIMPLE`, `RDFS_DEFAULT`, and `RDFS_FULL` for RDF. As for OWL, they are mainly `OWL_DL_MEM_RDFS_INF` (a specification for OWL DL models that are stored in memory and require inference over RDFS statements), `OWL_DL_MEM_RULE_INF` (a specification for OWL DL models that are stored in memory and require the OWL rules inference engine for additional entailments), and `OWL_DL_MEM_TRANS_INF` (a specification for OWL DL models that are stored in memory and require the transitive inference for additional entailments). All standard Jena reasoners differ in the set of inference rules they use and optimization mechanisms applied for the derivation of new ontological statements. An OWL-based ontology model can also have `jena-reasoner` with the value `OWL_MEM`, which denotes that no reasoning mechanisms should be applied.

G) WPPS additionally supports the application of Jena inference rules [232]. Listing 5.2 demonstrates an example of Jena rules.

Listing 5.2: Example of Jena forward inference rules for symmetric and inverse object properties

```
1 @prefix rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
2 @prefix owl: http://www.w3.org/2002/07/owl#
3 [symProp: (?sp rdf:type owl:SymmetricProperty), (?a ?sp ?b)
4   -> (?b ?sp ?a)]
5 [inverseProp1: (?a ?sp1 ?b), (?sp1 owl:inverseOf ?sp2)
6   -> (?b ?sp2 ?a)]
7 [inverseProp2: (?a ?sp1 ?b), (?sp2 owl:inverseOf ?sp1)
8   -> (?b ?sp2 ?a)]
```

Rules should be placed into a separate file and referenced in the configuration file as it is demonstrated in Listing 5.1, lines 10–13. A maximum of one `rules` element is allowed in the `ontology` element. It specifies rule execution strategy (`ruleMode`) and location of the file with rules, which can be either absolute (`uri`) or relative (`relative-uri`) to the `tuwien.dbai.wpps.core` plug-in. There are four reasoning modes: `FORWARD`, `FORWARD_RETE` (forward reasoning using the standard RETE algorithm), `BACKWARD`, and `HYBRID` (which operates over forward, backward,

⁴WPPS supports reasoners listed in <http://jena.apache.org/documentation/javadoc/jena/com/hp/hpl/jena/vocabulary/ReasonerVocabulary.html> for RDF graphs and those mentioned in <http://jena.apache.org/documentation/javadoc/jena/com/hp/hpl/jena/ontology/OntModelSpec.html> for the OWL-oriented representation.

and hybrid rules). In the forward modes, the rule engine evaluates all the rules as soon as the underlying ontology is queried. The inference process stops when there is no rule which can be matched to the ontology. In the backward chaining mode, the rule reasoner uses a strategy similar to Prolog engines. Thus, when the ontology model is queried then the query is translated into a goal and the engine attempts to satisfy that goal by matching the backward chaining rules to any stored triples and by goal resolution against them. In the hybrid mode, the rule engine operates with all types of rules. The interested reader may refer to [232].

When using inference rules, WPPS forms a cascade of two reasoners where the rule reasoner corresponds to the top layer, whereas the standard reasoner provided by Jena refers to the bottom layer. Thus, the rule reasoner operates over the ontology incorporating the results of the standard reasoner [232, Sec. 6.9].

Objects Configuration

WPPS requires the developer to explicitly specify which types of objects are automatically instantiated by the framework and which are manually created in web page processing methods. Automatic generation is controlled by the parameter `create-in-ontology`, in which the developer lists relevant types of objects. These are mainly objects of the PM which are created together with their attributes and relevant relations during the PM instantiation phase by WPPS (see Section 5.3.3). Manually created objects are listed in the `support-in-ontology` parameter. In regards to these types of objects, WPPS creates all the necessary implementations, such as implementation providers for ontology instances, ontology instances implementations, instances adapters and adapters factories (see Figure 5.6). An example of configuring the objects instantiation is demonstrated in Listing 5.3.

Listing 5.3: Example of the configuration of objects, their attributes and relations in the WPPS configuration file

```
1 <wpps-config>
2   <create-in-ontology>
3     <!-- Instances in the BGM -->
4     <item>DOCUMENT_BLOCK</item>
5     <item>PAGE_BLOCK</item>
6     <item>VIEW_PORT_BLOCK</item>
7     <item>BOX</item>
8     <item>QNT_BLOCK</item>
9     <item>QLT_BLOCK</item>
10    <!-- Attributes in the QntBGM-->
11    <item>X_MIN</item>
12    <item>X_MAX</item>
13    <item>DRAW_ID</item>
14    <!-- Instances in the Interface Model -->
15    <item>HTML_IMAGE</item>
16    <item>HTML_TEXT</item>
17    <item>HTML_LINK</item>
18    <item>HTML_WEB_FORM</item>
19    <item>HTML_CHECKBOX</item>
20    <item>IM_CHECKBOX_GROUP</item>
21    <!-- Attributes in the Interface Model -->
22    <item>IM_TEXT_VALUE</item>
23    <item>IM_FONT_SIZE</item>
24    <item>IS_IM_CHECKED</item>
```

```

25 <!-- Relations in the Interface Model -->
26 <item>HAS_HTML_LINK</item>
27 <item>HAS_HTML_WEB_FORM_ELEMENTS</item>
28 </create-in-ontology>

30 <support-in-ontology>
31 <!-- Instances in the LM -->
32 <item>SEQUENCE</item>
33 <item>SEQUENCE_ITEM</item>
34 <item>TREE</item>
35 <item>TREE_NODE</item>
36 <!-- Relations in the QltBGM-->
37 <item>RCC8</item>
38 <item>OrthogonallyVisibleBlock</item>
39 </support-in-ontology>

41 <compute-by-request basis="quantitative">
42 <!-- Attributes in the QntBGM-->
43 <item>QNT_WIDTH</item>
44 <item>X_CENTER</item>
45 <!-- Relations in the QntBGM-->
46 <item>QNT_DISTANCE</item>
47 <item>QNT_DIRECTION</item>
48 <item>QNT_BORDER_DISTANCE_BB</item>
49 <!-- Relations in the QltBGM-->
50 <item>IR2D</item>
51 <item>PDirection</item>
52 </compute-by-request>

54 <composite-basic-dependence>
55 <!-- Relations in the QltBGM-->
56 <item>RCC8</item>
57 <item>OrthogonallyVisibleBlock</item>
58 </composite-basic-dependence>
59 ...
60 </wpps-config>

```

It is important to note that only basis classes (i.e., leaf elements in the class hierarchy) can be listed in these parameters.

Properties Configuration

For the object generated and supported by the WPPS platform, the developer can specify a set of attributes (datatype properties) and relevant relations (object properties) that should also be instantiated (see Listing 5.3). Relations mentioned in the parameter `create-in-ontology` are materialized during the physical model generation process. Usually, relations in the `support-in-ontology` parameter are either manually created by the user in WPP methods or instantiated due to the invocation of corresponding enriching procedures (*enrichers*) in WPP methods.

Instead of storing properties (attributes and relations) in the ontology, they can be computed “on-the-fly” each time the user requests them. A parameter `compute-by-request` defines a set of such properties. This possibility mainly concerns properties of the BGM. Quantitative properties assigned to `compute-by-request` can be computed based on other quantitative properties. For example, the width of the block is acquired from the endpoints of its vertical projection on abscissa and the distance between blocks is calculated based on the extreme points

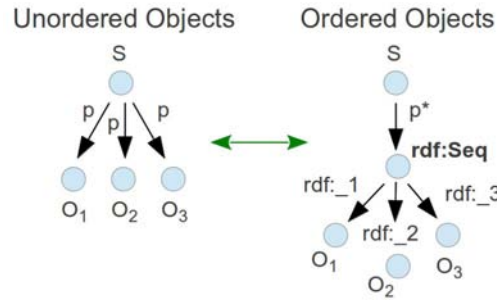


Figure 5.8: Two realizations of the “one-to-many” relationship

of blocks. Basic properties, for example, `X_MIN`, `X_MAX`, and `DRAW_ID`, cannot be computed based on other properties in the UOM and have to be computed during the PM generation based on the DOM tree and the CSS Object Model (CSSOM).

Qualitative properties computed upon request can be acquired in one of three ways: 1) based on quantitative information, 2) based on fundamental qualitative information, and 3) in cases of composite relations based on basic relations.

Qualitative properties dependent on quantitative properties require relevant information to be available in WPPS. For example, properties of the BGM can be computed based on the coordinates and spatial extension of blocks that require properties such as `X_MIN`, `X_MAX`, `Y_MIN` and `Y_MAX` to be stored in the ontology. The dependency on quantitative properties is denoted by the attribute `basis="quantitative"` of the `compute-by-request` parameter. A dependency on fundamental information is not implemented in WPPS. An example of this dependency is a definition of spatial relations presented in the UOM via Two-Dimensional Interval Relations with Centering (2DIRC) (see Section 3.10.4). In cases when composite properties are to be computed based on basic properties, the latter should be presented in the ontology or computable upon request. Whereas the basic property is a leaf node, we refer to the composite property as a non-leaf node in the tree of subsumption hierarchy of the same type of properties. For instance, examples of composite RCC8 relations (see Figure 3.3 on page 70 and 4.11 on page 117) are `PP`, `P-`, and `C`, and examples of basic ones are `PO`, `EQUAL`, and `TPP-`. This dependency can be specified with the element `composite-basic-dependence`.

A “one-to-many” relationship can be presented in the UOM as a set of statements in the form of subject-predicate-object or by means of an RDF container, such as a sequence. These two cases are illustrated in Figure 5.8, and can be specified in the WPPS configuration file by means of the `struct-one-to-many-relation` parameter as follows:

```

1 <wpps-config>
2   <struct-one-to-many-relation sub-model="STRUCT_BLOCK_GEOM_MODEL">
3     SEPARATE_STATEMENTS</struct-one-to-many-relation>
4   <struct-one-to-many-relation sub-model="DOM">IN_COLLECTION</struct-one-to-many-
5     relation>
6   ...
7 </wpps-config>

```

This configuration parameter is mainly related to instances of the BGM (its sub-models) and Extended DOM. For example, WPPS will use relationship `containsBlock` to relate a bounding block to the blocks contained in it for the BGM when using a set of statements and `containsBlocks` when utilizing an RDF container. Although the order of adding blocks into the bounding block is not important for the developer in the former case, it is very important in the latter case.

Fuzziness of Spatial Relationships

WPPS implements all the relations mentioned in Section 3.5. Consideration of the uncertainty in computing qualitative relations based on quantitative information in WPPS is compliant with the definitions presented in Section 3.10.2 for intervals and Section 3.10.3 for the geometric centers of intervals. Thus, relationships between blocks (be it topological, direction, or alignment relations) are translated into the relationships between their projections on abscissa and ordinate. This technique supports the proposition presented in Section 3.10.4 of using 2DIRC as a basis for computing the other spatial relations. Therefore, for the sake of the efficiency, WPPS computes qualitative spatial relations mainly based on the endpoints of the projections and geometric centers taking into account equations (3.20) for topology, (3.21) for the O-direction, (3.22) for the P-direction, and (3.23) for alignment.

In order to ensure the convenience in specifying fuzziness of spatial relationships by the developer, we decided to define membership function $\mu_0^-(x)$ as presented in (3.17) (on page 85) for endpoints of the intervals and the central point⁵. This approach also avoids the mutual presence of different disjoint relations in spite of the fact that the WPPS framework can deal with them (for example, with the disjoint qualitative relations stored in the instance of the UOM created manually or by the use of another tool). WPPS utilizes the same membership functions for both interval projections. Thus, when specifying the membership functions, the developer should only specify the endpoints (i.e., σ^- and σ^+ , of the equality intervals for the endpoints and geometric center of the block's projections).

For the $\sigma^\pm = \pm 0.4$ for all extreme points, $\xi = 0.5$, and $\varepsilon^\pm = 0$, the relevant configuration in the WPPS file will look as follows:

```

1 <wpps-config>
2   <qltbm-border-mu-type>INTERVAL</qltbm-border-mu-type>

4   <qltbm-left-border-mu>
5     <left-point>-0.4</left-point>
6     <right-point>0.4</right-point>
7   </qltbm-left-border-mu>

9   <qltbm-right-border-mu>
10    <left-point>-0.4</left-point>
11    <right-point>0.4</right-point>
12  </qltbm-right-border-mu>

14  <qltbm-center-mu>
15    <left-point>-0.4</left-point>
16    <right-point>0.4</right-point>
17  </qltbm-center-mu>

```

⁵It is important to note that according to Definition 3.16 (on page 84), $\mu_0^+(x) = \mu_0^-(-x)$.

```

19 <qltbn-border-nu>
20   <center>0.5</center>
21   <delta>0</delta>
22 </qltbn-border-nu>
23 ...
24 </wpps-config>

```

It is important to note that ξ and ε^\pm are not relevant at the moment due to the use of the membership functions of 0-neighborhoods with values $\{0, 1\}$, which represents the 0-neighborhoods (fuzzy sets) into crisp sets.

In spite of the limitations imposed upon the WPPS configuration file, the WPPS framework fully supports definitions of fuzziness in relations presented in Sections 3.10.2 and 3.10.3.

Relevant Web Page Area

The developer can define the area for CSS boxes of all *pages* of a *document* (i.e., a web page), relevant to the area specified which will be considered in the process of instantiating the PM. The area is chiefly determined by the parameter `location`, which can correspond to different structural elements and segments of a web page. The `location` parameter can have the value `ALL` and thus refer to the whole web page and all CSS boxes within it. It can also be related to the web browser's viewport and relevant CSS boxes either contained within the viewport (value `INSIDE_VIEW_PORT`) or overlapping with it (value `OVERLAPS_VIEW_PORT`). Furthermore, the `location` parameter can refer to an arbitrary area and therefore CSS boxes either contained within the specified area (value `INSIDE_AREA`) or overlapping with it (value `OVERLAPS_AREA`). An arbitrary area is defined with additional parameter `area`, which enables setting top-left and bottom-right points of the required segment according to coordinate system of a web page specified in Section 3.3.

An example below demonstrates a configuration of an arbitrary instantiation area in which all the CSS boxes contained in it will be considered by WPPS for their instantiation within the PM.

```

1 <wpps-config>
2   <location>INSIDE_AREA</location>

4   <area>
5     <top-left-x>80</top-left-x>
6     <top-left-y>200</top-left-y>
7     <bottom-right-x>1024</bottom-right-x>
8     <bottom-right-y>768</bottom-right-y>
9   </area>
10  ...
11 </wpps-config>

```

For the efficient generation of the PM, it is preferable to set the instantiation area with the smallest size possible.

Simplification

A simplification procedure is part of the PM generation process presented in Section 5.3.3 and activated by the boolean parameter `simplification`, which accordingly allows either values

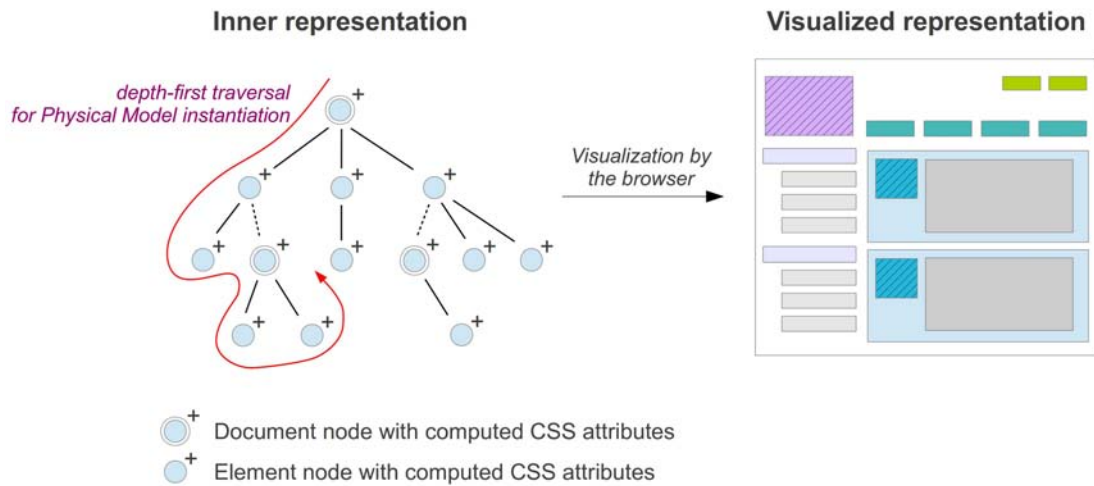


Figure 5.9: Web page instantiation operating over rendered DOM trees with computed CSS attributes

true or false. It removes those *visible visualized elements* (see Section 3.11.6) from the PM, which are fully overlapped and hidden by other visualized elements on a web page canvas and therefore are not visible by the user. Thus, a simplification parameter enables removing elements which do not play any role in the process of scanning and understanding a web page by the sighted user and therefore are not important in developing WPP methods based on visual cues.

5.3.3 Physical Model Instantiation

The process of Physical Model (PM) instantiation is applied for the rendered web page visualized on a web page canvas for the end user. Symbolically, we represent this procedure as a depth-first traversal over rendered DOM trees taking into consideration CSS attributes computed by the web browser engine (see Figure 5.9). Thus, for instantiating the PM (see Section 4.4) for a certain web page, the procedure analyses both DOM tree structures, types and attributes of elements and computed CSS attributes providing mainly quantitative information reflecting the position and size of CSS boxes, color, and font parameters for example [260]. The process of the PM instantiation has a linear time complexity that was practically confirmed in the experiment presented in Section 5.5 (see also Figure 5.22 on page 179). It is a procedure primarily dependent on the quantity of nodes populating DOM trees of a web page.

WPPS instantiates the UOM in two phases (see Figure 5.10): In the first phase, WPPS creates Jena ontology models with the required topology and reasoners applied and generate relevant implementors in accordance to the WPPS configuration. On the second phase, WPPS populates ontologies with relevant individuals (instances), attributes (datatype properties) and relations (object properties).

According to the WPPS configuration (see Section 5.3.2), the framework instantiates models such as the Extended DOM (or DOM*), Interface Model, the Structural Block-based Geometric

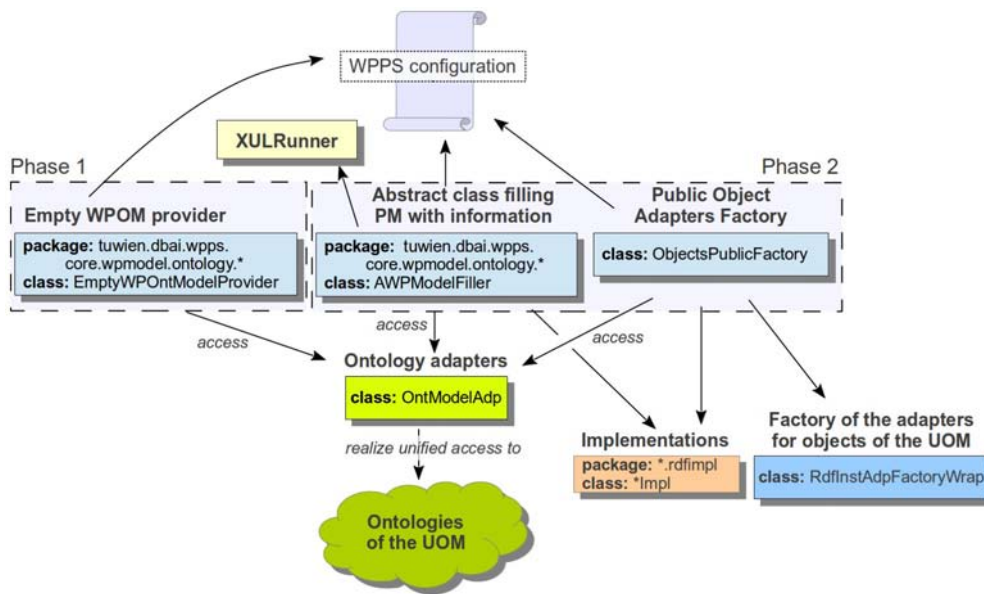


Figure 5.10: Diagram of classes used in the physical model instantiation process

Model (StrBGM), Quantitative Block-based Geometric Model (QntBGM), and Qualitative Block-based Geometric Model (QltBGM). Instantiation of the DOM* reflects structures of DOM trees, types of their elements, attributes of DOM elements and rendered CSSOM representing them in the form of one model as described in Section 4.4.1. Sub-models of the BGM are presented in Section 4.4.2. For building the StrBGM, the DOM tree along with the Browser Object Model (BOM) [136] are analyzed to identify the main structural elements of a web page mentioned in Section 3.4. For the quantitative description of the layout, objects of the CSSOM and the BOM such as *client rectangle* and *window* are leveraged. A coordinate system with the unit of measure are defined according to the Section 3.3. Furthermore, the block with its attributes is specified as presented in Section 3.4 (in particular, in Definition 3.5 on page 67). For the sake of efficiency, qualitative information is not instantiated during the process of PM instantiation and mainly generated in the QltBGM by means of enrichers which can be invoked in the WPPS methods. The block in the QltBGM is compliant with Definition 3.7 on page 68. Thus, generation of quantitative and qualitative spatial attributes and relationships are in accordance with the definitions presented in Chapter 3. During the instantiation of the Interface Model (see Section 4.4.3), information such as names of element nodes and their attributes as well as certain computed CSS attributes (e.g., *display*) are used to identify basic functional roles of interface objects and their types of layout described both by the markup language (e.g., UL elements are used to form unordered lists and TABLE elements are applied for table layouts) and CSS style sheets (e.g., an attribute `display=list-item` forms the list layout while `display=table` and `display=inline-table` are utilized for the table layout). It is important to note that the DOM* contains all the necessary information for materializing the StrBGM, QntBGM, and Interface Model, and therefore can be used as a source for instantiating these models.

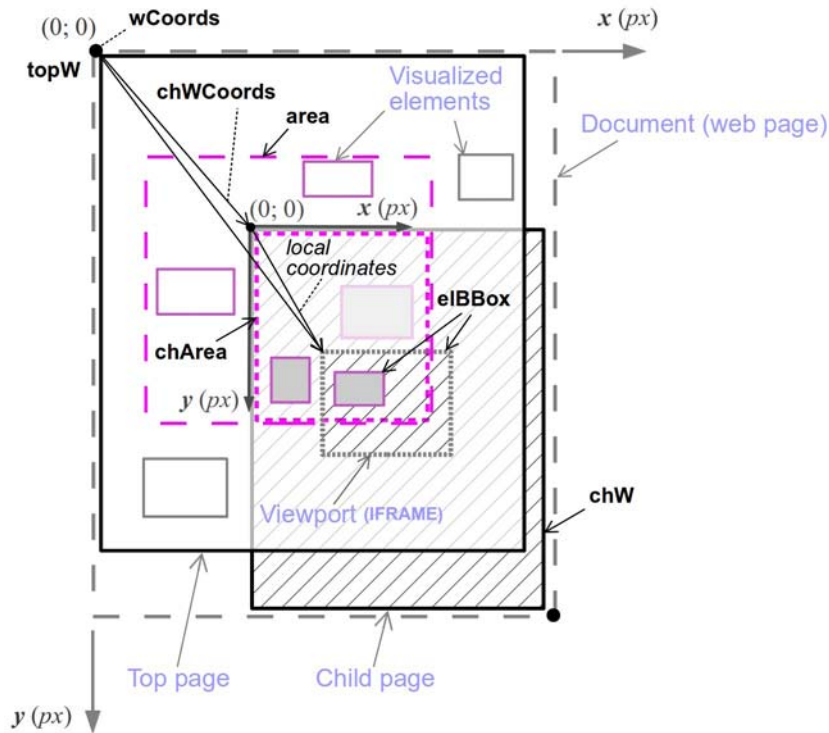


Figure 5.11: Web page objects in the process of the Physical Model (PM) instantiation

Algorithms 5.1, 5.2, and 5.3 corresponding to the second phase of the PM instantiation describe the process of the PM instantiation in more detail. They take into account CSS specifications and some peculiarities of the Java language leveraged to develop the WPPS framework. Some variables used in the generic algorithms are explained in Figure 5.11. In particular, `wCoords` are coordinates of the current *page w* (i.e., DOM window of the CSSOM and BOM). The variable `chW` is a child page (DOM window of the BOM) relative to the current one and `chWCoords` is its global coordinates (relative to the coordinate system of the top page `topW`). The variable `area` specifies an *instantiation area*. CSS boxes relevant to this area are considered to be instantiated in the UOM in accordance with the WPPS configuration. `chArea` is an instantiation area for the child page. Global coordinates of the extreme points of a CSS box are stored in the variable `elBBox` referring to the minimum bounding block. In the algorithms, the variable `locObj` defines a type of the instantiation area which can correspond to the whole web page (i.e., document), viewport, or specific area as well as relation of the CSS boxes to this area (i.e., overlapping or containment).

The algorithm `PMGenerationMain` is applied for empty instances of sub-models of the PM generated according to a specific WPPS configuration (see Section 5.3.2) and contains four main procedures:

1. Instantiation of the R-tree according to the fuzziness introduced in the WPPS configuration file. Leveraging dynamically generated implementations (for the R-tree) which, in particular,

Algorithm 5.1: PMGenerationMain

Input: A top DOM window (*topW*), i.e., *page*.
A type of the object which defines the instantiation area (*locObj*).
An instantiation area for the current DOM window (*area*).

- 1 *rTree* ← InitRTree();
- 2 *«Instantiate document based on topW in the instance of the BGM»*;
- 3 DOMTreeTraversal(*topW*, [0; 0], *locObj*, *area*, *rTree*); // see Algorithm 5.2
- 4 *«Compute painting order and layers»*;

Algorithm 5.2: DOMTreeTraversal

Input: A current DOM window (*w*).
Coordinates of the top-left corner of the DOM window (*wCoords*).
A type of the object which defines an instantiation area (*locObj*).
An instantiation area for the current DOM Window (*area*).
A spatial index (*rTree*).

- 1 *«Instantiate w in the instance of the BGM as a page»*;
- 2 *«Instantiate viewport of w in the instance of the BGM»*;
- 3 *docN* ← GetDOMDocumentNode(*w*);
- 4 *«Instantiate docN in the instance of the DOM*»*;
- 5 WrapTextNodes (*docN*);
- 6 *walker* ← GetDOMWalker (*docN*, *SHOW_ELEMENT*);
- 7 DepthFirstTraversal (*walker*, *w*, *wCoords*, *locObj*, *area*, *rTree*); // see Algorithm 5.3

specify RCC8 relations P and O necessary for the index.

2. Instantiation of the *document* corresponding to the top page (DOM window).
3. Invocation of the procedure `DOMTreeTraversal` to iterate over DOM trees of the current web page.
4. Computation of the painting order according to [260, App. E] and layers by means of the algorithm implemented in [70]. Instantiation of the painting order as attributes (i.e., datatype properties) *drawId* for blocks in the instance of the QntBGM.

The procedure `DOMTreeTraversal` recursively operates over DOM trees of a web page in the depth-first order and contains seven main procedures:

- 1–2. Instantiate the current DOM window as a *page block* and corresponding viewport in the instance of the BGM according to the WPPS configuration.
- 3–4. Instantiate a document node *docN* of the current DOM tree (which corresponds to the DOM window *w*) in the instance of the DOM* model.
5. When given the DOM tree with a document node *docN*, wrap all nonempty text nodes specified in Section 3.11.6 by supplementary elements with the name `WPPS-TEXTELEMENT`. However, certain types of web form elements such as `INPUT` and `TEXTAREA` are exceptions in this case. This procedure gives us a possibility to create an element node in the DOM

Algorithm 5.3: DepthFirstTraversal

Input: A TreeWalker of the Document Object Model Traversal (walker).

A current DOM window (w).

Coordinates of the top-left corner of the DOM window (wCoords).

A type of the object which defines an instantiation area (locObj).

An instantiation area for the current DOM window (area).

Spatial index (rTree).

```
1 n ← GetCurrentNode (walker) ;
2 if IsElement (n) then
3   el ← ToElement (n) ;
4   elBBox ← GetBoundingBox (el, w, wCoords) ;
5   if IsAcceptableElement (el, elBBox, area, locObj) then
6     <<Instantiate el in the instance of the BGM, DOM*, and Interface Model>>;
7     IndexInRTree (GetInstanceFromBGM (el), elBBox, rTree) ;
8     if IsAcceptableFrame (el, elBBox, area, locObj) then
9       chW ← GetChildDOMWindow (el) ;
10      chWCoords ← GetChildDOMWindowCoordinates (chW, elBBox) ;
11      chArea ← GetChildArea (area, chWCoords) ;
12      DOMTreeTraversal (chW, chWCoords, locObj, chArea, rTree) ;           // see
          Algorithm 5.2
13    if IsAcceptableChildren (el) then
14      foreach chN ∈ GetChildNodes (walker) do // go through all child elements
          utilizing the API of walker
15        DepthFirstTraversal (walker, w, wCoords, locObj, area, rTree) ; // direct
          recursion
16    SetCurrentNode (walker, el) ;
```

tree which serves as minimum bounding blocks and thus, obtain precise coordinates of text nodes. This is a practical solution for XULRunner 1.9.2 and this version of the Firefox platform does not provide coordinates for text nodes.

6–7. Initialize the DOM walker traversing over element nodes and invoke the procedure `DepthFirstTraversal` with specified parameters.

The `DepthFirstTraversal` procedure populates the ontologies with relevant objects and contains the following main procedures:

1–2. Acquiring a current node from the DOM walker and considering only those which are element nodes (a DOM walker provided by the Mozilla XULRunner 1.9.2 also navigates through document elements).

3–4. Obtain an element node specific interface for a given node `n` and acquire global coordinates `elBBox` of its minimum bounding block. Leveraging the XULRunner, the variable `elBBox` is computed based on the *bounding client rectangle* (which contains coordinates relative to the viewport and acquired by means of the interface `nsIDOMNSElement` of the current element `el`), the local position (relevant to the current DOM Window, i.e., page block)

of the relevant viewport (acquired by means of the interface `nsIDOMWindow`), and global coordinates `wCoords` of the current DOM window `w`.

5. Check if the element is located on the first quadrant of the *page canvas* and whether it is relevant to the instantiation area according to the WPPS configuration.
- 6–7. Instantiation of the element `e1` in the PM and indexing it in the R-tree.
8. Check if a current element `e1` is a frame and if it overlaps with the instantiation area.
- 9–10. Acquisition of the child DOM window `chW` related to the frame element `e1` (interface `nsIDOMNSHTMLFrameElement` of `XULRunner` is used by WPPS for this) and its global coordinates (`chWCoords`). The position of the child page block is computed based on the global position `e1BBox` of the element `e1` and local position of the viewport formed by the frame element `e1` within the child page `chW`.
- 11–12. Compute child area relevant to the child page, and for the acquired child window invoke `DOMTreeTraversal` in recursion.
13. Check if the child elements can be visible (i.e., a CSS attribute `display≠none`); the current element should be a document or element node.
- 14–15. Walk through the child elements of `e1` and invoke `DepthFirstTraversal` for each of them.
16. Set the `e1` element as a current node for the DOM walker.

The PM instantiation also includes the process of the model simplification which is invoked according to the WPPS configuration. It removes all elements which are in relation P (see Section 3.6) with other elements and have smaller draw id (i.e., were drawn earlier) while primarily leveraging SPARQL.

Considering various web pages from the WPPS-HTML-DS1 dataset [72], the reduction in object amounts in the instance of the PM due to the “simplification” procedure is 26% on average, whereas the first quartile is 22% and the third quartile is 31%.

5.3.4 WPPS API

The API of the WPPS framework provides all the necessary functionalities for the developer in realizing web page processing (WPP) methods. It is implemented on top of the *adapter layer* (see Section 5.3.1), treating ontological concepts as Java objects and being independent from particular configurations of the UOM.

When realizing WPP methods, a developer has access to instances of class `WPUMethodState`, which contains all information relevant to the current state of the WPPS framework. She also has access to `IEAPI` which provides the necessary objects for querying and enriching an instance of the UOM. A Java object of type `WPUMethodState` gives access to relevant ontologies (which can be queried by means of Jena API and SPARQL), the web browser (i.e., Firefox 3.6) and therefore rendered DOM trees and CSSOMs, as well as current WPPS configuration. By crawling

web pages and applying certain methods, the developer can obtain different states which can be accessed by these methods and thus, conduct comparative analysis of different web pages. An object `IEAPI` available for the developer can be used for acquiring integrated enrichers and basis API, represented by the interface `IIEBasisAPI`. The latter is the most important Java object implementing different functions for interacting with the UOM according to the WPPS configuration.

The functions of the basis API can be split into 3 main groups: 1) selectors, 2) processing functions, and 3) statistical functions.

1. Selectors are those functions that allow the selection of a specified subset of objects from the instance of the UOM. Extracted ontological individuals are adapted by the corresponding Java objects by the use of the bridged adapted design pattern and wrapped by a Java object of type `IResults` representing them as a sequence. Selection can be performed based on the type of object (e.g., “image,” “box,” or “html link”), predicate specified, or SPARQL query. Listing 5.4 demonstrates the use of a selector with a predicate specified targeting objects of specific types. Furthermore, the object contained or intersecting specified area can be selected; in this case R-tree is involved for efficiency. Listing 5.5 presents an example of a selector which extracts objects contained within the specific area and satisfying the specific predicate.

Listing 5.4: Example of using a selector function for extracting images and text elements from the instance of the UOM and which has a width longer than x

```

1 IResults rez = api.getObjectsByType( // select all blocks which are images and
   text elements satisfying the predicate specified
2   new IIEPredicate() {
3     @Override public Boolean apply(IInstanceAdp avar) {
4       return avar.as(IQntBlock.class).getWidth()>x; // request the interface of
   the 'quantitative' block and check the width
5     } }
6   , IHtmlImage.class
7   , IHtmlText.class);

```

Listing 5.5: Example of using a selector function for extracting blocks contained within the specific rectangular region *region* by means of R-tree and which has an area larger than x

```

1 IResults rez = api.getObjectsContainedInArea(region // the containments is tested
   by means of the R-tree
2   , new IIEFilter() {
3     @Override public EFilterResult apply(IQntBlock avar) {
4       return (Rectangle2DUtils.area(avar.getArea())>x) // check area
5         ?EFilterResult.ACCEPT:EFilterResult.REJECT;
6     } } );

```

2. Processing functions were designed to process objects acquired from the instance of the UOM by means of selector functions. The following functions are available when treating a result collection of objects as a set: intersection, union (see line 11 in Listing 5.7), and filtering by the predicate. Moreover, functions of this group make it possible to order elements as well as group result objects into subsets (see Listing 5.6), split the result into sequences (see Listing 5.7), and form trees and grids using the predicates specified by the developer.

Listing 5.6: Example of using a processing function for grouping results *rez1* into sets with similar foreground color

```
1 IResults rezSameFGColorGroups = api.groupInSets(rez1
2 , new IIEPredicate2() {
3   @Override public Boolean apply(IInstanceAdp avar1, IInstanceAdp avar2) {
4     // cast types of avar1 and avar2 to IHtmlElement and compare foreground colors
5     :
6     return avar1.as(IHtmlElement.class).getForegroundColor()
7     .equals(avar2.as(IHtmlElement.class).getForegroundColor());
8   } } );
```

Listing 5.7: Example of using a processing function for grouping results into sequences according to the order of elements in the result *rez1*; each pair of adjacent elements in the acquired sequences are in relation `EAST_ORTHOGONAL_VISIBLE_BLOCK_OF` with each other

```
1 IResults rezSameHorizontalDirectionGroups = api.groupInSeq(rez1
2 , new IIEPredicate2() {
3   @Override public Boolean apply(IInstanceAdp avar1, IInstanceAdp avar2) {
4     IQltBlock qltb1 = avar1.as(IQltBlock.class); // cast the type of avar1 into
5     the 'quantitative' block
6     IQltBlock qltb2 = avar2.as(IQltBlock.class);
7     // check presence of relation EAST_ORTHOGONAL_VISIBLE_BLOCK_OF between qltb2
8     and qltb1:
9     return qltb2.hasRelation(qltb1
10    , EBlockQltRelation.EAST_ORTHOGONAL_VISIBLE_BLOCK_OF);
11  } } );

11 IResults rez2 = api.union(rezSameHorizontalDirection, rezSameVerticalDirection);
```

3. Statistical functions provide means for computing aggregated values, such as mean, median, variance, minimal and maximal value, over a set of objects and set of pairs of adjacent objects in the result sequence. The latter is useful for computing some characteristics regarding relationships between adjacent objects in the result collection, such as average spatial distance between neighboring elements.

Listing 5.8: Example of using a simple statistical function for computing average distance between pairs of adjacent objects in the horizontally oriented sequences from Listing 5.7

```
1 final List<Double> distAvgArr = new ArrayList<Double>(
2   rezSameHorizontalDirectionGroups.size()); // array of distances

3 api.forEach(rezSameHorizontalDirectionGroups // iterate over the horizontally
4   oriented sequences of objects
5 , new IIEProcedure() {
6   @Override public void apply(IInstanceAdp avar) {
7     distAvgArr.add(
8     api.avgSeqPairs(avar.as(IResults.class) // get average distance between
9     pairs of adjacent objects in avar
10    , new IFunction2<IInstanceAdp, Double>() {
11      @Override public Double apply(IInstanceAdp avar1,
12      IInstanceAdp avar2) {
13        // cast the type of avar1 into the 'quantitative' block and request
14        distance between avar1 and avar2:
15        return avar1.as(IQntBlock.class)
16        .getRelationAsDouble(avar2, EBlockQntRelationType.QNT_DISTANCE);
17      } } )
18    )
19  } } )
```

```
15     );  
16 } } );
```

It is also important to mention class `ObjectsPublicFactory` providing methods for instantiating objects within the UOM (and particularly within the LM).

5.3.5 WPPS GUI

A GUI of WPPS (see Figure 5.12) enables users and developers to apply different web page processing methods and visualize results. A WPPS GUI contains versatile visual components, provided by the ATF project, for visualizing and modifying the DOM tree and CSSOM. It also contains a component (Ontologies Graph view) for visualizing the UOM by means of graph diagrams (see Figure 5.13) and interactive visual synchronization with a web page rendered within the web browser.

5.4 Developing Methods by Means of the WPPS Framework

Leveraging the API provided by the WPPS framework, the developer can realize methods for web objects identification, web page understanding, and web information extraction. We believe it is important to take into account activities proposed in Section 5.4.1 which help in structuring the process of developing Web Page Processing (WPP) methods. In Section 5.4.2, we also give some example of wrappers developed within the WPPS framework.

5.4.1 WPPS Methods Development Life Cycle

When developing a WPP method, the developer can follow any software development model she finds the most appropriate, be it waterfall model or spiral model. This is dependent both on the habit and the complexity of a method to be implemented. In spite of this, we find it important to consider the following sequence of phases in the WPPS methods development life cycle:

1. **Analysis of relevant features:** The objective of this phase is to conduct an analysis of features of the object to be identified or processed (e.g., in case of web page segmentation) and features of relevant objects of a web page while taking into account different aspects of web page representation, (i.e., the DOM* reflecting the structure of a web page's source code), BGM (describing spatial configurations and relationships between blocks), and Interface Model (introducing different functional roles of web page elements). For example, in regards to the main content identification, it is usually located in the middle of a web page and occupies the majority of the web page canvas. Furthermore, the major part of the text has the same font, style, color, and so on.
2. **Design:** This consists of specifying configuration of the WPPS framework (see Section 5.3.2) and defining the approach for web page processing. The WPPS configuration is provided to the framework as an XML file listing all necessary features which should be supported by the system and modes of their computation.

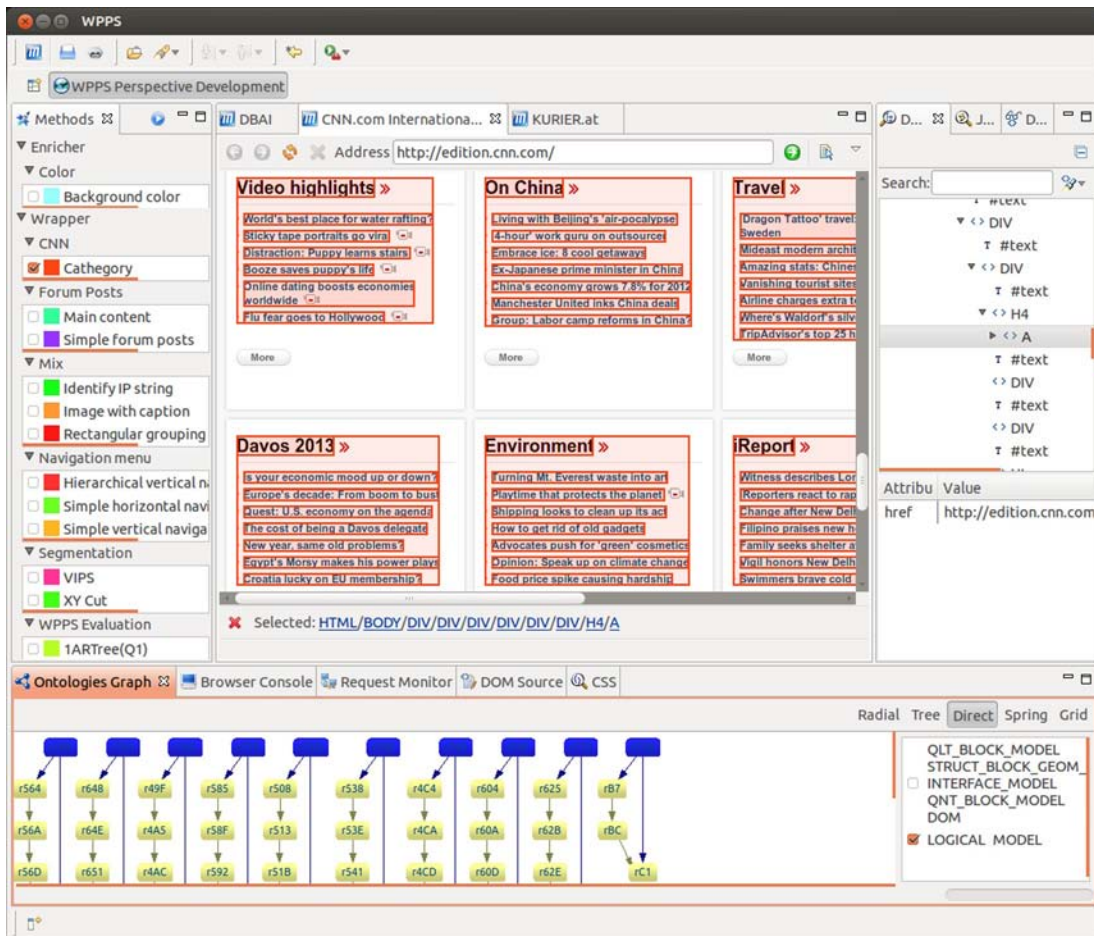


Figure 5.12: Screenshot of the WPPS GUI. The figure demonstrates the result of applying a wrapper “Category” for a web page edition.cnn.com

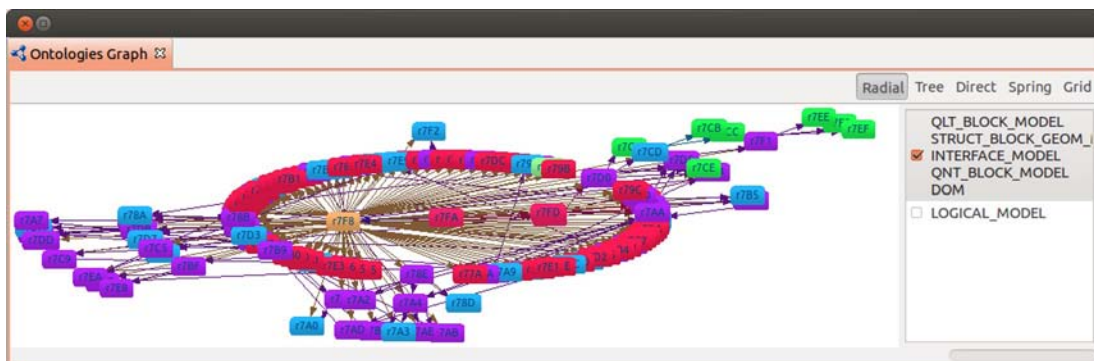


Figure 5.13: Screenshot of the Ontologies Graph view of the WPPS GUI. The figure shows the result of instantiating the PM for a web page www.dbai.tuwien.ac.at

3. **Development:** This is an implementation of the proposed approach in the WPPS framework using the WPPS API (see Section 5.3.4).
4. **Testing:** In this phase, effectiveness, efficiency, and robustness of the method are evaluated. The most common metrics used in the field of Web Page Processing (see Section 2.3) are precision and recall [68, 163, 169, 177, 305] adopted from the area of Information Retrieval.
5. **Integration:** At this stage, the method is integrated into other external systems. This integration primarily refers to providing the external system with understandable representation of a web page's LM. It can be achieved, for example, by leveraging the Linked Data approach and annotating objects of the UOM with concepts known by the external system. In this thesis, the integration is mainly concerned with modeling the structural data within the instance of the LM (see Section 4.5), which is used together with the PM to instantiate the MANM for enhancing accessibility of a web page (see Section 6.3).
6. **Maintenance:** This phase is related to correcting the developed method when it is necessary to improve its productivity or in case of deterioration of its effectiveness. The latter can be due to substantial changes within the interface of target web pages and their design and therefore an unexpected presence of different spatial configurations and features which were not taken into account during the *analysis phase*.

In regards to the *design phase*, we need to distinguish rule/heuristics-based approaches [62, 91, 305] and machine learning approaches [138, 186]. The first group depends on a set of predefined rules which can be web- or domain-specific. These manually constructed rules and heuristics usually reflect semantics hidden in spatial relationships between web page elements (e.g., [305]). Machine learning approaches focus on automatically deriving a model which describes specific aspects or features of web pages (e.g., [121, 138, 145, 186]). WPPS does not provide any specific interface for building rules or classifiers. Instead, the framework provides the interface for interacting with the UOM as well as querying and populating it with relevant objects. Examples in Section 5.4.2 demonstrate the use of the WPPS API for realizing heuristic-based approaches. Section 5.6 describes the use of WPPS in the development of a feature computation tool ObjIdent in the approach based on machine learning for basic web object identification.

Regarding the *development phase*, a method realizing a certain approach proposed during the design phase should be declared in the file `wpps-methods-config.xml` and implemented in the Java language in the corresponding plug-in fragment hosted by the WPPS plug-in `tuwien.dbai.wpps.core` according to the Eclipse RCP architecture. The XML file with method declarations should contain the following information (see example in Listing 5.9):

- An `id` of the method should be specified; it is a string for the unique identification (and can be a universally unique identifier, UUID).
- `gen-type=INTERNAL` denotes that a WPP method is either part of the WPPS framework (i.e., provided by default with the release of WPPS) or provided by means of a plug-in fragment. At the moment, only this value of the parameter is allowed.

- A parameter `type` specifies the type of method consisting of either `WRAPPER`, `ENRICHER`, or `CRAWLER`. A wrapper is a WPP method which populates an instance of the LM with assertions based on a web page's PM (where the PM is automatically instantiated by WPPS, see Section 5.3.3). An enricher gives the possibility to enrich the PM with additional information necessary for the analysis. A web crawler in WPPS is a WPP method which can navigate through web pages and conduct a cross-page analysis.
- `major-name`, `minor-name`, and `description` convey a generic name (or category) of the method, its precise name, and description respectively.
- A parameter `java-class` has a full name of a Java class implementing the WPP method. Thus, whether integrated into the system by default or by means of a plug-in fragment, a corresponding WPP method can be found by WPPS using this parameter.

Listing 5.9: An example of a method declaration in the file `wpps-methods-config.xml`

```

1 <wpps-methods-config>
2   <method id="horNavMenu"
3     gen-type="INTERNAL"
4     type="WRAPPER"
5     major-name="Navigation_menu"
6     minor-name="Horizontal_navigation_menu"
7     description="Extraction_of_basic_horizontally_oriented_navigation_menu">
8     <java-class name="tuwien.dbai.wpps.exmplmethods.navmenu.BasicHorNavMenuWrapper
9       "/>
9   </method>
10  ...
11 </wpps-methods-config>

```

A web crawling is currently limited in the WPPS and will be developed in further work on WPPS. A comparative analysis of web pages will be possible due to the generation and full control over different states of instances of the UOM provided by Java objects of type `WPUMethodState` (see Section 5.3.4).

Java classes implementing certain WPP methods should subclass specific classes provided by WPPS. For example, a method of type “wrapper” (i.e., `type=WRAPPER`) should be a subclass of `AWPUWrapper` and implement class methods with signatures and return types `List<IResults> _extractResults()` and `void _dumpIntoLM(List<IResults>)`. The former class method is used for realizing an extraction mechanism for the PM and representing the acquired results in a collection of type `List<IResults>`. The latter class method is leveraged to serialize the acquired results within the instance of the LM by means of a class `ObjectsPublicFactory` and a method `<U extends IInstanceAdp> U convertTo(Class<U>)` of the class `IResults`.

5.4.2 Basic Examples of WPPS Methods

By leveraging the WPPS API and adapters, one can query the UOM and treat individuals as Java objects abstracting from the ontologies and reasoners. In this section, we consider two examples of wrappers in detail: extraction of the horizontal navigation menu (see Example 5.1) and extraction of images with their captions (see Example 5.2). As we can see, both examples

operate merely on a web page’s visual representation and with visually perceptible features while taking into account assertions in the instances of the BGM and Interface Model.

Example 5.1 (Extraction of the horizontally oriented navigation menu). *To demonstrate, we present a wrapper for extracting a horizontally oriented navigation menu. The wrapper is implemented using the WPPS API, and its source code demonstrating the extraction process (a class method `_extractResults()`) and instantiation of the acquired results in the LM (a class method `_dumpIntoLM(List<IResults>)`) is presented in Listing 5.10. We define a navigation menu as a sequence of menu items (line 44) with spatial relations “east-orthogonal-visible-block-of” (lines 11–13 and 35) and “bottom-aligned-with” (line 36) defined between them (lines 31–37). Each menu item is a link (line 24) containing nonempty textual elements (line 25). The expected location is on the top of a web page in a rectangular area with the height of 250 px (lines 17–18).*

Listing 5.10: The source code (in the Java language) of a WPPS-based wrapper for identifying a horizontally oriented navigation menu

```

1 public class BasicHorNavMenuWrapper extends AWPWrapper {

2
3 public BasicHorNavMenuWrapper(AWPMethodDescription description) {
4     super(description);
5 }

6
7 @Override protected List<IResults> _extractResults() {
8     final IIEBasisAPI api = super.getIEAPI().getIEBasisAPI(); // get basis WPPS API

9
10 // 1. enrich a web page’s PM with a supplementary relation
11     EAST_ORTHOGONAL_VISIBLE_BLOCK_OF only for html links
12 AsymmetricOrthogonalVisibilityEnricher e = getIEAPI().getEnricher(
13     AsymmetricOrthogonalVisibilityEnricher.class);
14 e.init(IHtmlLink.class, EBlockQltRelation.EAST_ORTHOGONAL_VISIBLE_BLOCK_OF);
15 e.enrich();

16
17 // 2. get all links, which contain nonempty textual elements, from the top part of
18 // a web page
19 IWebDocumentBlock doc = api.getObjectByType(IWebDocumentBlock.class); // get
20 // document block
21 Rectangle2D area = doc.getTopWebPage().as(IQntBlock.class).getArea(); // get top
22 // page block and request for the occupied area on a web page canvas
23 area.yMax = 250; //px

24
25 IResults res = api.getObjectsContainedInArea(area // leverage R-tree to get
26 // objects from the top area of a web page
27 // consider only those elements which are links containing nonempty text elements
28 // , new IIEFilter() {
29 //     public EFilterResult apply(IQntBlock v) {
30 //         if (v.canAs(IHtmlLink.class)
31 //             && v.as(IHtmlLink.class).getString().length() > 0)
32 //             return EFilterResult.ACCEPT;
33 //         else return EFilterResult.REJECT;
34 //     }
35 // }); // apply, new IIEFilter, getObjectsContainedInArea

36
37 // 3. join objects by relations EAST_ORTHOGONAL_VISIBLE_BLOCK_OF and
38 // BOTTOM_ALIGNED_WITH
39 res = api.groupInSeq(res, new IIEPredicate2() {
40     public Boolean apply(IInstanceAdp v1, IInstanceAdp v2) {

```

```

33   IQltBlock b1 = v1.as(IQltBlock.class);
34   IQltBlock b2 = v2.as(IQltBlock.class);
35   return b2.hasRelation(b1, EBlockQltRelation.EAST_ORTHOGONAL_VISIBLE_BLOCK_OF)
36      && b2.hasRelation(b1, EBlockQltRelation.BOTTOM_ALIGNED_WITH); // these
      relations are acquired by WPPS according to its configuration
37 } } ); // apply, new IIEPredicate2, groupInSeq

39 return (List) res.getResultContent(); // return list of sequences of objects (i.e
    ., navigation menus) formed by the method 'groupInSeq'
40 } // _extractResults

42 @Override protected void _dumpIntoLM(List<IResults> results) {
43 for(IResults r :results) {
44   ISequence seq = r.convertTo(ISequence.class); // instantiate an identified
      navigation menu as a sequence in a web page's LM
45   addLogicalStructure(seq); // report to the framework and other WPPS methods that
      this wrapper has created an object
46   // make a sequence of navigation items to highlight in the WPPS GUI
47   r.convertTo(IBoundingBox.class); // instantiate navigation menu object as a
      bounding block
48   r.convertTo(IQntBlock.class); // compute coordinates for the bounding block
      which is minimum bounding block for the contained elements
49 } } // for, _dumpIntoLM

51 } // BasicHorNavMenuWrapper

```

In the implementation presented in Listing 5.10, we invoke a default enricher for the relation `EAST_ORTHOGONAL_VISIBLE_BLOCK_OF` (lines 11–13), but do not apply it for `BOTTOM_ALIGNED_WITH`. This fact requires the developer to specify corresponding settings in relevant WPPS configuration file. Regarding the former relation in particular, the relation type `OrthogonallyVisibleBlock` should be defined in the parameter `support-in-ontology` (which means that WPPS should generate necessary *implementors* to support this type of relations). However, the relation `BOTTOM_ALIGNED_WITH` and corresponding relation type `Alignment` should be set in the parameter `compute-by-request` with attribute `basis="quantitative"` (which means that relations of this type should be computed “on-the-fly” utilizing quantitative information) as it is described in Section 5.3.2.

To obtain better precision, the wrapper presented in Example 5.1 can have additional constraints regarding the distance between menu items (e.g., the maximum distance between menu items should be less than the width of the widest menu item and should not differ much), number of items (e.g., from 3 to 20), font similarity, and number of lines (e.g., at most two lines). Results of applying such an enhanced wrapper in WPPS is presented on the screenshots in Figure 5.14.

Example 5.2 (Extraction of images with their captions). *To demonstrate, we represent a caption as a textual element (line 18 in Listing 5.11) which overlaps with the area of an image (line 18) extended downward (lines 16–17) and has the biggest font size among the candidates (lines 23–27). Example of the application of the wrapper presented in Listing 5.11 is demonstrated on screenshots in Figure 5.15.*

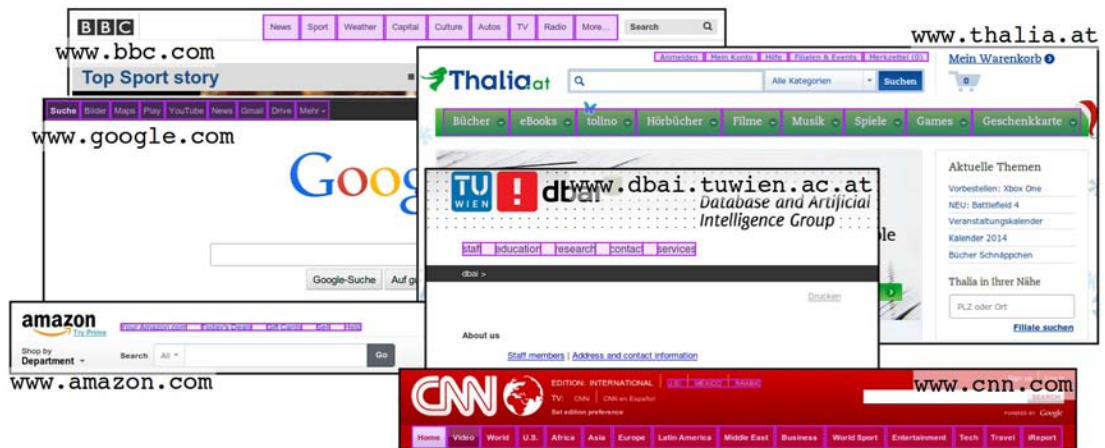


Figure 5.14: Screenshots of results of identifying a horizontally oriented navigation menu with the use of WPPS

Listing 5.11: The source code (in the Java language) of a class method `_extractResults` of a WPPS-based wrapper for identifying images with their captions. (imageMinArea=5000, captionIndent=20)

```

1 final IIEBasisAPI api = getIEAPI().getIEBasisAPI();
2 final List<IResults> rez = new LinkedList<IResults>();

4 // 1. get all images with the area at least 'imageMinArea' (px^2)
5 IResults imgRes = api.getObjectsByType(
6     new IIEPredicate() {
7         @Override public Boolean apply(IInstanceAdp avar) {
8             return Rectangle2DUtils.area(avar.as(IQntBlock.class).getArea()) >=
9                 imageMinArea;
10        } } // apply, new IIEPredicate
11    , IHtmlImage.class);

12 //2. find relevant caption for each image
13 api.forEach(imgRes, new IIEProcedure() {
14     @Override public void apply(IInstanceAdp img) {
15         // 2.1 get all text elements which intersect a specific area
16         Rectangle2D area = img.as(IQntBlock.class).getArea();
17         area.yMax += captionIndent; // increase the area changing the yMax by '
18         captionIndent' (px)
19         IResults txts = api.getObjectsIntersectingArea(area, null, IHtmlText.class);
20         // use R-tree to get text elements intersecting the specific area; null --
21         // additional filter is not used

22         // 2.2. find a caption (text element with the biggest font size)
23         if (txts.size() > 0) { // if at least one text element was selected
24             // order a list of the acquired text elements by their font size in the
25             // descending order
26             txts = api.orderBy(txts, new IFunction<IInstanceAdp, Comparable<?>>() {
27                 @Override public Comparable<?> apply(IInstanceAdp avar) {
28                     return avar.as(IHtmlText.class).getFontSize();
29                 }
30             }, -1); // <0 is descending order, >=0 is ascending order
31             rez.add(api.toResults(Lists.newArrayList(txts.get(0), img))); // add the
32             first element from the list of text elements (caption) and an image into

```



Figure 5.15: Screenshots of results of extracting images with relevant captions with the use of WPPS

```

28         }
29     else
30         rez.add(api.toResults(Lists.newArrayList(img))); // add only image into the
31     } } ); // apply, new IIEProcedure, forEach
32 return rez;

```

To present the possibility of applying methods from the field of Document Processing in the field of Web Page Processing that was noted in Section 2.3.2, we implemented segmentation algorithms presented in [106] based on the XY-cut technique and intended for document images in the WPPS framework. The results of applying this method on contemporary web pages is demonstrated in Figure 5.16.

The developer, when leveraging WPPS, has the possibility to take into account spatial configurations and different geometric peculiarities of web page elements. This permits her to create methods which are applicable on a significantly wider range of web pages in contrast to methods operating on the technical level of a web page (i.e., on the source code and DOM tree levels). This supports our findings regarding the properties of different forms of web page representation presented in Section 2.4.8.

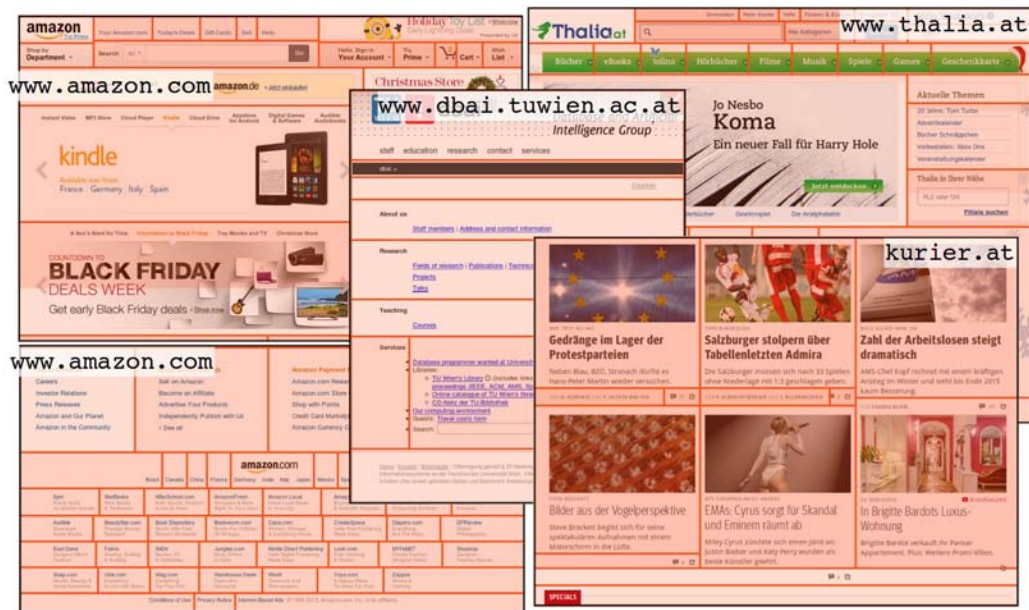


Figure 5.16: Screenshots of results of leveraging a web page segmentation based on the XY-cut algorithm presented in [106] with the use of WPPS

5.5 WPPS Evaluation

In this section, we conduct a comparative analysis of the efficiency of leveraging different WPPS configurations and joint application of declarative and object-oriented approaches by example of basic interactions with the instance of the UOM. The intention of this experiment is to demonstrate the feasibility of the approach integrating declarative and object-oriented paradigms by example of WPPS framework as well as to consider different aspects of this approach in its application to web page processing.

5.5.1 Goal and Objectives

The **goal** of this empirical research is to study the efficiency of the WPPS framework for different configurations related to the object-oriented abstraction (see Section 5.2) as well as to provide the developer, based on the data acquired from the experiment, with information necessary for choosing the most appropriate configuration for realizing a web page processing method.

According to the aim specified, we pose the following **objectives**:

- Select the most relevant WPPS parameters for the analysis and specify their values (see Section 5.5.2).
- Choose representative basic queries for the UOM (see Section 5.5.2).
- Develop corresponding wrappers according to the identified configurations and queries specified (see Section 5.5.3).

- Collect data related to the performance of different wrappers and conduct their analysis (see Section 5.5.4).
- Form conclusions regarding different configurations and their purpose (see Section 5.5.5).

5.5.2 WPPS Parameters and Queries

Parameters to be Considered

According to the goal specified, we consider WPPS parameters which define a realization of the UOM and influence the modes of interacting with the UOM and methods of acquiring required information worthy of leverage in the experiment (Detailed description of the WPPS parameters can be found in Section 5.3.2.) These are mainly:

- A parameter `type` specifying a realization of the ontology in Jena either based on the RDF graph (`type=RDF`) or a set of statements in Jena OWL model (`type=OWL`).
- A type of the standard reasoner provided by Jena which is set by the parameter `jena-reasoner`. We believe it is important to consider both RDFS and OWL DL reasoners, such as Jena's `RDFS_SIMPLE` (with simplified, high performance rules) for reasoning over RDF(S) model and `OWL_DL_MEM_RULE_INF` for reasoning over OWL statements grasping the full expressive power of the UOM [231, 232].
- Inference rules which should be applied over the relevant ontology and which are determined by the parameter `rules`.
- Parameters configuring the mode of instantiating information in the UOM. Mainly, we refer to the parameter `create-in-ontology` (which requires information such as objects, their attributes, and relations to be automatically instantiated in the UOM during the Physical Model generation phase by WPPS) and the parameter `support-in-ontology` (which requires manual instantiation of relevant information in the web page processing methods, for example, by means of so called *enrichers*).
- Parameters identifying the mode of acquiring information from the UOM. For the dynamic computation “on-the-fly,” it is `compute-by-request` with the attribute `basis="quantitative"` (i.e., computation of specified attributes of objects and relations based on basis quantitative information) and `composite-basic-dependence` (i.e., computation of the specified composite relations based on basic relations).

Basic Queries to be Evaluated

Based on the WPPS API proposed (see Section 5.3.4), it is important to consider the following two basic queries:

Definition 5.1 (Query \mathcal{Q}_1). Given ontology $\mathcal{O} = \langle C, H_C, R_C, H_R, I, \iota, R_I, A \rangle$ as it is specified in Definition 2.1 (on page 48), object $b^* \in I$ (a set of objects, i.e., instances of classes), and relation $\rho \in R_C$ (a set of relations), a query $\mathcal{Q}_1(\rho, b^*)$ returns the set B such that $\forall b_i \in I[\rho(b^*, b_i) \rightarrow b_i \in B]$.

Algorithm 5.4: A generic algorithm for the scenario realizing the query Q_1 (see Definition 5.1)

Input : API of the WPPS framework (wppsAPI).

Output : Blocks from the central area of a web page (blocks).

```
1 area ← ComputeArea (wppsAPI);
2 EnrichPMWithRCC8 (wppsAPI);
3 blocks ← SelectBlocks (area, wppsAPI);
```

Algorithm 5.5: A generic algorithm for the scenario realizing the query Q_2 (see Definition 5.2)

Input : API of the WPPS framework (wppsAPI).

Output : Blocks from the central area of a web page (blocks).

```
1 EnrichPMWithRCC8 (wppsAPI);
2 blocks ← SelectBlocks (area, wppsAPI);
```

Definition 5.2 (Query Q_2). Given ontology $\mathcal{O} = \langle C, H_C, R_C, H_R, I, \iota, R_I, A \rangle$ as it is specified in Definition 2.1 (on page 48) and relation $\rho \in R_C$, a query $Q_2(\rho)$ returns the set B such that $\forall b_i \in I, b_j \in I[\rho(b_i, b_j) \rightarrow \langle b_i, b_j \rangle \in B]$.

In general, with the use of the exhaustive search and ignoring the complexity of relevant reasoning mechanisms, the time complexity of the query Q_1 can be described by $O(n)$ and the time complexity of the query Q_2 by $O(n^2)$.

5.5.3 Evaluation Wrappers

Based on the queries presented and the parameters determined, we define two main scenarios for the experiment. The **first scenario** for the query Q_1 consists of extracting *blocks* (mainly *visualized elements*, *viewport blocks*, *page blocks* and a *document block*, see Section 4.4.2) of the BGM (see Section 4.4.2) from the area which is located in the center of the web page (a document block). Its width and height take 50% of the width and height of the web page respectively. The containment of a block within the specific area is defined by means of the topology of blocks such as RCC8 relations. A generic procedure is presented in Algorithm 5.4. It includes three main operations: 1) computing the required area (ComputeArea), 2) enriching the PM with basic RCC8 relations (optional procedure EnrichPMWithRCC8 which enables instantiation of the topological relationships between blocks, see Section 3.6), and 3) selecting blocks from the area acquired (SelectBlocks). The **second scenario** for the query Q_2 is related to the extraction of pairs of blocks with topological relation P between them. A generic algorithm for this scenario is presented in Algorithm 5.5. It has the same procedures as the algorithm of the first scenario except for computation of the required area.

For the purpose of the evaluation, we prepared 28 wrappers (14 wrappers for each scenario) realizing the specified algorithms. Table 5.1 presents 14 types of wrappers' configurations independent from the scenarios and 6 main dimensions of these configurations reflecting the implementation of corresponding wrappers and their WPPS parameters. Thus, dimensions 3, 5,

and 6 are realized by the wrappers, whereas dimensions 1, 2, and 4 reflect the WPPS parameters that can be specified in the configuration file (see Section 5.5.2). In particular, the **first dimension** defines a standard reasoner used (i.e., the `jena-reasoner` parameter) and realization of the ontology (i.e., the `type` parameter for the ontology) accordingly. Thus, the value “RDFS” in this dimension corresponds to `jena-reasoner=RDFS_SIMPLE` and `type=RDF`, whereas the value “OWL” refers to `jena-reasoner=OWL_DL_MEM_RULE_INF` and `type=OWL`. The **second dimension** determines the set of rules to be additionally applied (i.e., the `rules` parameter). We present four types of rules capturing different TBox statements of the UOM (see Section 4.3) such as property (relation) subsumption, symmetry and inversion which are required for some wrappers to realize relevant queries. In the evaluation, we use the forward chaining rule engine provided by Jena [232] (i.e., `ruleMode=FORWARD`). The **third dimension** defines the need to invoke enricher to instantiate basic RCC8 relations (i.e., topological relations) in the ontology (procedure `EnrichPMWithRCC8` in Algorithms 5.4 and 5.5). We use the enricher `BasicSpatialRelationsEnricherBasedOnQntInfo` integrated into WPPS and instantiating basic qualitative relations of a certain type (e.g., RCC8, O-direction relation, distance, and alignment) in the ontology. The enricher has two modes which differ in that one mode does not materialize relations between objects which are symmetric or inverse to the materialized ones. Thus, the enrichers add $n(n-1)/2$ or $n(n-1)$ assertions depending on the mode of enrichment forming the complete graph (n is the number of the blocks considered). Therefore, the enriching procedure should at least consist of the quadratic complexity, such as $O(g(n^2))$. The **fourth dimension** refers to the property (relation) configuration (see Section 5.3.2). If RCC8 relations should be computed “on-the-fly” by the WPPS framework and their computation is “based on quantitative information,” we assume RCC8 to be mentioned in the `compute-by-request` parameter with attribute `basis="quantitative"` within the WPPS configuration file. If relations (i.e., basic RCC8 relations in our case) are presented in the ontology due to the enriching procedures and can be queried by WPPS without additional computations (value “select from the PM” in this dimension), the value `RCC8` should be specified in the WPPS parameter `support-in-ontology`. If relations are presented in the ontology and composite ones should be computed “on-the-fly” by WPPS based on the basic ones (value “composite rel. are based on the basic” for the dimension), the value `RCC8` should also be assigned to the parameter `composite-basic-dependence` within the WPPS configuration file. The **fifth dimension** defines topological relations which should be considered by the wrapper for querying the UOM. The **sixth dimension** defines a method used in the wrapper for implementing relevant query (i.e., Q_1 or Q_2). Thus, we distinguish three of such methods: 1) the application of the R-tree, 2) traversing all blocks from the PM (“exhaustive search”) with the use of the WPPS interface providing the abstraction from the UOM realization, and 3) directly querying the model (in this case, the user’s query is translated by WPPS into the Jena API without the use of SPARQL).

According to the configurations presented in Table 5.1, we name the corresponding wrappers as the following: $\langle config_name \rangle(\langle query\ type \rangle)$. For example, a wrapper with configuration `8DExhSearchBasicRel` implementing the second scenario is named as `8DExhSearchBasicRel(Q2)`.

The specified dimensions enable different ways of implementing the defined scenarios. For example, wrappers `1ARTree(Q1, Q2)` query WPPS for the containment relation P by means of

Table 5.1: Configurations of wrappers evaluating WPPS regardless of the implemented scenario

Wrapper	1. Standard reasoner	2. Rules	3. RCC8 relations enricher	4. Computation of the relations	5. Considered relations	6. Blocks selection
1ARTree	—	—	—	based on quantitative information	P	R-Tree
2BExhSearchQntRel	—	—	—	based on quant. inform.	P	exhaustive search
3CExhSearchBasicRel	—	—	with symmetric and inversive relations	select from the PM	NTPP, TPP, EQUAL	exhaustive search
4CExhSearchCompRel	—	—	with sym. and invers. rel.	composite rel. are based on the basic	P	exhaustive search
5CQueryBasicRel	—	—	with sym. and invers. rel.	select from the PM	NTPP, TPP, EQUAL	query the PM
6CQueryRDFSCompRel	RDFS	—	with sym. and invers. rel.	select from the PM	P	query the PM
7CQueryCompRelPRules	—	property subsumption	with sym. and invers. rel.	select from the PM	P	query the PM
8DExhSearchBasicRel	—	—	without sym. and invers. rel.	select from the PM	NTPP, TPP, EQUAL, NTPP ⁻ , TPP ⁻	exhaustive search
9DExhSearchBasicRelSIRules	—	symmetry and inversion	without sym. and invers. rel.	select from the PM	NTPP, TPP, EQUAL	exhaustive search
10DQueryBasicRel	—	—	without sym. and invers. rel.	select from the PM	NTPP, TPP, EQUAL, NTPP ⁻ , TPP ⁻	query the PM
11DQueryBasicRelSIRules	—	symmetry and inversion	without sym. and invers. rel.	select from the PM	NTPP, TPP, EQUAL	query the PM
12DQueryDLCompRel	OWL DL	—	without sym. and invers. rel.	select from the PM	P	query the PM
13DQueryRDFSCompRelIRules	RDFS	inversion	without sym. and invers. rel.	select from the PM	P	query the PM
14DQueryCompRelSIPRules	—	prop. subsump., sym., inversion	without sym. and invers. rel.	select from the PM	P	query the PM

the R-tree, which in turn uses quantitative information (coordinates of blocks) for this purpose. `2BExhSearchQntRel(Q1, Q2)` with the “exhaustive search” approach requests the WPPS framework for the composite relation P (between blocks) which the framework also computes “on-the-fly” based on quantitative information. `13DQueryRDFSCompRelRules(Q1, Q2)` utilizes RDF graph to instantiate the ontology and standard RDFS reasoner together with supplementary inference rules for inverse relations. It enriches the ontology with basic RCC8 relations without symmetric and inverse counterparts and applies direct queries regarding P relations to the ontology. These queries are feasible due to the automatic reasoning over the property subsumption by the RDFS reasoner, which derives P and P⁻ relations, and the subsequent application of inference rules for the inverse relations. These inference rules provide the possibility of obtaining P for identified P⁻ relations (see the subsumption hierarchy for RCC8 in Figure 3.3 on page 70 and Figure 4.11 on page 117).

Considering the sixth dimension in more detail, it is important to note that different configurations and implementations possess different efficiencies. For example, the R-tree ensures the time complexity of search between $O(\log_m n)$ and $O(n)$. Therefore, a blocks selection procedure (`SelectBlocks`) implementing the query Q_1 of the wrapper `1ARTree(Q1)` should have similar time complexity, whereas the query Q_2 , a `SelectBlocks` procedure of the wrapper `1ARTree(Q2)` should have the time complexity between $O(n \log_m n)$ and $O(n^2)$ since the R-tree is queried for each block. For the wrappers which do not have logical reasoning in the background, an “exhaustive search” method should correspond to the linear time complexity for the query Q_1 (e.g., wrappers such as `2BExhSearchQntRel(Q1)`, and `3CExhSearchBasicRel(Q1)`) and quadratic complexity for the query Q_2 (e.g., wrappers such as `4CExhSearchCompRel(Q2)` and `8DExhSearchBasicRel(Q2)`). The time complexity of selecting blocks with the use of reasoners (e.g., wrappers such as `9DExhSearchBasicRelSIRules(Q1, Q2)`) mainly depends on the considered properties of the UOM (i.e., `TBox`). The time complexity of direct queries to the ontology depends both on query optimization realized by Jena and applied reasoners (e.g., wrappers such as `5CQueryBasicRel(Q1, Q2)` and `13DQueryRDFSCompRelRules(Q1, Q2)`).

5.5.4 Performance Analysis

Although all the wrappers were evaluated over a relatively small subset of web pages from the WPPS-HTML-DS1 [72] dataset, it was sufficient in obtaining a conclusion regarding the time complexity of different WPPS configurations and implementations. Section 3.11 conveys general statistical characteristics of WPPS-HTML-DS1 and Section 3.11.6 in particular reports the average amount of visualized elements in web pages.

We collected data on the performance of different procedures relevant to the scenarios given (see Algorithms 5.4 and 5.5): generation of the PM, computation of the required area (the procedure `ComputeArea` of the first scenario), enriching the ontology with basic RCC8 relations (the procedure `EnrichPMWithRCC8`), and selection of the required blocks (the procedure `SelectBlocks`). Due to the fact that the wrappers of both scenarios implement procedures such as generation of the PM and enriching the PM (`EnrichPMWithRCC8`), each pair of wrappers with the same configuration (but implementing different scenarios) have the same performance characteristics of such procedures. Therefore, it is sufficient to consider only wrappers from one of the scenarios. Based on this knowledge, we mainly focus on the wrappers of the first scenario

and consider differing procedures of the wrappers of the second scenario separately.

For a detailed illustration of the efficiency of different wrappers, we generated the following graphs based on the data acquired: Figures 5.17, 5.18, and 5.19 reflect the overall time complexity of wrappers. Figures 5.20 and 5.21 profile wrappers demonstrating the average efficiency of the realized procedures. Figure 5.22 reflects the complexity of the process of instantiating the PM. Figures 5.23, 5.24, and 5.25 describe in detail the performance of computing the required area on a web page canvas. Processes of enriching the PM with basic RCC8 relations is presented in Figures 5.26 and 5.27. The final blocks selection procedure is illustrated in Figures 5.28, 5.29, 5.30, and 5.31 in detail.

It is worth mentioning that the results of the experiment can differ to some extent depending on certain wrappers when using other ontology frameworks (e.g., OWL API). This specifically refers to the application of the standard reasoner (e.g., RDFS and OWL DL) and supplementary inference rules. Due to the limitations of the Jena framework, for example, with the mutual use of a standard reasoner and custom inference rules WPPS cascades relevant reasoners building two “layers” as suggested in [232, Sec. 6.9] (i.e., a reasoner leveraging custom inference rules operates over the results of the standard reasoner). When interacting with the ontology, this approach ensures sequential application of the standard reasoner and the reasoner for custom inference rules. This fact explains the higher complexity of `13DQueryRDFSCompRelRules`, using both RDFS reasoner and logical rules for inverse relations, in contrast to `12DQueryDLCompRel` leveraging solely integrated OWL DL reasoner. Furthermore, we did not apply WPPS parameter `PROPenableTGCCaching` passed to Jena in the experiment, requiring the Jena reasoner to cache transitive closures (e.g., for the subproperty and subclass lattices). According to our additional experiments, its use may boost up the performance of the reasoner by 5–10%.

Overall Evaluation

The measurement of the time complexity of the wrappers is illustrated in Figures 5.17, 5.18, and 5.19, in which the dependency between quantity of blocks (such as *visualized elements*, *viewport blocks*, *page blocks* and a *document block*) and time required for wrappers is reflected.

Figures 5.20 and 5.21 represent program profiling for the wrappers. It is important to note that certain pairs of wrappers with the same WPPS configuration but implementing different scenarios have similar overall time complexity, for example, `5CQueryBasicRel(Q1)` and `5CQuery-BasicRel(Q2)`. This fact is highlighted in the figures.

Most of the wrappers are between quadratic and exponential complexity (see Figures 5.17 and 5.18). (In this thesis we do not conduct a formal analysis of the complexity of different wrappers.) Based on the illustrations, we can distinguish several principal groups of wrappers with similar time complexity characteristics:

- `1ARTree(Q1, Q2)` and `2BExhSearchQntRel(Q1, Q2)` are the most efficient evaluation wrappers. They do not need any logical inference and are purely based on the quantitative information, in particular, coordinates of blocks stored in the instance of the QntBGM. `2BExhSearchQntRel(Q1)` has the complexity close to the linear (see Figures 5.19a), that is mainly ensured by the PM instantiation procedure (see Figure 5.20b), and requires less than two seconds for web pages with less than 2000 visualized elements (average web

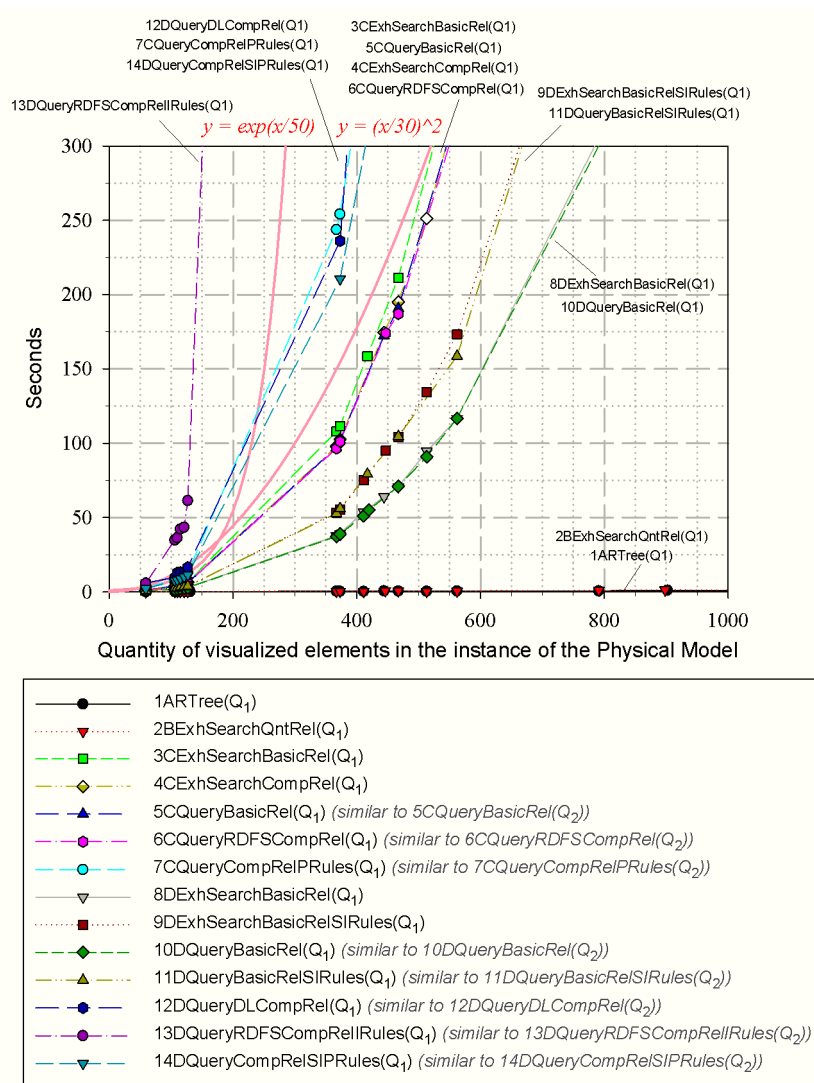


Figure 5.17: Overall evaluation of wrappers implementing the first scenario, leveraging different WPPS configurations and methods of querying the PM. Estimations of the time complexity of some wrappers of the first scenario are similar to the wrappers of the second scenario, as specified on the graph

page in the WPPS-HTML-DS1 dataset has 1105 visualized elements, see Section 3.11.6). 2BExhSearchQntRel(Q_2) in turn reflects the quadratic complexity of the query Q_2 (see Figure 5.18) that, in contrast, is mainly ensured by the blocks selection procedure (see Figure 5.21b). Wrappers 1ARTree(Q_1 , Q_2) have surprisingly good performance (see Figure 5.19) and outperforms all the others. This fact is also supported by Figures 5.20a and 5.21a, where it is evident that the blocks selection procedure is dramatically more efficient than other procedures (i.e., the PM instantiation) in the wrappers.

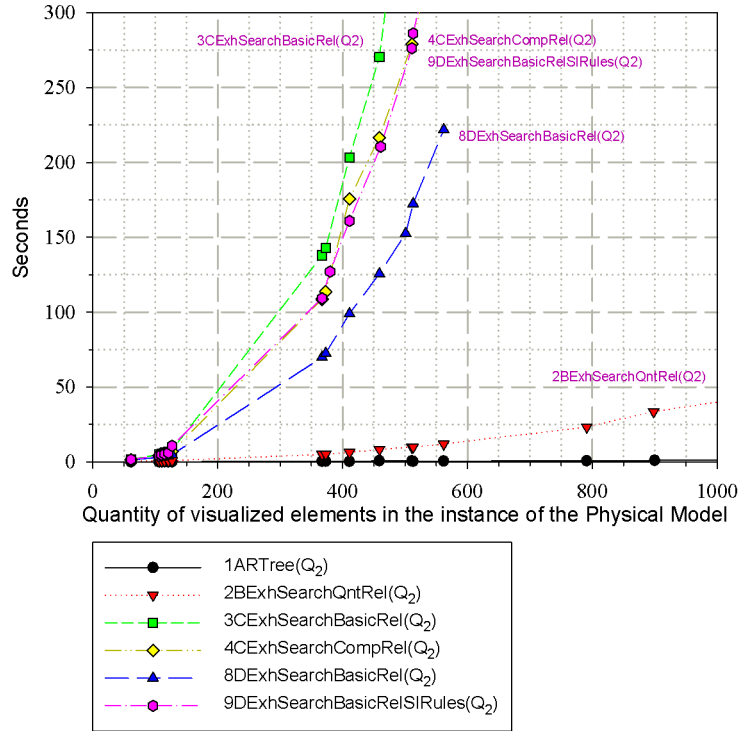
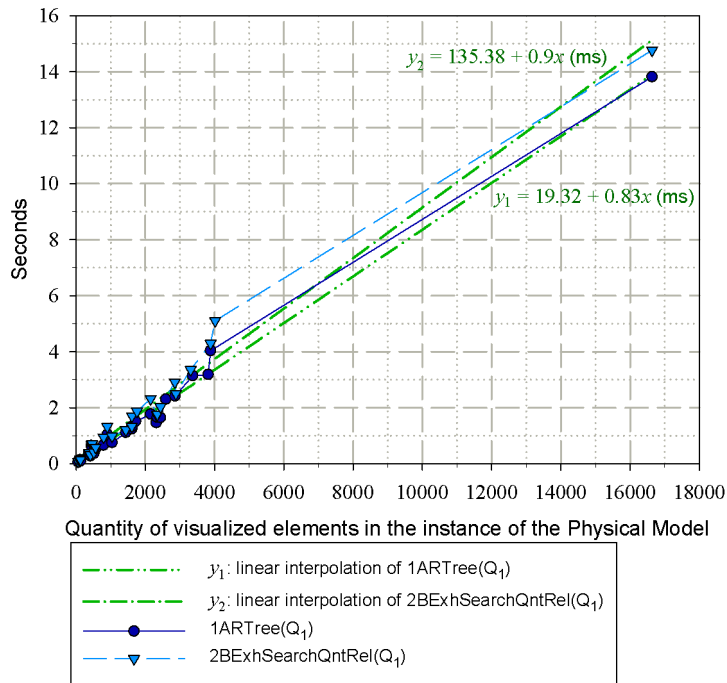
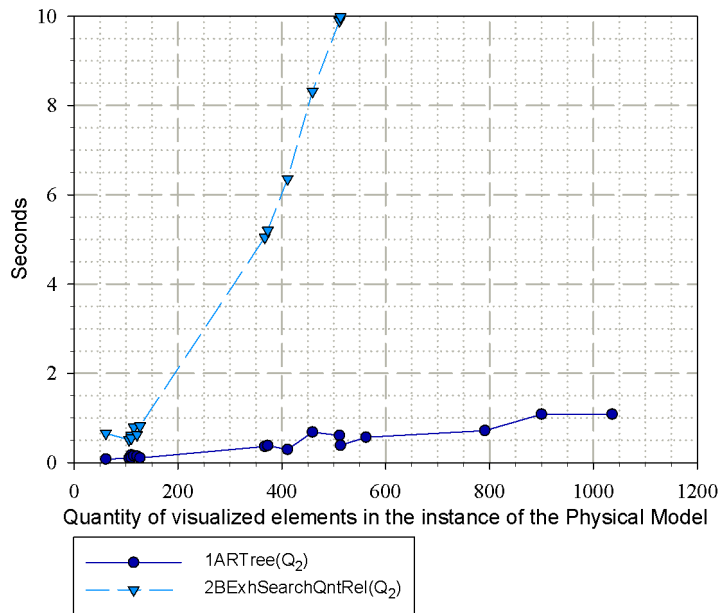


Figure 5.18: Overall evaluation of distinct wrappers implementing the second scenario, leveraging different WPPS configurations and methods of querying the PM

- $8DEXhSearchBasicRel(Q_1)$ and $10DQueryBasicRel(Q_1, Q_2)$ query basic relations in the instance of the PM which is enriched with RCC8 relations without symmetric and inverse counterparts. $8DEXhSearchBasicRel(Q_1)$ realizes queries via WPPS API providing the object-oriented abstraction over the ontology. Wrappers $10DQueryBasicRel(Q_1, Q_2)$ in turn leverage direct queries to the instance of the PM (particularly, the instance of the BGM without the abstraction mechanisms). The complexity is mainly caused by the enriching procedure (see Figure 5.20).
- In contrast, $9DEXhSearchBasicRelSIRules(Q_1)$ and $11DQueryBasicRelSIRules(Q_1, Q_2)$ apply inference rules for deriving symmetric and inverse relations for basic RCC8 relations instantiated in the ontology. It permits us to consider only three basic topological relationships (i.e., NTPP, TPP, and EQUAL) when selecting relevant blocks. The complexity is mainly ensured by the enricher and partially by the blocks selection procedure (see Figure 5.20).
- $8DEXhSearchBasicRel(Q_2)$ is less efficient than $8DEXhSearchBasicRel(Q_1)$ due to the implementation of the query Q_2 through the “exhaustive search.” This fact is supported in Figures 5.20h and 5.21e, where we can see that the `SelectBlocks` procedure takes 48% of the run time for $8DEXhSearchBasicRel(Q_2)$ and only 0.21% for $8DEXhSearchBasicRel(Q_1)$.



(a) Overall evaluation of wrappers 1ARTree(Q₁) and 2BExhSearchQntRel(Q₁) implementing the first scenario



(b) Overall evaluation of wrappers 1ARTree(Q₂) and 2BExhSearchQntRel(Q₂) implementing the second scenario

Figure 5.19: Overall evaluation of wrappers with configurations 1ARTree and 2BExhSearchQntRel

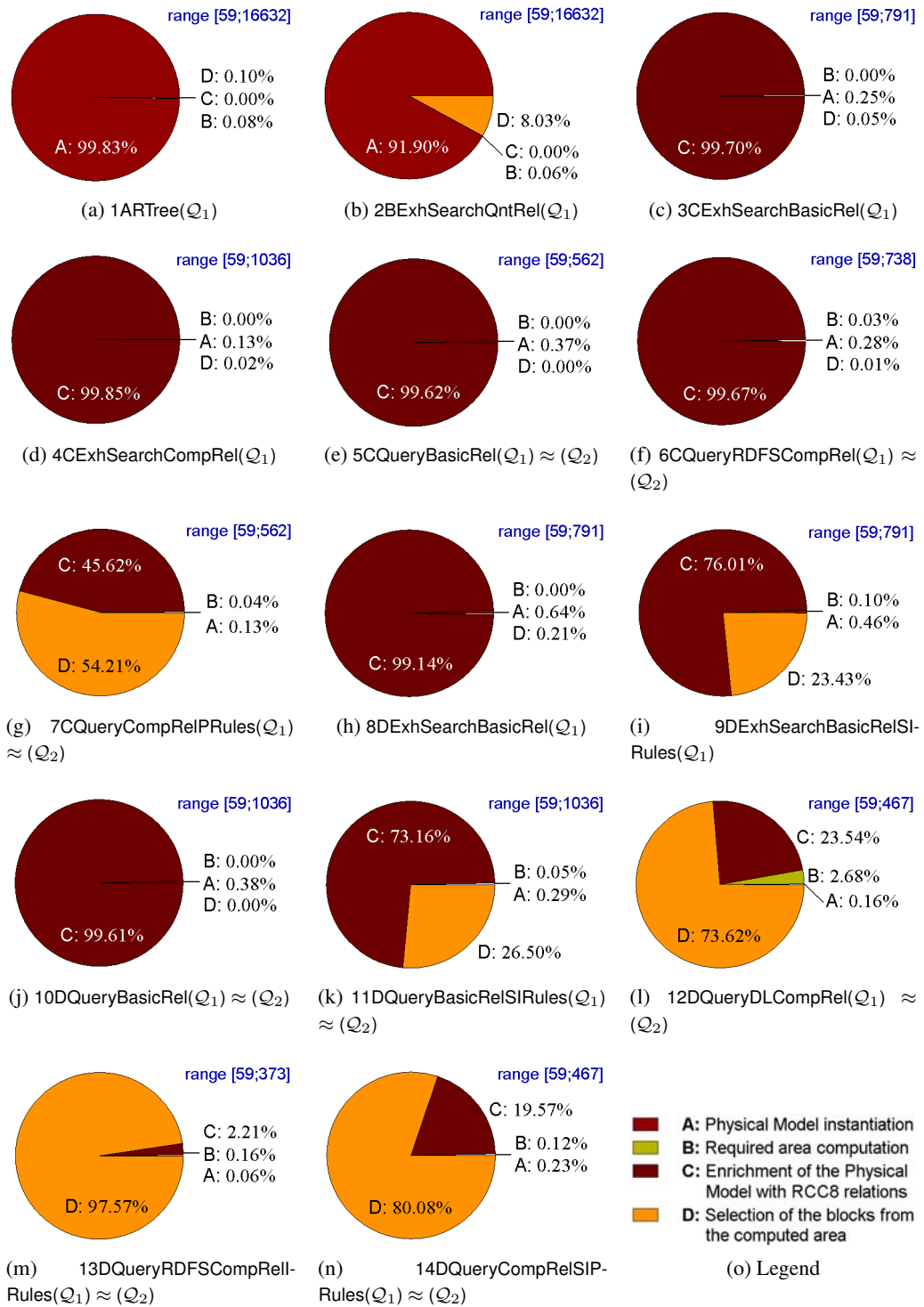


Figure 5.20: Profiling the evaluation wrappers: average values of the performance

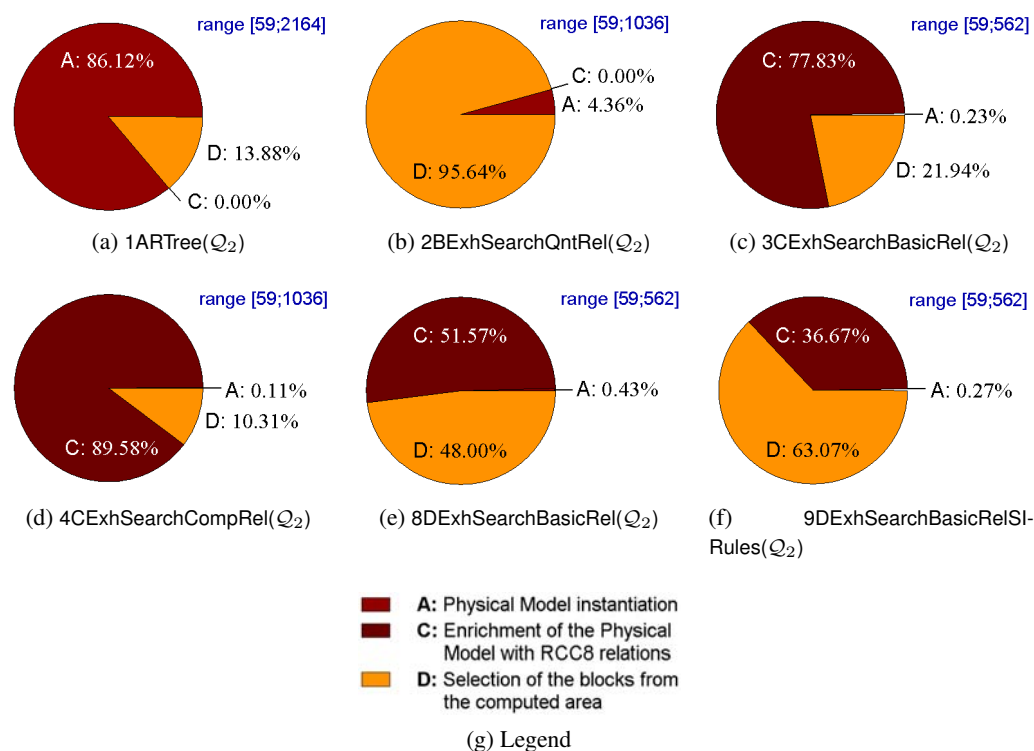


Figure 5.21: Profiling the wrappers implementing the second scenario with exhaustive search: average values of the performance

- 3CExhSearchBasicRel(Q_1), 4CExhSearchCompRel(Q_1, Q_2), 5CQueryBasicRel(Q_1, Q_2), and 6CQueryRDFSCompRel(Q_1, Q_2) leverage the enricher materializing RCC8 relations with their symmetric and inverse counterparts. This enricher is the main time-consuming procedure in these wrappers (see Figures 5.20 and 5.21). It makes them less efficient in comparison with wrappers of the groups mentioned earlier. Wrappers within this group select relevant relations from the instance of the PM. For 4CExhSearchCompRel(Q_1, Q_2), WPPS provides composite relations computing them based on the basic ones. For 6CQueryRDFSCompRel(Q_1, Q_2), WPPS utilizes the RDFS reasoner for deriving P. We also refer 9DExhSearchBasicRelSIRules(Q_2) (which uses custom inference rules to derive symmetric and inverse relations) to this group. This adds supplementary complexity to the procedure of querying the ontology and makes the wrapper less efficient than its counterpart in the first scenario.
- 3CExhSearchBasicRel(Q_2) is more time-consuming than 3CExhSearchBasicRel(Q_1) only due to the additional complexity introduced by the blocks selection procedure realizing the query Q_2 .

- $7CQueryCompRelPRules(Q_1, Q_2)$, $12DQueryDLCompRel(Q_1, Q_2)$, and $14DQueryCompRelSIPRules(Q_1, Q_2)$ apply enrichers and directly query the PM regarding the P relation. All of these wrappers leverage automatic reasoning to derive queried composite relationship P from the basic ones acquired by means of enrichers. Performance of the wrappers $12DQueryDLCompRel(Q_1, Q_2)$ and $14DQueryCompRelSIPRules(Q_1, Q_2)$ is mainly ensured by the blocks selection procedure which operates in conjunction with reasoning process performed within the WPPS framework (see Figures 5.20l and 5.20n). Efficiency of $7CQueryCompRelPRules(Q_1, Q_2)$ is defined both by the enriching ontology and blocks selection processes with automatic reasoning applied (see Figure 5.20g). The enricher in this case, in contrast to other wrappers in this group, instantiates symmetric and inverse pairs of relations. This influences the overall performance of this wrapper.
- $13DQueryRDFSCompRelRules(Q_1, Q_2)$ are the most expensive wrappers (see Figure 5.17) utilizing both the RDFS reasoner and custom inference rules for inversions (see Table 5.1 on page 170). The blocks selection procedure contributes to the complexity of wrappers (see Figure 5.20m). This example reveals the inefficiency in the joint use of the standard and custom rule-based reasoners in WPPS.

Instantiation of the Physical Model

The efficiency of the process of PM instantiation (described in Section 5.3.3) is illustrated in Figure 5.22. As we can see, the performance of the process does not depend on the configuration. This is mainly due to the fact that WPPS does not invoke a procedure of automatic reasoning during the PM instantiation and thus, the time complexity is almost the same for different wrappers. The graph reflects the linear complexity which primarily depends on the number of nodes in DOM trees of a web page.

Computation of the Required Area for the First Scenario

The efficiency of the procedure `ComputeArea` of the first scenario is illustrated in Figures 5.23, 5.24, and 5.25 according to different scales. This procedure includes querying the ontology for the document block, acquiring its coordinates, and calculating the coordinates of the required area. Thus, querying the ontology causes the underlying Jena to invoke relevant reasoners if they are present. This explains the complexity of wrappers which use reasoners.

The most time-consuming wrappers within the ranges considered are $12DQueryDLCompRel(Q_1)$ with OWL DL reasoner and $13DQueryRDFSCompRelRules(Q_1)$ which mutually leverages RDFS and custom rule reasoners (see Figure 5.23). $7CQueryCompRelPRules(Q_1)$ and $14DQueryCompRelSIPRules(Q_1)$ differ due to their use of custom inference rules for reasoning over (transitive) property (relation) subsumption. Interestingly, the $6CQueryRDFSCompRel(Q_1)$ wrapper with RDFS reasoner used for computing property subsumption is more efficient than $9DExhSearchBasicRelSIRules(Q_1)$ and $11DQueryBasicRelSIRules(Q_1)$ taking into account symmetry and inversions by means of corresponding custom rules (see Figure 5.24). In contrast, the absence of reasoning ensures relatively efficient performance (see Figure 5.25).

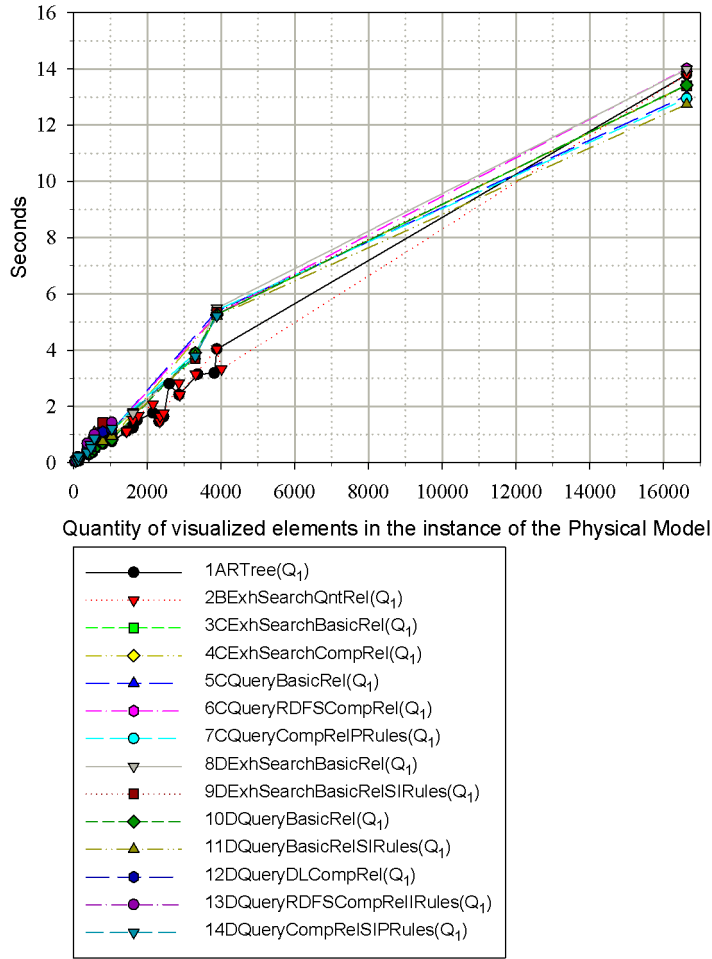


Figure 5.22: Instantiation of the PM in different evaluation wrappers implementing the first scenario (performance characteristics are the same for wrappers of the second scenario)

Enrichment of the Physical Model with RCC8 Relations

The process of enriching the ontology (a procedure `EnrichPMWithRCC8`) includes instantiation of the area required for the first scenario and application of the enricher `BasicSpatial-RelationsEnricherBasedOnQntInfo`. Figures 5.26 and 5.27 show the performance of different wrappers. As was expected, the enriching process in its practical application has a quadratic complexity and consideration of inverse and symmetric relations makes the method more time-consuming.

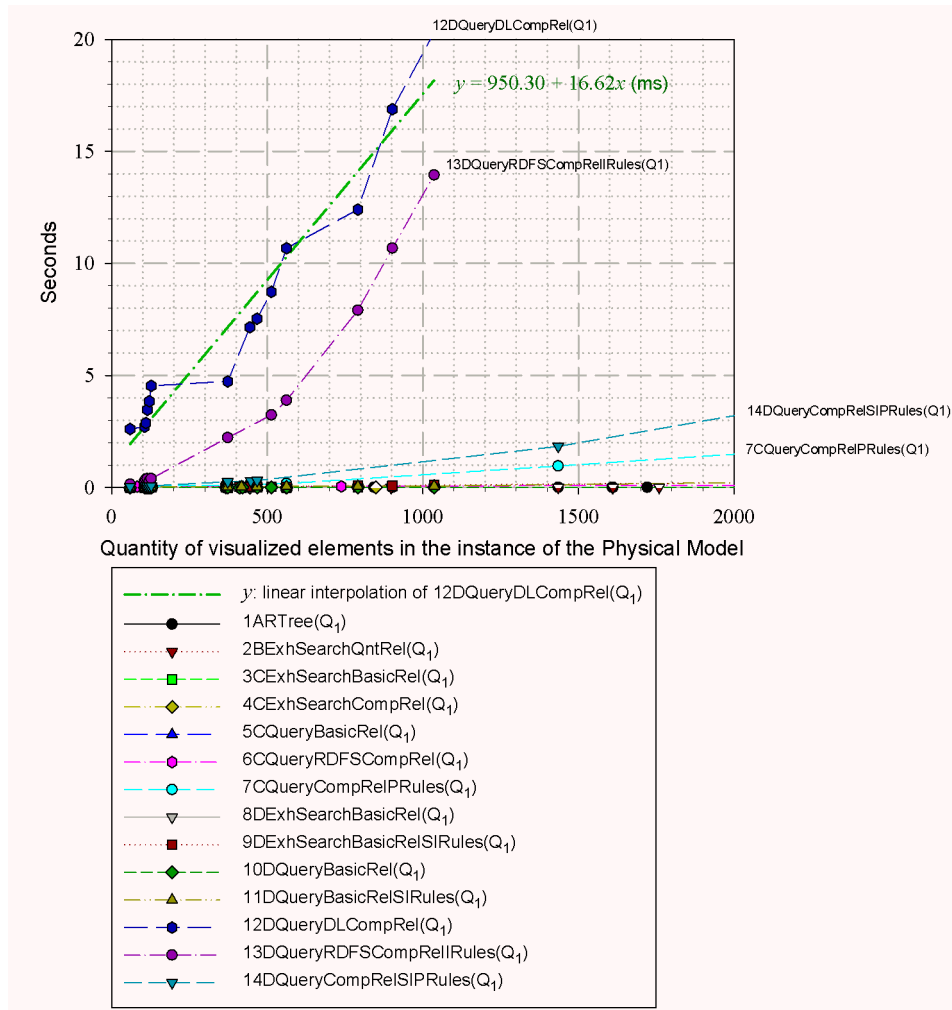


Figure 5.23: Computation of the required area by the evaluation wrappers implementing the first scenario

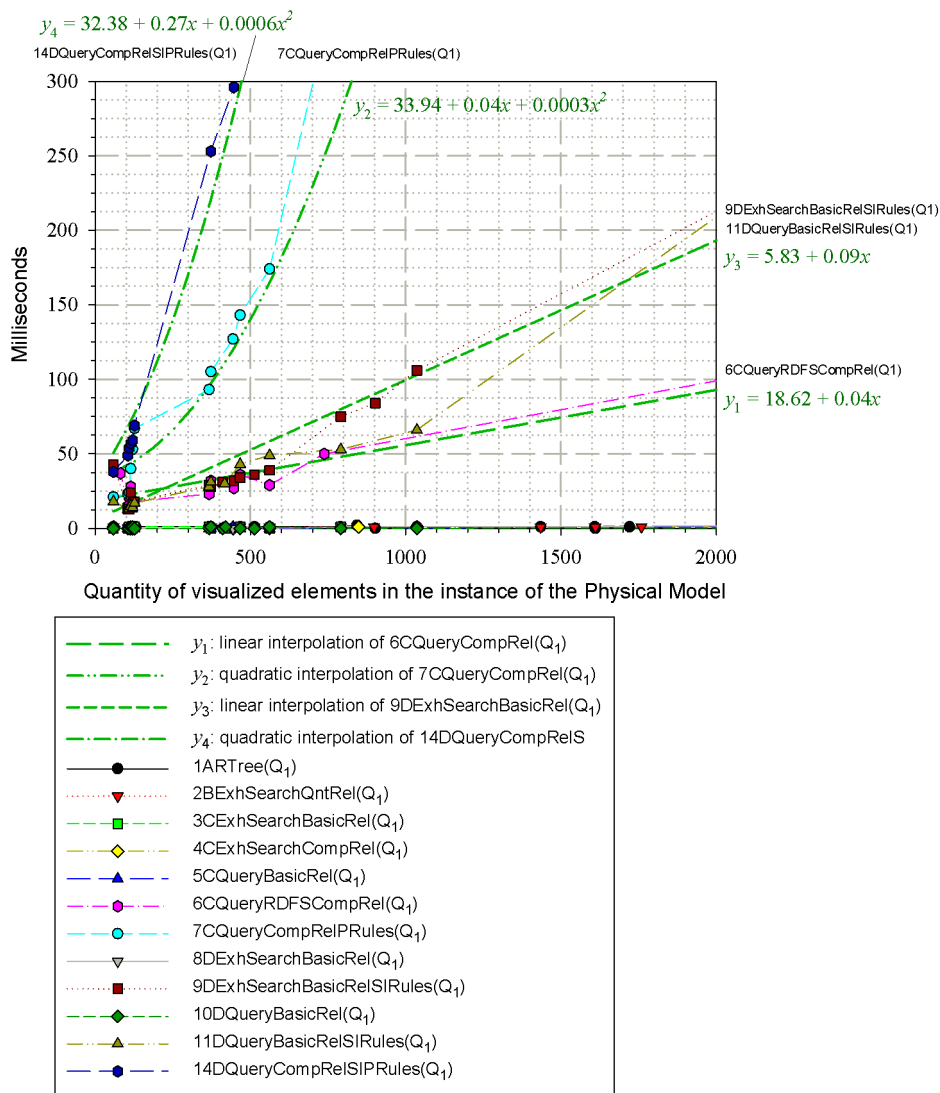


Figure 5.24: Computation of the required area by the evaluation wrappers implementing the first scenario (a 100 times larger scale on ordinate)

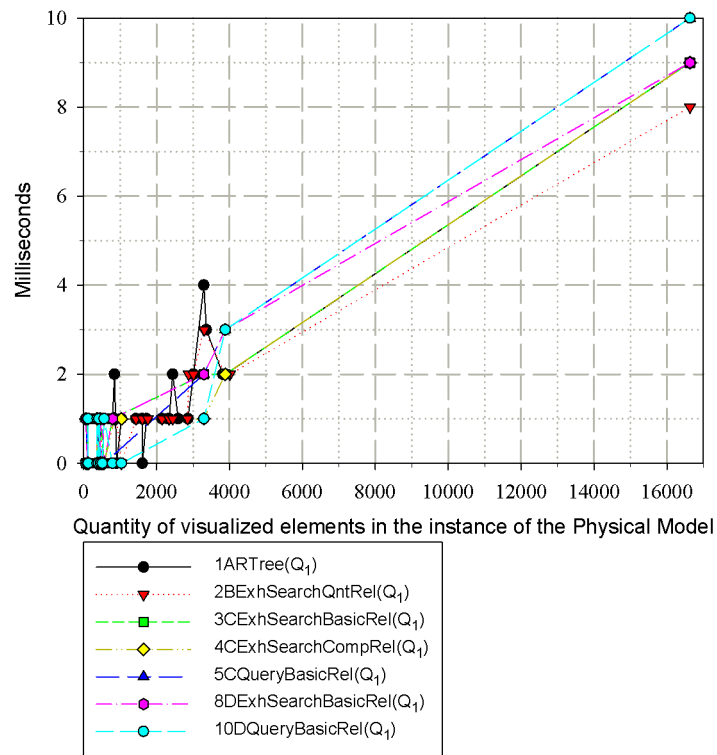


Figure 5.25: Computation of the required area by the evaluation wrappers implementing the first scenario (a 2500 times larger scale on ordinate)

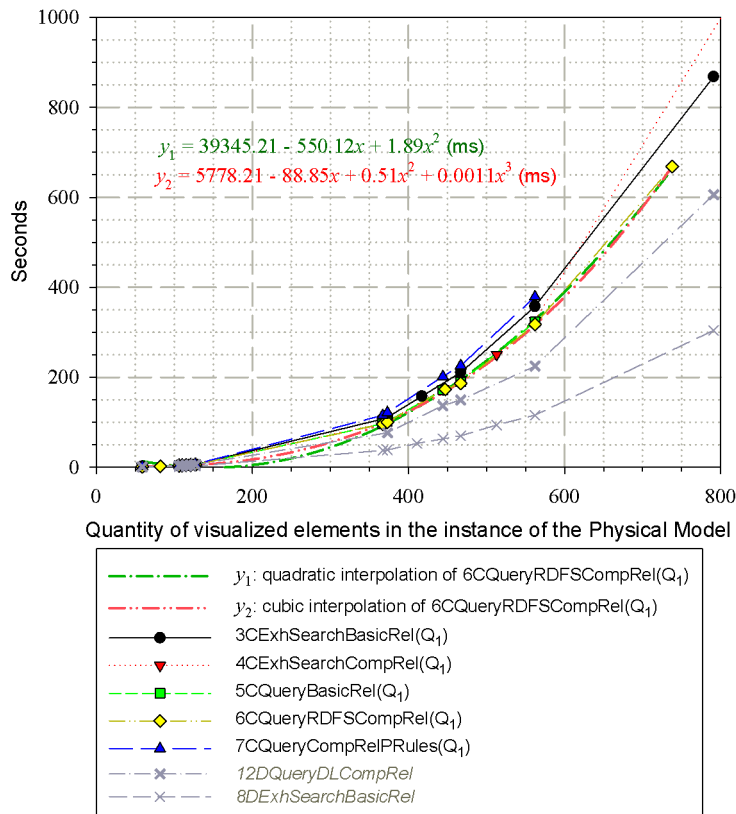


Figure 5.26: Enriching the PM with basic RCC8 relations between blocks by the wrappers implementing the first scenario (performance characteristics are the same for wrappers of the second scenario)

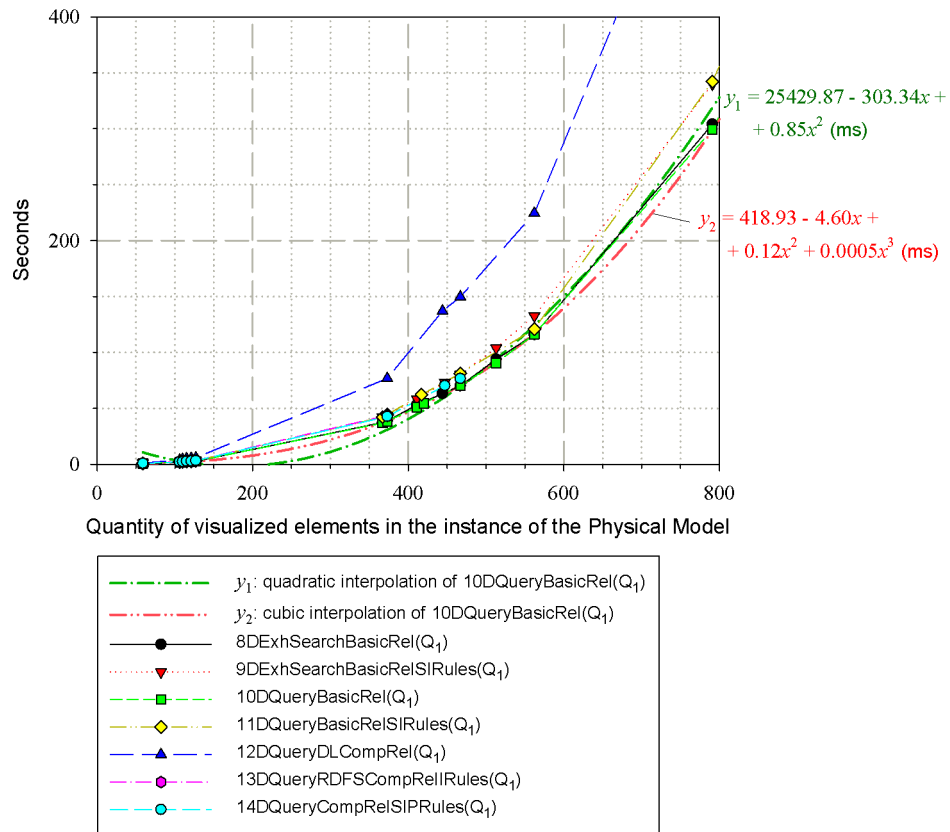


Figure 5.27: Enriching the PM with basic RCC8 relations between blocks without symmetric and inverse counterparts by the wrappers implementing the first scenario (performance characteristics are the same for wrappers of the second scenario)

Selection of Blocks

Among other wrappers realizing the “exhaustive search” for selecting the required blocks, 9DExhSearchBasicRelSIRules(Q_1, Q_2) are the only wrappers which additionally utilize a reasoner (see Table 5.1 on page 170). Therefore, their implementation of the blocks selection procedure (`SelectBlocks`) is the most time-consuming, as we can see in Figures 5.28 and 5.30. Although 3CEXhSearchBasicRel(Q_1, Q_2) and 8DEXhSearchBasicRel(Q_1, Q_2) are more efficient wrappers, they are inferior to wrappers 4CEXhSearchCompRel(Q_1, Q_2) respectively. This is due to the fact that the former conducts more interactions with the UOM (by the use of the method `hasRelation` of the interface `IQltBlock`) querying for basic RCC8 relations and makes more type casts in contrast to the latter. As was expected, 2BEXhSearchQntRel(Q_1, Q_2) are the most efficient among those leveraging the “exhaustive search” approach and have the time complexity close to the linear and quadratic respectively.

It is worth mentioning that the `SelectBlocks` procedure of the wrappers with direct queries to the instance of the PM such as 5CQueryBasicRel(Q_1, Q_2) and 10DQueryBasicRel(Q_1, Q_2) (which do not use additional reasoning) as well as 6CQueryRDFSCompRel(Q_1, Q_2) (which utilizes the RDFS reasoner) is more efficient than in 2BEXhSearchQntRel(Q_1, Q_2), which is in compliance with the scenario (see Figures 5.28 and 5.29 demonstrating the query Q_1 of the first scenario, 5.30 and 5.31 for Q_2 of the second scenario). In practice, the efficiency of a `SelectBlocks` procedure leveraging the R-tree is comparable with corresponding procedures which leverage direct queries to the ontology without the application of reasoners.

Predictably, wrappers, leveraging direct queries to the instance of the PM are significantly more efficient than those which have similar configurations and use “exhaustive search.” For example, 5CQueryBasicRel(Q_1, Q_2) and 3CEXhSearchBasicRel(Q_1, Q_2), 10DQueryBasicRel(Q_1, Q_2) and 8DEXhSearchBasicRel(Q_1, Q_2), 11DQueryBasicRelSIRules(Q_2) and 9DEXhSearchBasicRelSIRules(Q_2) in accordance with their respective scenarios (see Table 5.1 on page 170 presenting wrappers’ configurations). However, this is not the case for 11DQueryBasicRelSIRules(Q_1) and 9DEXhSearchBasicRelSIRules(Q_1), which have almost the same performance results due to the application of automatic reasoning (for symmetric and inverse relations) which takes the majority of the run time.

It is important to note that the time necessary for direct queries to the instance of the PM does not differ as much in cases of the “exhaustive search” for pairs of wrappers with the same parameters and within different scenarios. For example, for 1036 *visualized elements* and for the configuration 10DQueryBasicRel, the query Q_2 takes 35ms (see Figure 5.31) which is 17.5 times longer than the time required for the query Q_1 (2ms, see Figure 5.29). Meanwhile, for the configuration 2BEXhSearchQntRel, this ratio is 421.22 (with an accuracy of two decimal places) and the time necessary for the query Q_2 amounts to 41280ms (see Figure 5.30) in contrast to 98ms for the query Q_1 (see Figure 5.28). Furthermore, pairs of wrappers with the same configuration while performing direct queries to the PM and leveraging supplementary reasoning procedures, such as 7CQueryCompRelPRules(Q_1, Q_2), 11DQueryBasicRelSIRules(Q_1, Q_2), 12DQueryDLCompRel(Q_1, Q_2), 13DQueryRDFSCompRelIRules(Q_1, Q_2), and 14DQueryCompRelSIPRules(Q_1, Q_2), do not differ much in their efficiency. This is due to the fact that, in regards to the `SelectBlocks` procedure, the reasoner occupies the major part of the run time and operates over the same subset of assertions in ABox considering the same subset in TBox for both scenarios. However, the

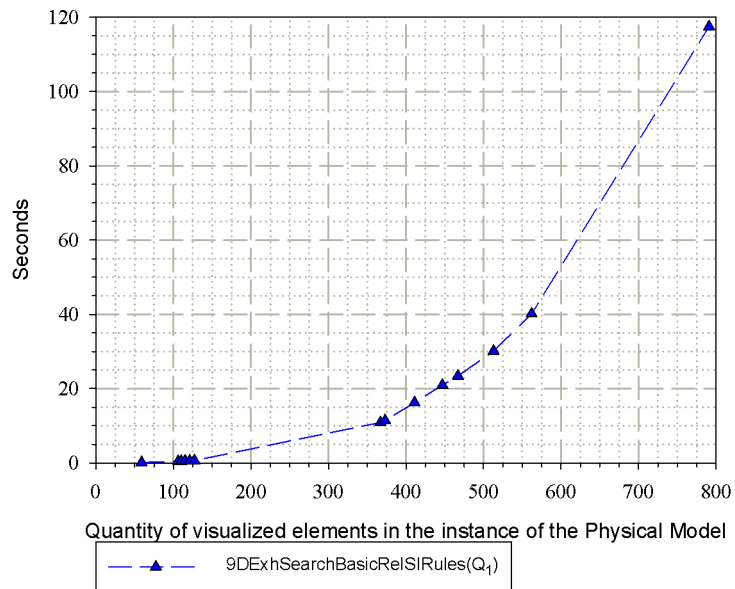
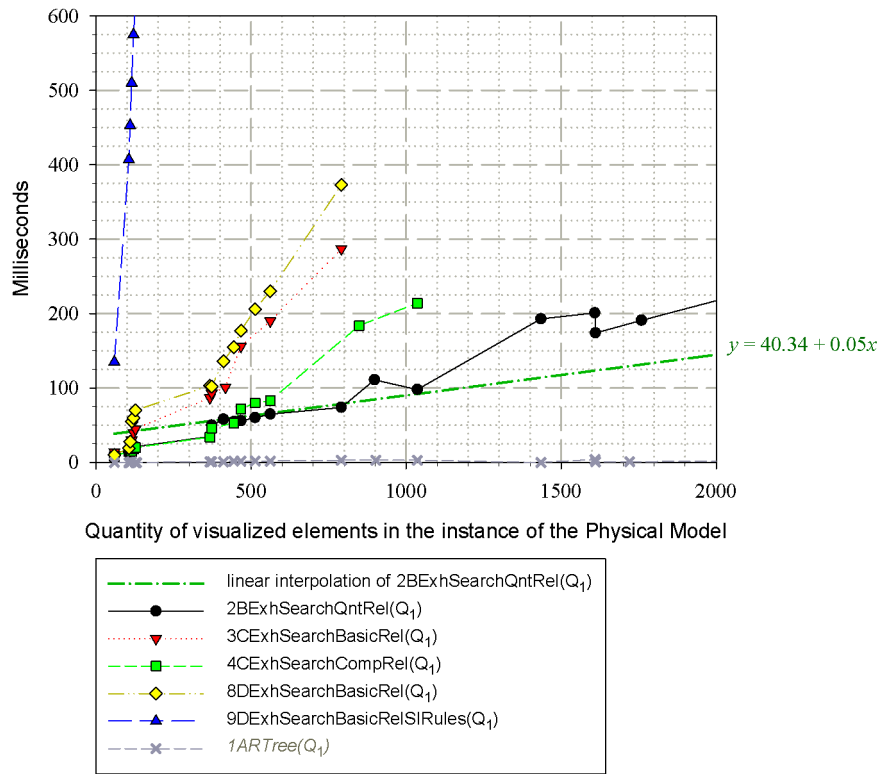


Figure 5.28: Application of the “exhaustive search” in the evaluation wrappers implementing the first scenario

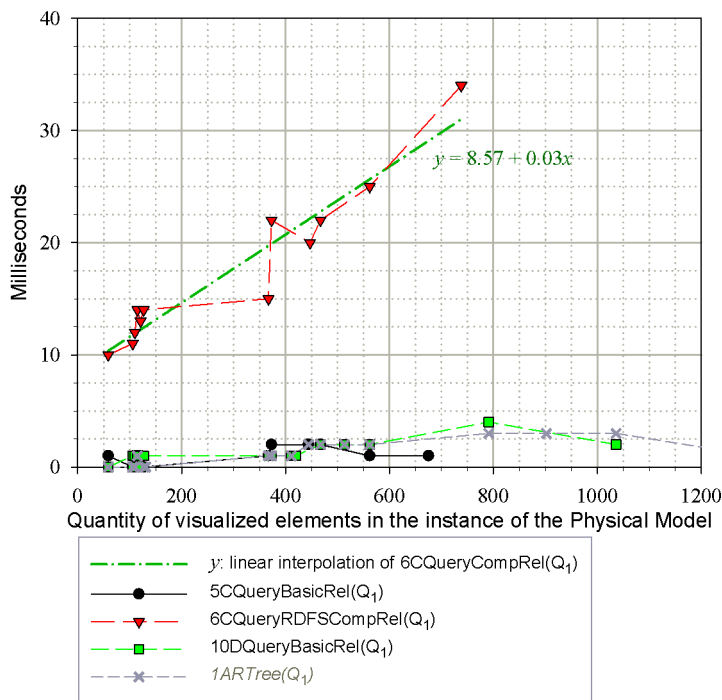
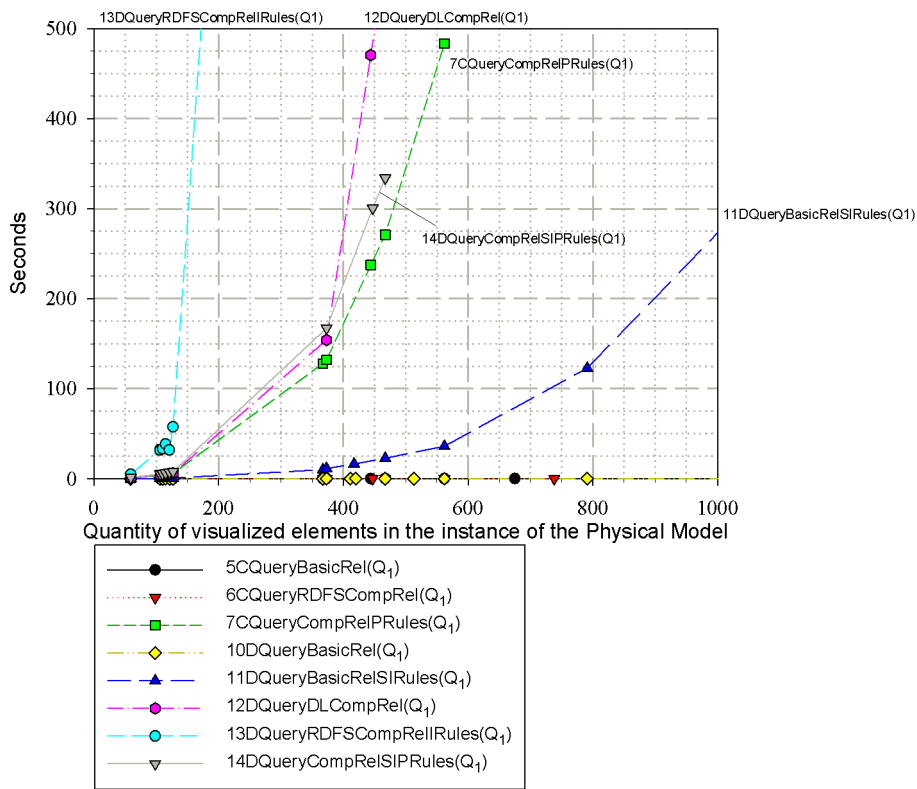


Figure 5.29: Direct querying the PM in the evaluation wrappers implementing the first scenario

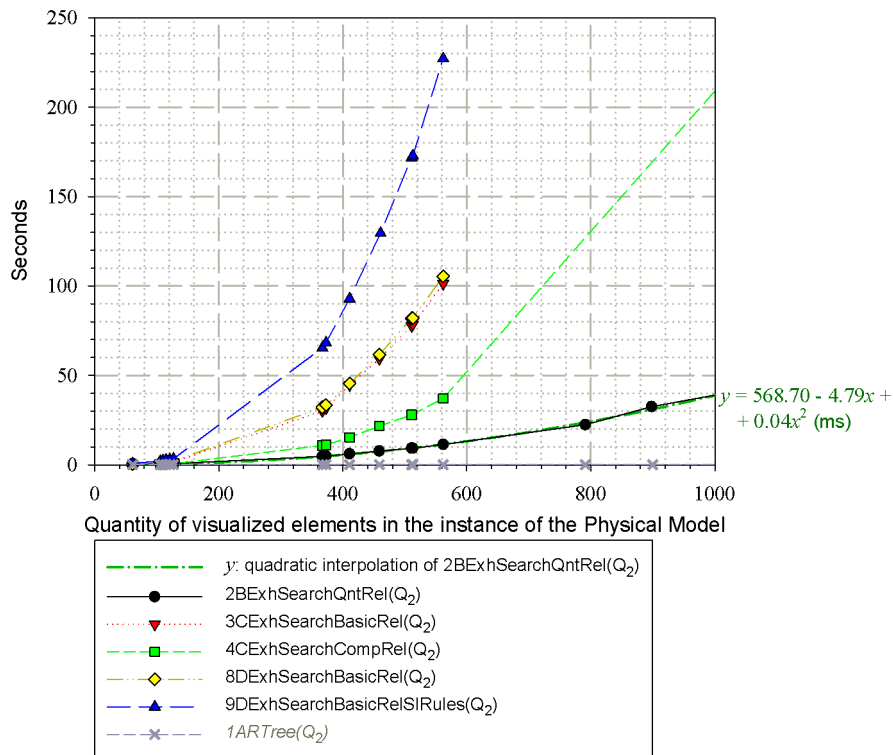


Figure 5.30: Application of the “exhaustive search” in the evaluation wrappers implementing the second scenario

situation is different for the couple $6CQueryRDFSCompRel(Q_1, Q_2)$, where the optimized RDFS reasoner is used ensuring good performance characteristics.

Another important fact is that instantiation of relevant statements in ABox is performed once by means of Jena. Thus, in cases where all the statements relevant to the query are materialized, it performs with the same efficiency as a query applied to the ontology without the use of reasoning. As such, in regards to the second and successive application of the queries of wrappers $6CQueryRDFSCompRel$, $7CQueryCompRelIPRules(Q_1, Q_2)$, $11DQueryBasicRelSIRules(Q_1, Q_2)$, $12DQueryDLCompRel(Q_1, Q_2)$, $13DQueryRDFSCompRelIRules(Q_1, Q_2)$, and $14DQueryCompRelSIPRules(Q_1, Q_2)$, their efficiency becomes comparable with the queries of wrappers $5CQueryBasicRel(Q_1, Q_2)$ and $10DQueryBasicRel(Q_1, Q_2)$ according to their respective scenario (i.e., type of query). Moreover, the second application of the query in wrapper $11DQueryBasicRelSIRules(Q_1)$ will be more efficient than in $9DExhSearchBasicRelSIRules(Q_1)$.

5.5.5 Conclusions on the WPPS Evaluation

It is important to note that WPPS is a prototype implementing the proposed object-oriented abstraction for the UOM and therefore, the framework can be optimized in the sequel according to the results of this experiment.

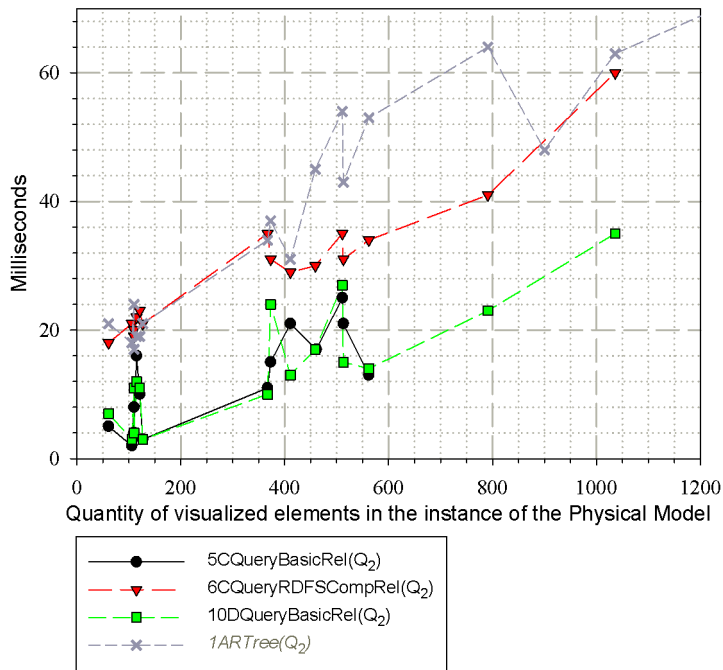
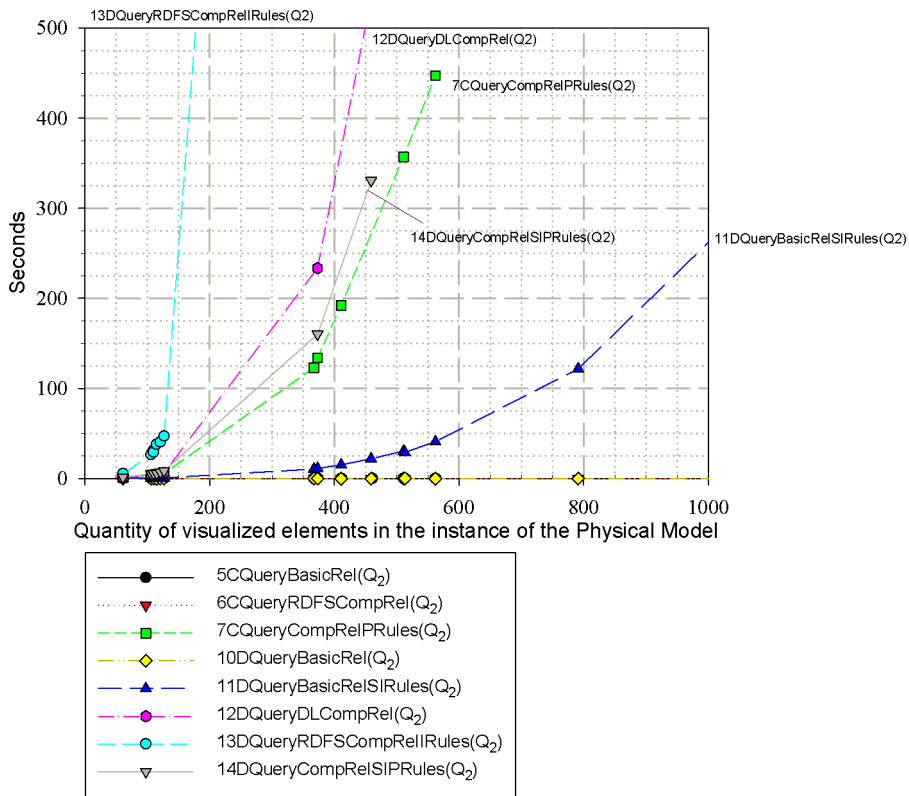


Figure 5.31: Direct querying the PM in the evaluation wrappers implementing the second scenario

The experiment confirms the feasibility of the mutual application of the declarative and object-oriented approaches with the use of the ontological model (see Section 5.2). Furthermore, it is evident procedures enriching the ontology with additional information as well as processes of automatic reasoning are quite time-consuming in general (see Figures 5.20 and 5.21). However, the application of direct queries to the ontology without the use of reasoners as well as the subsequent (not the first) application of direct queries over the materialized assertions with the use of reasoners is distinguished by the significant efficiency in contrast to the computation of the required information “on-the-fly” based on quantitative information or querying the model via the abstraction mechanisms using the “exhaustive search” approach. As was noticed in Section 5.5.4, the automatic reasoning is mainly applied to instantiate relevant ontology statements. Therefore, further queries that do not require consideration of additional assertions (i.e., when all relevant statements were instantiated) will perform as fast as queries which do not use any reasoner (logical inference). Furthermore, the presence of materialized statements (acquired by the use of enrichers and reasoners) presents great potential for the efficient application of different direct queries to the ontology by means of Jena API and SPARQL. Thus, in spite of the fact that wrappers leveraging the computation of qualitative information “on-the-fly” based on quantitative information revealed significant efficiency in single queries, their use tends to be unfavorable for complex and versatile analysis.

For example, we considered application of the query Q_2 for wrappers $2BExhSearchQntRel(Q_2)$ (which does not require application of the enricher), $6CQueryRDFSCompRel(Q_2)$ (which enriches the ontology with basic RCC8 relations), and $10DQueryBasicRel(Q_2)$ (which enriches the ontology without symmetric and inverse counterparts, see Table 5.1 on page 170). Taking into account the time required for the enriching procedure and the run time of the `SelectBlocks` procedure of $2BExhSearchQntRel(Q_2)$, we can calculate that for 511 blocks, for example, application of the configuration $2BExhSearchQntRel$ is profitable relative to $6CQueryRDFSCompRel$ if there are at most 10 queries of type Q_2 . In addition, it is profitable if there are at most 26 queries for the configuration $10DQueryBasicRel$. The values are 15 and 45 respectively in regards to 1036 visualized blocks. Thus, we recommend the developer to estimate the efficiency of the potential configuration taking into account graphs presented in this evaluation as well as illustrated estimating interpolations.

It is worth mentioning that the efficiency of the R-tree is practically similar to direct queries to the ontology without automatic reasoning. Therefore, it is recommended to use the integrated R-tree index for spatial queries related to the containment and intersection relationships.

5.6 WPPS in the Problem of Basic Web Object Identification

The problem of identifying basic web objects was posed and defined by the author of this thesis during his work in the TAMCROW project [237]. This challenge is related to the web automation (one of the main focuses of the project) intended for automating an agent’s interaction with web pages, where the agent is represented by the sighted user, blind user, mobile (phone) user, and web spider. Web automation as a technology for ameliorating accessibility of web pages is considered in Section 2.1.3 (page 17).

It is well known that the majority of existing solutions related to the web object identification problem operates on *technical layers* (see Figure 4.3 on page 106) which are prone to frequent changes and do not reflect semantics (e.g., the logical structure) formed by the visual representation [121, 145]. Consideration of a web page's visual representation enables the development of significantly more robust and effective approaches which can be applied on a wider range of web pages. (This aspect is discussed in Section 2.4.8 in detail.) Thus, the relevant goal was specified as following: Develop a generic approach for identifying basic web objects, such as departure input fields, forum threads, and menu items leveraging those features which are visually perceivable by the sighted user. The following objectives were specified to achieve this goal: **1.** Specification of an object's basic features of the *visual layers* which can be used to determine similarity between objects of the same category or genre. **2.** Development of metrics to estimate the similarity of objects. **3.** Development of an approach for identifying a basic web object of a certain category or genre on different web pages with different visual characteristics (layout, color, font, etc.). Realization of the objectives posed was conducted in collaboration with the TAMCROW team [237].

1. In the collaborative work, we identified 49 different features potentially important in the recognition of basic web objects of a certain genre [84] (see Table 5.2). These features refer to different aspects of web page representation (see Sections 2.4 and 4.2), in particular, the interface, spatial configurations, visual characteristics and text. We also distinguish between *inherent* and *relative features*. The former describes the characteristics of the web page elements themselves (e.g., of the selected object, context, or page) and are computed independently from other structural elements. Examples include font color, tag name, height, or font size. Relative features in turn reflect mutual characteristics of several structural elements, such as the number of web objects in the context or the average color distance between the selected object and all other objects within the context. Thus, relative features are considered for a pair of structural elements, such as a selected object and its context or a selected object and the document (where a context is a surrounding area of a selected object as it is illustrated in Figure 5.32). To compute these features and generate a *feature matrix* for basic web objects, the author developed an interactive annotation tool ObjIdent which is based on the WPPS framework and extensively uses the WPPS API (see Section 5.3.4). A screenshot of ObjIdent is illustrated in Figure 5.32. By means of WPPS API, ObjIdent query instances of the BGM and Interface Model providing all necessary information for computing the required features.

2. To estimate the similarity between objects, we introduced metrics for estimating the similarity between corresponding features (from the feature matrices) of relevant objects. This is based on the notion *feature distance* quantitatively expressing the similarity between corresponding features of objects. We distinguish seven types of distances dependent on the type of compared features: *absolute* and *relative distances* for numeric values, *boolean distance*, *equality distance* for nominal features, *string edit distance*, *color distance*, and *grid overlap distance* [84, 145]. The computed distances are represented in the so-called *distance matrix*.

3. Web object identification approach was realized by means of machine learning, where *distance matrices* were used as an input. Thus, trained classifiers were able to determine similarity or dissimilarity between objects based on feature distances provided. Different classification techniques were evaluated for building a classifier with the highest accuracy. These include

Table 5.2: Object features and distances in the problem of basic web object identification [84, 145]

	Feature (Feature distances)	T	Description	Dist.	
Selected object	Object type	E	Type of an object, e.g., button, input field.	Equ.	IF
	Editable	B	Whether the object is editable.	Bool.	
	Selection	B	Whether a checkbox or radio button were selected	Bool.	SF
	Area	R	The area of the bounding box in pixels.	Rel.	
	Aspect ratio	R	The aspect ratio between width and height.	Rel.	VPF
	Foreground color	C	Foreground color of the object in HSV color space.	Color	
	Background color	C	Background color of the object in HSV color space.	Color	TF
	Emphasis	R	Value representing the level of emphasis (font weight, style, etc.).	Rel.	
	Font size	R	Font size of the text.	Rel.	TF
	Text	S	Text of the selected object.	Edit	
	Number of lines	I	Number of rows (CSS client rectangles) the object contains.	Rel.	TF
	Number of tokens	I	Number of tokens (primarily words) the text contains.	Rel.	
Selected object – Context	Type of the dominant object	E	The most common type among the orthogonally visible objects.	Equ.	IF
	Objects of the same type	I	Number of objects in the context that have the same type as a selected one.	Rel.	
	Aligned objects (context)	I	Number of objects horizontally or vertically aligned with the selected object.	Rel.	SF
	Horizontally aligned objects	I	Number of objects horizontally aligned with the selected object.	Rel.	
	Vertically aligned objects	I	Number of objects vertically aligned with the selected object.	Rel.	VPF
	Horizontal index (context)	I	Index of the selected object in the sequence of horizontally aligned objects.	Abs.	
	Vertical index (context)	I	Index of the selected object in the sequence of vertically aligned objects.	Abs.	TF
	Alignment factor	R	Ratio of number of objects aligned with the selected object to unaligned ones.	Rel.	
	V/H alignment ratio (context)	R	Ratio of number of vertically aligned objects to number of horizontally aligned ones.	Rel.	VPF
	Orthogonally visible obj.	I	Number of orthogonally visible objects.	Rel.	
	Aligned orth. visible obj.	I	Number of orthogonally visible objects aligned with the selected object.	Rel.	TF
	Fully aligned orth. visible obj.	I	Number of orthogonally visible objects fully aligned with the target object.	Rel.	
	Sq. pixels to character ratio	R	Average area (in px ²) in the context that is occupied by the single character.	Rel.	VPF
	Average foreground color distance	R	Average HSV foreground color distance between the selected object and all other objects in the context.	Abs.	
	Average background color distance	R	Average HSV background color distance between the target object and all other objects in the context.	Abs.	TF
	Upper text	S	Merged text of the upper orthogonally visible objects.	Edit	
	Right text	S	Merged text of the orthogonally visible objects on the right.	Edit	TF
	Lower text	S	Merged text of the lower orthogonally visible objects.	Edit	
	Left text	S	Merged text of the orthogonally visible objects on the left.	Edit	TF
	Most similar text distance	-	Distance comparing the upper, lower, left and right text, taking the most similar ones.	-	
	Orthogonally nearest text	S	Text of the nearest orthogonally visible object in the context.	Edit	TF
Nearest text	S	Text of the nearest object in the context.	Edit		
Selected object – Page	Relative Width	R	Width of the selected object in relation to the page.	Abs.	SF
	Relative Height	R	Height of the selected object in relation to the page.	Abs.	
	Relative X	R	X-position of the the selected object in relation to the page.	Abs.	SF
	Relative Y	R	Y-position of the selected object in relation to the page.	Abs.	
	Selected object – Top page	Link Type	E	Specifies whether the target of a link is within the same page, same domain or outside.	Equ.
3x3 Grid Location		M	Dividing the web page in a 3x3 grid, this feature specifies which segments are overlapped.	Grid	SF
Selected object – Document	Alignments (document)	I	Number of objects in the document which are in any alignment with the selected object.	Rel.	SF
	Horizontal Alignments (document)	I	Number of objects in the document which are horizontally aligned with the selected object.	Rel.	
	Vertical Alignments (document)	I	Number of objects in the document which are vertically aligned with the selected object.	Rel.	TF
	Horizontal Index (document)	I	Index of the selected object in the sequence of horizontally aligned objects.	Abs.	
	Vertical Index (document)	I	Index of the selected object in the sequence of vertically aligned objects.	Abs.	TF
	V/H Alignment Ratio (document)	R	Ratio of objects vertically aligned to objects horizontally aligned regarding the selected one.	Rel.	
Context	Objects	R	Total number of objects contained in the context.	Rel.	IF
	Text density	R	Area within the context occupied by the text divided by the context's area.	Abs.	SF
	Link density	R	Area within the context occupied by links divided by the context's area.	Abs.	
	Link character density	R	Ratio of number of characters in links to all characters in the context.	Abs.	TF

Data Type (T): real (R), integer (I), enumeration (E), boolean (B), RGBA color (C), bitmap (M), string (S). **Feature groups:** interface feature (IF), spatial feature (SF), visual perception feature (VPF), textual feature (TF). **Distance type (Dist.):** absolute distance (Abs.), relative distance (Rel.), boolean distance (Bool.), equality distance (Equ.), string edit distance (Edit), color distance (Color), grid overlap distance (Grid).

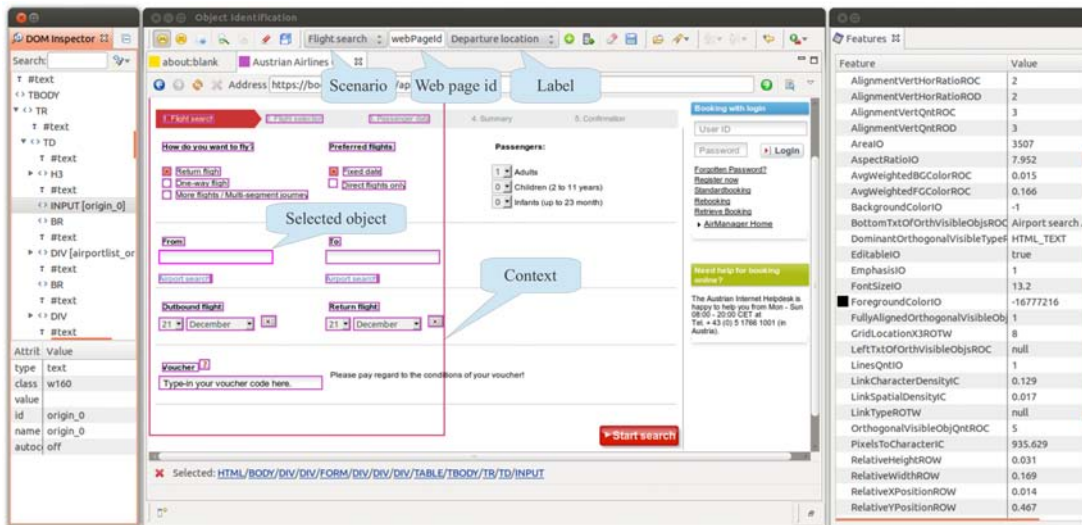


Figure 5.32: Screenshot of ObjIdent, the feature extraction tool

logistic regression, C4.5 for the decision tree, RPart for building regression trees, M5P, and SVM with linear, polynomial, radial, and sigmoid kernels. In [121], we applied some of these techniques taking into account certain features from Table 5.2. The approach was evaluated in different scenarios such as identification of forum threads, items of the navigation menu, and elements of the login web form. The classification rate was achieved up to 90% with the use of regression tree. In [145], we present the problem of identifying the most similar object to the required one. This approach also includes a post-processing of the classifiers' output. In this procedure, we count the number of times an object was affiliated with the class "similar" by the classifier and compare it with original examples. Thus, only one object with the highest score is selected. Different machine learning techniques with 50 different feature distances (see Table 5.2 [84]) in their input were applied for three scenarios such as bus search, flight search, and train search. Each scenario includes the identification of main controls such as departure location, arrival location, dates, and submit button. The classification rates achieved by combining the logistic regression with the sigmoid kernel of the SVM resulted in the highest scores which were more than 90% for each scenario. For further details, we refer interested readers to [84, 121, 145].

It is possible to find the application of this approach in different areas, be it web page processing, web automation, or annotation-based web transcoding.

In turn, the WPPS framework demonstrated its effectiveness and efficiency in the rapid development of applications for the web page analysis. ObjIdent was developed as Eclipse RCP based application. It integrates plug-ins provided by WPPS, which enable utilizing the integrated web browser, ontology graph visualization, and widgets from the ATF project. Thus, ObjIdent was leveraged by the TAMCROW team as a reliable features provider.

5.7 Discussion

In this chapter, we presented the prototype Web Page Processing System (WPPS) which provides the developer with all the necessary means for developing effective and robust web page processing methods (see Section 5.3). Based on the Unified Ontological Model (UOM), the WPPS framework realizes a web page processing as a process of building the logical model based on the analysis conducted over a web page's physical model (see Section 5.1). This provides additional possibilities for the analysis of different aspects of web page representation, such as layout, different spatial configurations, interface with different basic visual features, and DOM trees in particular. WPPS also eliminates elements which are invisible or visually not perceptible by the user and therefore do not play any important role in the analysis of web page appearances. Thus, taking into account these elements (i.e., 15.85% which are *invisible visualized and unvisualized elements* specified in Section 3.11.6 and 26% which are removed during the simplification procedure), WPPS in general removes about 37.7% elements, which makes interaction with an instance of the UOM more efficient. Furthermore, ontological representation of information enables seamless integration with ontology-based external systems and Semantic Web technologies. In terms of efficiency, WPPS realizes a bridged adapter design pattern to form an object-oriented abstraction for the UOM (see Section 5.2.4). This abstraction provides the developer the possibility to build methods, leveraging the object-oriented paradigm which are independent from certain realization of the underlying ontologies. Therefore, with the WPPS configuration provided (see Section 5.3.2), the framework realizes the relevant strategy for providing a method developed with the required information, be it objects (individuals), attributes (datatype properties), or their relations (object properties). Utilizing WPPS configuration, it is possible to control the information handling by WPPS as well as processes which should be performed on the declarative level and those which should be performed on the object oriented level. The former includes the application of reasoners specified by the user and SPARQL queries, whereas the latter ensures the imperative approach by computing the requested information primarily "on-the-fly." Thus, WPPS provides the opportunity to benefit from both declarative and object-oriented approaches. An evaluation presented in Section 5.5 demonstrates the feasibility of the solution proposed as well as advantages and disadvantages of different configurations. It is worth noting the efficiency of leveraging the R-tree integrated into the framework for querying the containment and overlap spatial relationships. Furthermore, WPPS enables leveraging fuzziness in spatial relations. Realization of the UOM in WPPS, computations and queries as well as API (see Section 5.3.4) are compliant with definitions introduced in Chapters 3 and 4.

As was demonstrated in Section 5.4.2, the use of visual cues enables the development of more robust methods which can be applied on a significantly wider range of web pages, in contrast to approaches leveraging only the technical layers (i.e., the source code and DOM tree). This is mainly due to the fact that there are less design patterns and spatial configurations used to represent different web objects than various possible implementations (i.e., codings and tag trees) on the technical level. Furthermore, when leveraging visual features, developers primarily make use of the same lexicon and concepts which are used explicitly or by implication by sighted users familiarizing themselves with web pages. For example, distance, alignment, containment, color, type of element (image, text field, button, etc.) and layout (list, table). This can provide better

understanding about how sighted users perceive and recognize web pages. Interestingly, in the work [121, 145] mentioned in Section 5.6, we concluded that all 49 specified visual features were significant in the challenge of web object identification and obtained a surprisingly high accuracy of more than 90%. Moreover, there were no such subsets of the defined features which gave us similar scores. Within the scope of this research, the WPPS framework was used for the development of a relevant feature extraction tool confirming its effectiveness in the rapid application development.

Thus, WPPS proposed in this chapter demonstrated the feasibility of the proposed web page model (the UOM) and approach integrating declarative and object-oriented paradigms for web page processing. WPPS is an important tool for enhancing web accessibility. In particular, it can be used in annotation-based transcoding (providing the transcoding algorithm with required labels of recognized web objects) and web automation (improving the accuracy of the scripts interacting with different web page elements on behalf of the user). Within this thesis, WPPS is proposed as a web page understanding tool providing necessary information regarding the logical structure of a web page and its elements by means of the Logical Model for instantiating the Multi-Axial Navigation Model (MANM) used for effective and efficient web navigation.

Web Accessibility: A Multi-Axial Navigation

Representation is in the mind of the beholder.

— Winograd & Flores, *Understanding Computers and Cognition*, 1986

Page mobility is by far the hardest issue to address, due to the normally high complexity of visual information found in a hypermedia page or document.

— Simon Harper, Carole Goble & Robert Stevens, *Journal of Research and Practice in Information Technology*, 2001

It is commonly known that web pages are primarily authored with the sighted user in mind and X/HTML was designed as a visual formatting language. Therefore, navigation on the Web typically boils down to user interactions within the browser and in regards to sighted users, navigation is primarily defined as hypertext navigation by selecting hyperlinks. While this is acceptable for the click-and-drag paradigm of modern graphical user interfaces, navigation interactions are very costly for the blind user who does not have access to a mouse-centred interface. Navigation for blind users primarily depends on both hypertext navigation and scanning navigation, which corresponds to eye movements for the sighted [226]. Eye movements for “scanning” pages including peripheral vision enable sighted users to be able to quickly familiarize themselves with web pages in addition to perceiving and analyzing text by the use of skimming, various visual features, and spatial configurations. In contrast, blind users scan web pages by the relatively small chunks of text which they obtain by means of typhlotechnology (e.g., a screen reader and Braille display) and navigation commands (mainly keystrokes). Moreover, navigation and understanding of a web page is greatly exaggerated by the dependency of commonly used screen readers on the technical layers of web pages which often do not reflect the logical structure

of a web page. Thus, blind users have to grasp the content solely based on HTML tags provided by web authors.

In this chapter, we consider the challenge of enhancing web page accessibility by taking into account the peculiarities of contemporary typhlotechnology, which merely conveys one-dimensional navigation with aural and tactile output, as well as the recent developments of the author presented in this thesis by focusing specifically on the Unified Ontological Model (UOM) (see Chapter 4) and Web Page Processing System (WPPS) (see Chapter 5).

In Section 6.1, we identify the main types of objects within the UOM which should be accessible for blind users and describe the main peculiarities of blind users' web navigation in addition to the aspects of blind users' mobility which should be considered for enhancing web accessibility. The enhancement is related to the development of the Multi-Axial Navigation Model (MANM) (one of the main contributions of this thesis) providing various content serializations and transitions which is discussed in Section 6.2. Section 6.3 presents the main procedure and principles which should be taken into account for instantiating the MANM. Section 6.4 presents another important contribution for ameliorating web accessibility: the methodology of navigation over the MANM. The proposed solutions are realized in Blindzilla prototypes which are presented in Section 6.5 and evaluated in Section 6.6. The experiment provides evidence of the efficiency and effectiveness of the proposed concepts. Section 6.7 concludes the chapter.

The development of the MANM and navigation methodology proposed by the author of this thesis is based on the cooperative work within the scope of the ABBA project [236] and survey presented in Appendix B. The most significant studies which build the foundation of the research presented in this chapter include [21, 83, 85, 151]. For the state of the art within the field of Web Accessibility and relevant works, we refer the interested reader to Sections 2.1 and 2.2.

6.1 Ameliorating Blind Users' Mobility

To recap, we described the main methods and approaches of web page navigation available in contemporary screen readers (see Section 2.2.1) and also considered prominent works analyzing the application of the metaphor of spatial navigation in physical space to the information space of electronic documents (see Section 2.2.2). We also highlighted the main shortcomings of the contemporary implementations (see Section 2.2.3). Thus, among the existing issues of web accessibility, the most important and relevant to this thesis include the absence of semantically rich content annotations and the insufficient effectiveness of navigation methods in contemporary screen reading technology.

As we can see in the example illustrated in Figure 6.1, screen readers provide quite limited descriptions of web page elements mainly relying on HTML (e.g., tag `A` is interpreted as link, `IMG` as graphic, and `TABLE` as table). Thus, it is more difficult to navigate and understand web pages not compliant with accessibility guidelines (for example, HTML tags are misused and ARIA roles [259] are not applied). Unfortunately, most of today's web pages do not follow accessibility guidelines and standards [29, 137] which determines the importance of developing transcoding technologies (see Section 2.1.3) to transform web pages into more accessible representations. It is important to note that WPPS introduced in Chapter 5 can also be utilized as a transcoding system providing the blind user with a rich web page model.

Visual representation



Aural representation

cooking forum dot net - friendly cooking discussion forum
[<link>](#) [<graphic>](#) [<header>](#) logo [<link>](#) home [<link>](#)
[<link>](#) register [<link>](#) today's posts [<link>](#) community [<link>](#)
[<link>](#) search [<link>](#) [<graphic>](#) titan peeler you can search the
forums by entering your criteria here:...

[<table with 3 columns and 3 rows>](#) username:
added user name. remember me? [<link>](#) [<graphic>](#)
password: log in enter your username and password in
the boxes...

cookingforums dot net. notices we have detected that you
don't have an account...

[<table with 5 columns and 16 rows>](#) forum last post
threads posts [<link>](#) general [<link>](#) general cooking
discussion discuss all general cooking discussion here so
simple so good by [<link>](#) creekcorner 56 minutes ago
[<link>](#) [<graphic>](#) last post 519 6870 [<link>](#) celebrity
chefs and cooking on TV discuss celebrity chefssuch as
Rick Bayless, Emeril Lagasse, as well as cooking television
shows such as the Iron Chef here [<graphic>](#) question
mark [<link>](#) food after dark by [<link>](#) creekcorner 9 hours
ago [<link>](#) [<graphic>](#) last post 72 815 [<link>](#) baking
discuss everything baking from bread to cake here [<link>](#)
"egg bread" vs "brioche..." by [<link>](#) awesomeapril 4 hours
ago [<link>](#) [<graphic>](#) last post 135 1179 [<link>](#) health
and nutrition discuss everything from calories, exercise, and
saturated fats here [<link>](#) how to get your kids to eat... by
[<link>](#) lenovo11022 1 hour ago [<link>](#) [<graphic>](#) last post
77 1025 [<link>](#) tips and techniques discuss everything...

Figure 6.1: Visual perceptions of a web page and aural perception with the use of a typical screen reader reading through the page

6.1.1 Navigable Web Page Objects

The application of web page processing (WPP) methods implemented in the WPPS framework results in instantiating the UOM comprising various constructs for describing different aspects of a web page. Figure 6.2 illustrates an example of various functional and structural elements of a web page and some geometric relationships which can be recognized by the use of WPPS and expressed by means of the UOM.

We distinguish the following types of objects of the UOM which we believe should be accessible for blind users and used in navigation when it is necessary: *geometric*, *interface*, and *logical objects*, which we in turn divide into *data structures*, *web* and *domain specific objects* as well as *textual objects*. All of them correspond to different layers of the *web page conceptualization* introduced in Section 4.2 (see Figure 4.3 on page 106). **Geometric objects** are part of the Block-based Geometric Model (BGM) (see Section 4.4.2) [73] and refer to the *geometric layer* of the *conceptual model of a web page*. They represent various visible elements of a web page modeling them by the minimum bounding rectangle (*block*, see Definition 3.5 on page 67). There are several types of spatial relationships defined on the set of blocks: topological, direction, distance, alignment relationships, and so on (see Sections 3.5 and 4.4.2). **Interface objects** are elements of the Interface Model (see Section 4.4.3) and refer to the interface layer of

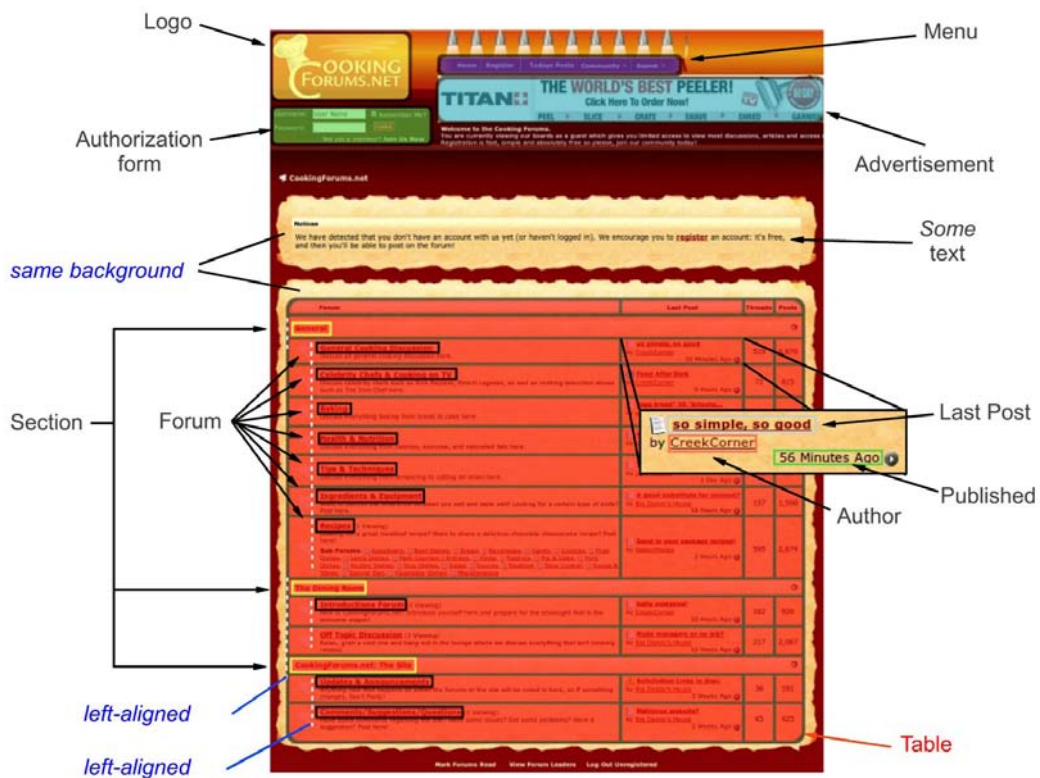


Figure 6.2: Semantic annotation of a web page content for enhancing its accessibility

the web page conceptualization. It represents various basic functional objects derived due to the analysis of the DOM tree and CSSOM, for example: web forms, text input fields, buttons, links, images, lists, tables and so on. *Logical objects* in turn correspond to the relevant logical layers and are modeled by the Logical Model (LM) (see Section 4.5). Logical *data structures*, such as sequence, tree and table grid, can be used for representing the structure of the relevant objects recognized on a web page. For example, a navigation menu can be represented as a sequence, and a news article with its sections and subsections, can be mapped into the tree. In regards to web accessibility, these mappings of web objects into the data structures enable intuitive ways of navigating them. According to the survey presented in Appendix B, most screen reader users are familiar with table and list navigation (see Section B.5). *Web specific* and *domain specific objects* model various logical objects (e.g. logotype, navigation menu, authorization form, etc.) of a web page by means of domain ontologies relevant to specific genres. *Textual objects* correspond to the layer of textual content semantics and in regards to navigation, are represented by named entities, sentences, tokens, and letters. The importance of considering “word-level design of content” is discussed in [226, page 29].

Regarding geometric objects and visual features (e.g., foreground and background colors, font style and size from the *interface layer*), we believe that they should be only used by blind users in exceptional cases, for example, to interact with the sighted assistant. As was discovered

in B.3, many blind people (40%) assert that familiarizing themselves with new websites with the aid of a sighted assistant is not easier than independent learning. This is mainly due to operating with different representations of a web page (i.e., the visual and aural) and using dissimilar terminology. We demonstrate this difference in Figure 6.1. Thus, blind users do not use visual cues and mainly leverage various serializations of a web page content which screen reading tools convey. Therefore, geometric features can be applied to reduce this semantic gap.

Some aspects of these accessible types of objects (navigable objects) can be expressed using W3C guidelines. For example, contemporary screen readers support various HTML types of elements with their properties (e.g. images with alternative text, web form elements with labels attached) and inner structures (e.g. HTML tables, HTML lists, HTML forms, etc.). Furthermore, the logical structure of a web page with various data structures (such as tree, table, and list) can be to some extent described by means of ARIA roles presented in [259, Sec. 5.3]. As reported in the inquiries conducted by WebAIM [293, 295, 296], most sightless users are aware of ARIA-based accessibility (e.g. landmarks) and can leverage it. However, these common standards are quite limited. They are not able to express various semantic relations between elements (e.g., semantic relation between a news article and comments relevant to it). Moreover, they are integrated into the source code of a web page as HTML attributes and thus dependent on the semantics established by the DOM tree structure. This limits automatic annotating web pages and is the reason why we propose a separate model (the UOM) as well as modeling a whole web page together with its different aspects, but which can still be incorporated into the web page source code by means of RDFa.

From the mobility point of view, the main goal of introducing these types of objects, which are also *mobility objects*, is to decrease the amount of *obstacles* and increase the amount of *cues*. As it is defined in [112], obstacles are primarily “objects that inhibit a users onward journey,” whereas cues are “objects or combinations of objects that a traveller actively uses to facilitate their onward journey.” Furthermore, providing the user with logical objects reflecting both web page structure (with logical data structures and web specific objects) and its domain (with domain specific objects) can help to understand the role of considered objects as well as remember and distinguish them from the rest.

6.1.2 Web Page Mobility

With the exception of mobility objects, the mobility of blind users is defined by the set of possible interactions as well as leveraged mobility techniques [97, 112]. Due to the aural output, contemporary screen reading technologies provide sequential (one-dimensional) access to the web page content (see Section 2.1.3) and therefore primarily one-dimensional character of navigation through various serializations of the content. These serializations generally correspond to the depth-first traversal over the DOM tree and can be represented, for example, by the list of links, headings, paragraphs, or web form elements. Navigation mainly comprises locomotion through different types of basic web page elements (primarily, HTML elements) going forward or backward to the object of the required type. Observation mainly includes obtaining the type of current web page element, probing (making a few locomotions), and obtaining some statistics regarding the number of objects of different HTML types presented on a web page. Thus, we can assert that the mental load of sightless users is mainly related to building a mental model of a

web page from its source code using HTML-based navigation.

In contrast, we focus on utilizing navigable objects introduced in Section 6.1.1 as a basis for building different serializations. As such, we believe it is important to switch from HTML-based or layout-based navigation to navigation through the logical model of a web page reducing the mental load of users. Furthermore, taking into account the metaphor of spatial navigation, we found it important to consider the problem of user mobility from the viewpoint of one-dimensional navigation through multidimensional information space (i.e., set of mobility objects with different attributes and relationships). Therefore, it is important to consider various serializations of the page content as certain mobility objects which can be used for orientation and in the selection of the most relevant path to the interesting content.

In addition, utilizing a concept of web mobility introduced in [97, 112] (see Section 2.2.2), the enhancement of web accessibility both from the viewpoint of interaction and underlying model should concern the following aspects:

Mobility objects:

(Cues, obstacles, memories) Incorporate types of navigable objects introduced in Section 6.1.1.

(Out-of-view) Decrease the amount of objects which cannot be reached within the short time interval (i.e., so-called *out-of-view objects*).

Mobility techniques:

(Preview and Probing) Provide effective techniques for the preview and probing by the blind user to obtain information regarding the surrounding content and be able to correctly choose the next locomotion according to the task fulfilled or aim posed.

(External Memory) A navigation model should provide blind users with easily memorable and recognizable objects and landmarks which can be used to both understand the current position and plan further navigation without probing.

Mobility principles:

(Information flow) Information flow formed during the interaction of a blind user with a web page should be detailed enough to understand the current context (i.e., what is read and which logical structure it has).

(Granularity) Landmarks should be easy to find and close enough together.

(Egocentricity) Anytime a sightless user has the possibility to obtain feedback regarding her position and be able to orient herself relative to her current position. In other words, the user should be able to obtain orientation by means of deictic frame of reference [50, 205].

(Memory) A blind user should be able to build a mental model relevant to a web page.

(Regularity) Mobility objects should be deployed in a regular manner.

(Spatial) A sightless user should be able to leverage its skills of navigating in physical space.

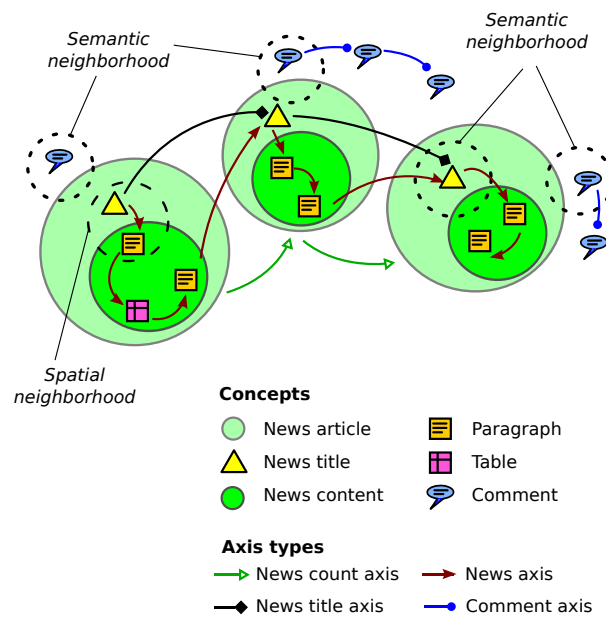


Figure 6.3: Example of the navigation model for a news web site

6.2 The Multi-Axial Navigation Model

The Multi-Axial Navigation Model (MANM) [79] realizes aspects and requirements highlighted in Section 6.1 for enhancing web page accessibility and a blind user’s mobility. It corresponds to the upper level of the conceptual model of web pages presented in Section 4.2 (see Figure 4.3 on page 106) providing an accessible representation [21, 74, 83, 85, 151]. Furthermore, the MANM is built on top of what we call the Unified Ontological Model (UOM) (see Section 4) and from the viewpoint of WPP, it is a result of the transformation of an instance of the UOM, as presented in Section 5.1.

6.2.1 Main Concepts

The MANM provides necessary constructions for modeling navigable objects (i.e., geometric, interface, and logical objects) and their different serializations represented by axes and their types. An *axis type* represents the abstract rule for reading certain concepts of the underlying model (i.e., the UOM) and thus defines a rule for ordering particular concepts. We distinguish four main types of axes: an interface axis (it mainly goes through interface objects), structural (logical) axis (it mainly includes logical objects), content axis (it mainly comprise textual objects), and basis axis which serializes all elements on a web page. An *axis* is an instance of an axis type and therefore serializes a certain subset of relevant objects. Serializations take into account various aspects of a web page representation provided by the UOM including visual and logical features.

In Figure 6.3, we give an abstract example of navigational axes within a hypothetical news site. The depicted model represents the main concepts (logical objects) of the news domain such as “news article,” “news title,” “content,” and “comment.” As part of a logical representation of web

pages, these concepts can have a complex structure on their own and include further sub-concepts such as news articles that can contain sections, paragraphs, tables, and figures. Spatial and semantic neighborhoods which define relationships between navigable objects including axes are also two important concepts. The model illustrated in the figure also contains four types of axes: “news title axis,” “news axis,” “comment axis,” and “news count axis.” Axes types applied for the model can have one instance (e.g., “news title axis”) or several (e.g., “comment axis”), or not at all. An axis can have arbitrary quantity of elements.

There are two types of relations which define the neighborhood in the multi-axial model: spatial and semantic. A spatial relation is based on relative distance between objects and defined in the UOM. The example can be the spatial relation between news title and a paragraph which is the first paragraph of the news article (see Figure 6.3). Semantic relations can be set either for objects in the underlying model or for axis types in an instance of the MANM. For the semantic relation defined between the axis types, the semantically nearest object of the particular axis will be the object of another axis which is spatially nearer than other ones of the same axis. For instance, a relation between “news axis” and “comment axis,” and a relation between news title and news content is an example of semantic relations.

To reflect the geometric allocation of objects, the MANM also supports non-symmetric relations based on the orthogonal visibility [84, p. 12] (see Figure 4.16 on page 120 and Figure 4.17) such as: “up” (leftmost north orthogonally visible object), “right” (topmost east orthogonally visible object), “down” (leftmost south orthogonally visible object), and “left” (topmost west orthogonally visible object).

6.2.2 Example of a Navigation Model

To illustrate an example of applying a navigation model for a web page, we considered different relevant navigation trails which can be provided for a news page by example of a CNN web page and presented them in Figures 6.4 and 6.5. We can see different structural axes going through different web page elements (i.e., navigable objects). For example: “navigation menu 1” and “navigation menu 2” which go through items of navigation menus; “categories 1” and “categories 2” which comprise news categories; “news list 1” and “news list 2” ordering news headlines. Aside from logical objects, we can see interface objects such as an image, button, and link. In addition, there are objects defining a certain layout such as a list and table which in turn should be accessible for navigation by means of functions similar to that provided by contemporary screen readers (e.g., JAWS, NVDA, and Window-Eyes). Thus, we distinguish between navigation with the use of axes and according to a certain data structure of an object (i.e., table, tree, and list).

Figure 6.6 illustrates the presented example regardless of web page layout. It is evident that the order of elements of provided navigation trails for the consistency should correspond to a serialization set by the basis axis. We also illustrated similarity relation between unrecognized segments presenting akin information (about the weather). We also reflected similarity relation between logical objects of navigation menu and news categories with the same name.

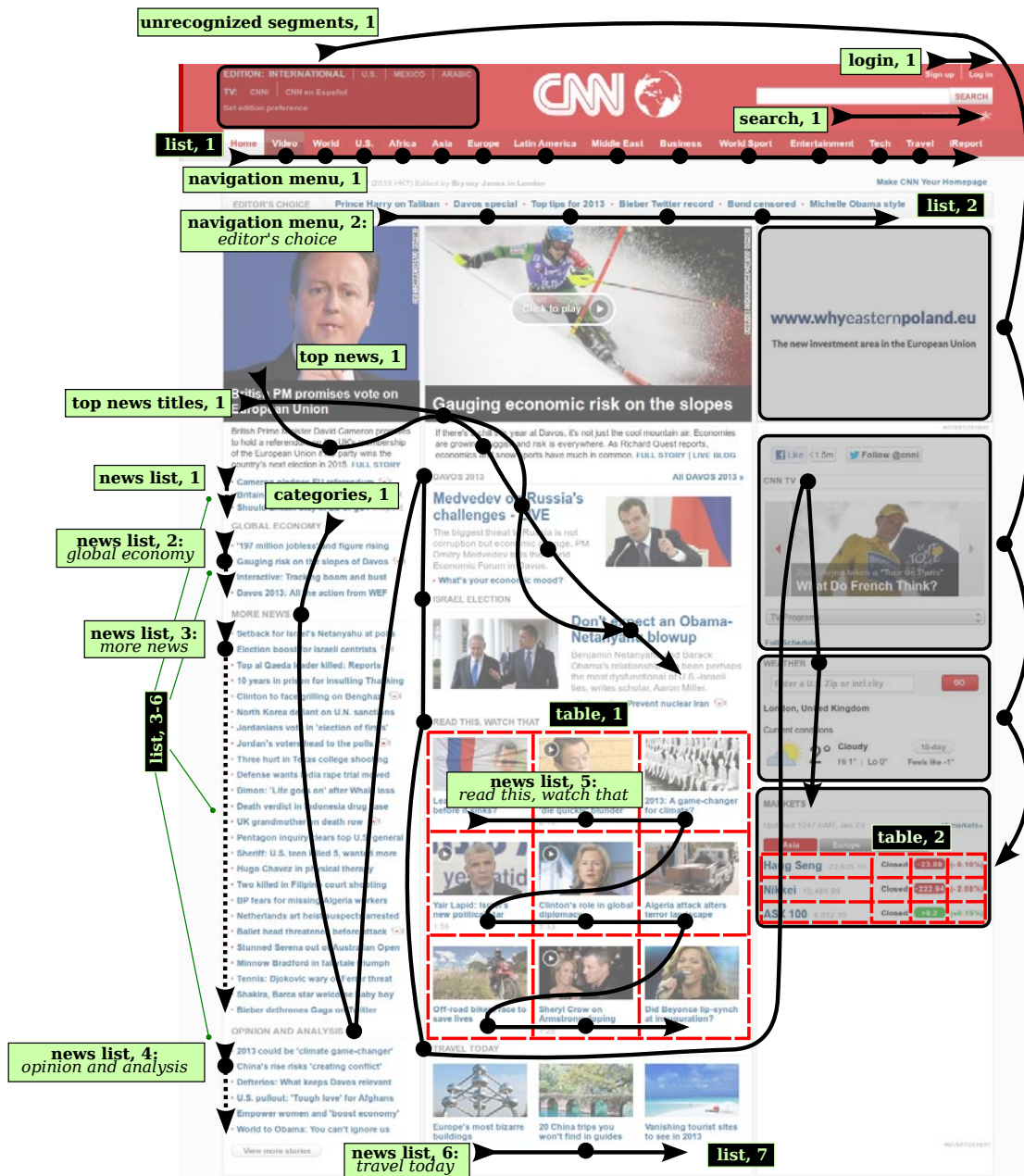


Figure 6.4: Example of the Multi-Axial Navigation Model of a CNN web page visualized on a web page canvas (part 1)

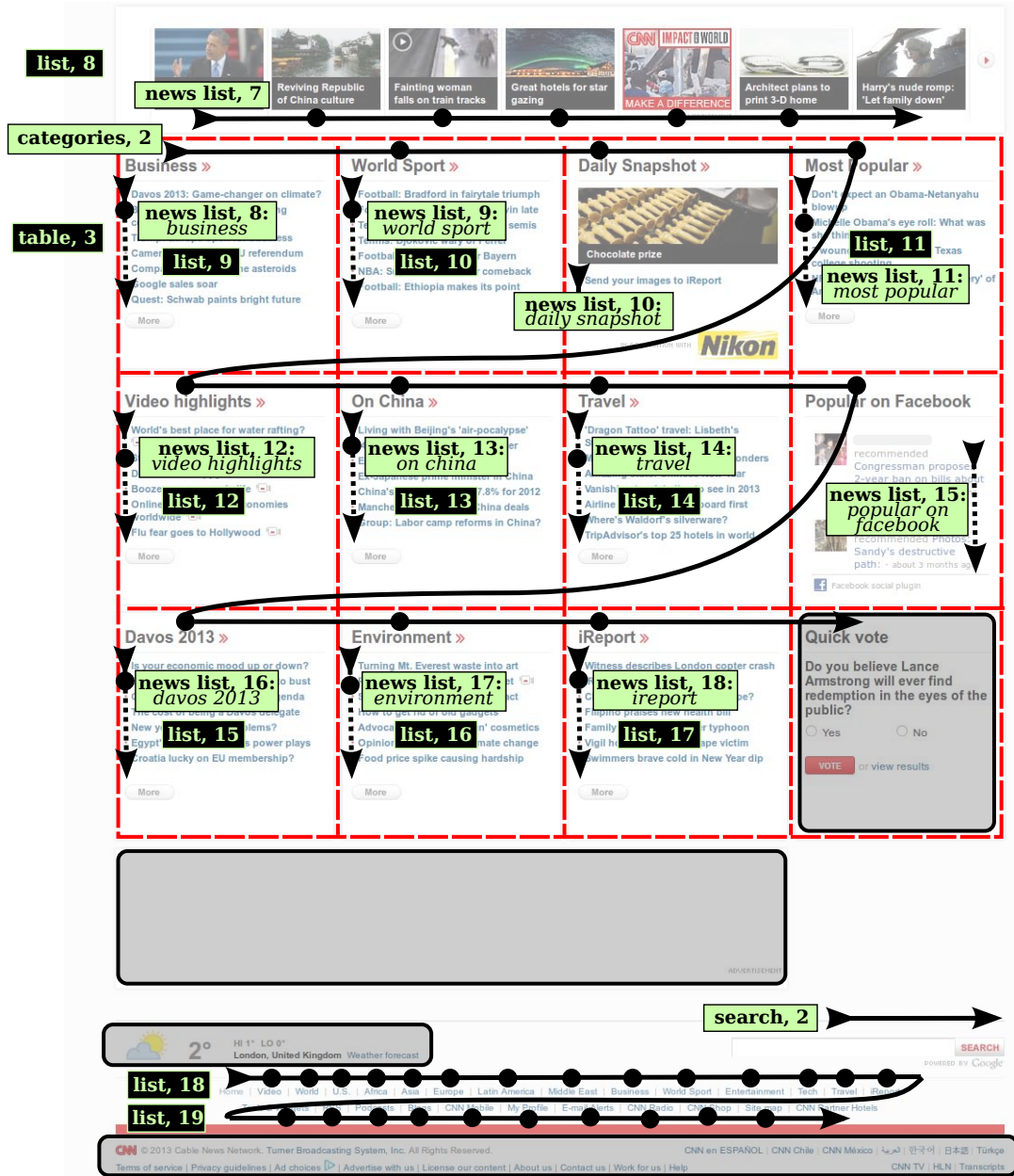


Figure 6.5: Example of the Multi-Axial Navigation Model of a CNN web page visualized on a web page canvas (part 2)

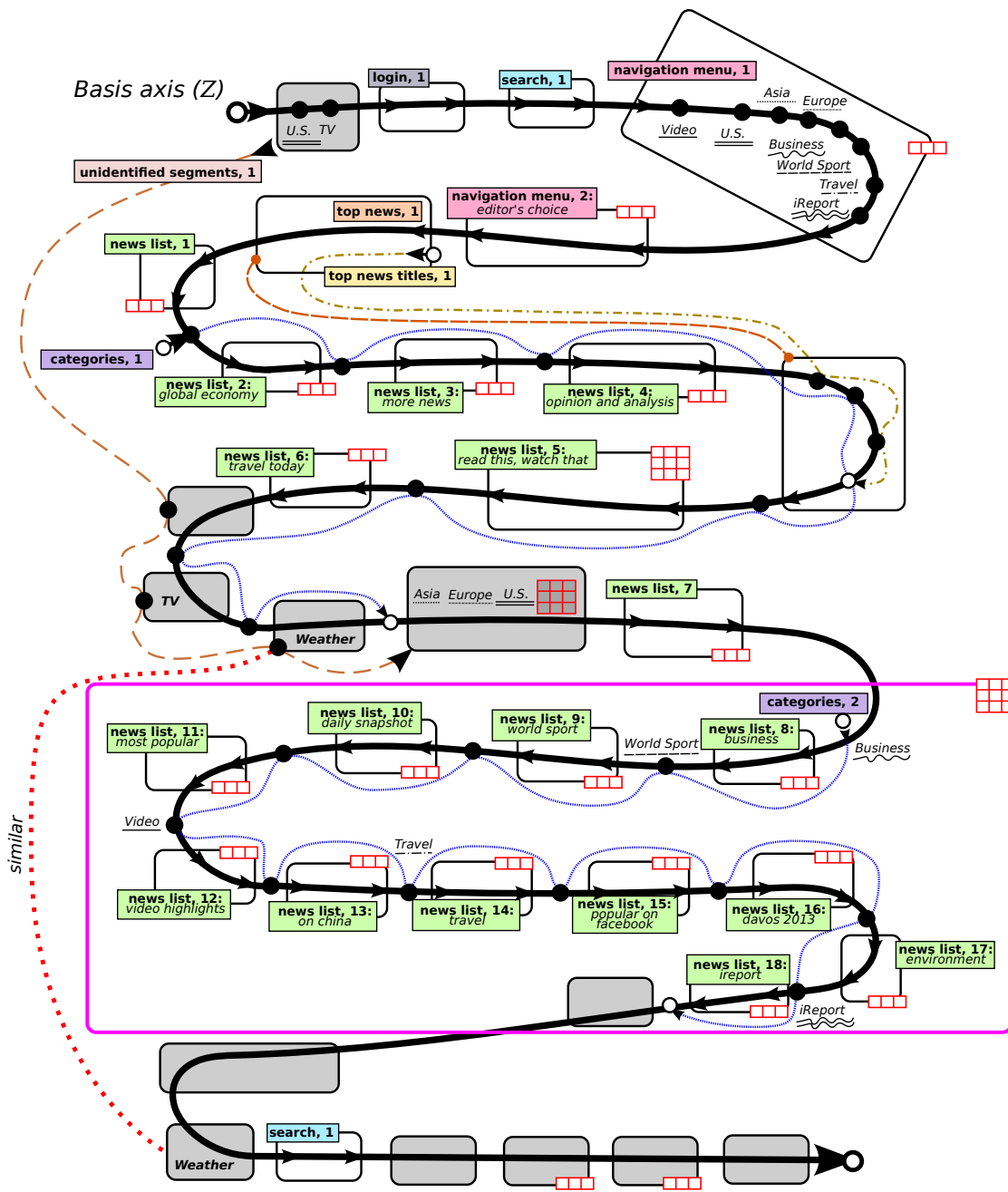


Figure 6.6: Example of the Multi-Axial Navigation Model of a CNN web page

6.2.3 Formal Presentation of Axes

An *axis* $A \in \mathcal{A}$ is a sequence (a_1, a_2, \dots, a_n) of basic navigable objects of the UOM, where $n \geq 0$ is the *length* of the axis A and \mathcal{A} is a set of all axes instantiated in the MANM [21, 85, 151]. If $n = 0$ then A is *empty*. $A(i)$ is the i -th element of A . $|A|^-$ is a set of elements of A . $|A|^+ \equiv |A|$ is a multiset of elements of A . Thus, in the realization of the navigation model in the form of the MANM, we consider axis items as basic objects. An *axis type* T refers to the set of axes which share the same semantics or are generated using the same algorithm. The axis type represents an abstract rule for reading certain concepts of the underlying model. There is a functional relation between axis and axis type. \mathcal{T} is a set of all axes types in the instance of the MANM. Axis type should have a meaningful name which reflects its purpose or name of corresponding navigable objects. An axis A is identified by the pair $\langle T, id \rangle$ of its type T and its sequence number id in the sequence of all axes of the same type: $\forall B \in \mathcal{A} : type(B) = type(A) \wedge id(B) = id(A) \rightarrow B = A$.

Base axis Z is an axis which order all basic navigable objects of the instance of the LM.

Features of Axes

We define the following features of axes in the MANM: *length*, *location*, *coverage*, *direction*, and *global direction*.

Length $n = \|A\| = |A|^+$ of an axis A is a quantity of elements in corresponding sequence subject to repetitions. Length gives an estimation of the axis extent which corresponds to the time needed to read an axis. We also utilize features such as $\|A\|^{word}$ and $\|A\|^{sent}$ which define quantity of words and sentences within the axis A respectively. If an axis corresponds to a certain logical object, length gives an estimation of its size from the viewpoint of the time required for reading it.

Location of an axis A in the MANM is a sequence number of the first occurrence of its first element $A(1)$ on the base axis Z : $pos_{\frac{1}{2}}^1(A(1))$. Location can also be a relative value: $(pos_{\frac{1}{2}}^1(A(1)) - 1) / (\|Z\| - 1)$ for $\|Z\| \geq 2$, 0 otherwise.

Coverage of axis A is the ratio of the quantity of unique navigable objects of A against the quantity of all navigable objects of a web page (of the instance of the UOM). It gives one an estimation of the amount of information contained within the axis relative to all information on a web page.

Spatial direction (*direction*) defines a prevail direction relationship between pairs of successive elements. If the direction relation is “east-of” or “west-of,” the spatial direction of an axis is horizontal (“left-to-right” or “right-to-left” respectively); if the direction relation is “south-of,” “north-of,” the spatial direction is vertical (“top-down” or “bottom-up” respectively). These directions are the most ubiquitous relationships in practice due to the peculiarities of information organization, web page layout (which in most cases is kept to “Manhattan” [110] or “near-Manhattan” [116, p. 19] layouts), and reading order. Employing quantitative information we define axis direction as an angle between abscissa and straight line interpolating locations of geometric centers of objects and averaging existing directions between successive objects of an axis. If this interpolation is impossible and directions are too diverse, an axis does not pose any spatial direction—it has an *undefined* direction.

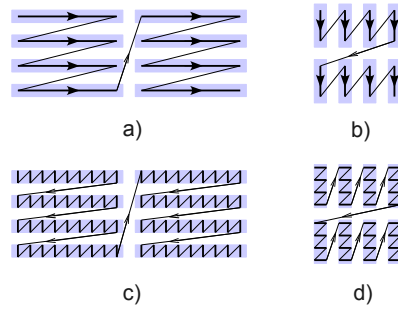


Figure 6.7: Main global directions of axes: a) horizontal, b) vertical, c) horizontal with deviation, d) vertical with deviation

Spatial global direction (*global direction*) defines a predominant direction of sub-sequences of an axis (see Figure 6.7). Considering the huge amount of various layouts of web pages, we single out four main global directions: *horizontal*, *vertical*, *horizontal with deviation*, and *vertical with deviation*. Together with these features, we define concepts such as *amount of columns* and *rows*. Horizontal and vertical global directions have a layout similar to homonymous directions with deviations. The only difference is that the latter allows elements to deviate from the current direction. The horizontally-oriented relations imply the presence of several objects in each line of a column while the vertically-oriented imply the presence of several objects in each vertical traversal of a row. Global direction reflects a spatial configuration of elements and their layout. For instance, directions in Figure 6.7a (e.g., axes “news list 5” in Figure 6.4 and “categories 2” in Figure 6.5 forming one column) and Figure 6.7c usually correspond to the multicolumn layout, while directions in Figure 6.7b (e.g., axis “categories 1” in Figure 6.4 forming one row) and 6.7d can be associated with a reading order for vertical lists. Also, all these directions can define a reading order for a tabular layout. Thus, axis and layout are interrelated: an axis gives one information about layout, while the layout affects the global direction of axes.

For an analysis of the MANM, we propose the following statistical characteristics: *set of repetitive elements* $\rho_1(A)$, *the most frequent repetitive elements* $\rho_1^{max}(A)$, *repetitive sequences of the length n* $\rho_n(A)$, *the most frequent repetitive sequences of the length n* $\rho_n^{max}(A)$, *the longest repetitive sequences* $\rho_{max}(A)$, and *the most frequent repetitive sequences* $\rho^{max}(A)$. These characteristics give information about structural singularity of an axis, particularly about repetitions and cycles. Repetitions of elements are features dual in nature. On the one hand, these reflect the importance of certain elements since axes with repetitions of items can be used for content difficult to understand or for learning. On the other hand, these statistical characteristics can be used to check the correctness of an axis and the generation process. For example, if a certain axis is not allowed to have repetitions of elements, than their presence will signal the incorrectness of the algorithm generating this axis accordingly.

The basis axis Z should not have any repetitions: $\rho_1(Z) = \emptyset$.

Relations of Axes and Functions

For the objects a and b from the UOM, we define a *closeness* relationship which reflects the similarity of objects relative to some particular feature or set of features. Examples of these relationships expressed via quantitative characteristics are **spatial** $d(a, b) \geq 0$ and **semantic closeness** $0 \leq w(a, b) \leq 1$. A *spatial closeness of objects* is expressed by their spatial distance. A relation of *semantic closeness of objects* is defined based on the LM.

According to these relations, we define concepts (functions) of *neighborhood* $\nu(a, \varepsilon)$ and particularly **spatial** $\nu_{sp}(a, \varepsilon_{sp})$ and **semantic** $\nu_{sem}(a, \varepsilon_{sem})$ *neighborhoods* for an element a respectively:

$$\nu(a, \varepsilon) \equiv_{def} \{b \mid \varepsilon(a, b)\},$$

where $\varepsilon(a, b)$ is a predicate which reflects a *closeness* relationship between elements,

$$\nu_{sp}(a, \varepsilon_{sp}) \equiv_{def} \{b \mid d(a, b) \leq \varepsilon_{sp}\},$$

$$\nu_{sem}(a, \varepsilon_{sem}) \equiv_{def} \{b \mid w(a, b) \geq \varepsilon_{sem}\}.$$

We also define functions such as *set of corresponding axes* $ax(a)$ for an element a and *set of positions* $pos(a, A)$ of an element a within the axis A :

$$ax(a) \equiv_{def} \{A \mid A \in \mathcal{A} \wedge a \in A\},$$

$$pos(a, A) \equiv_{def} \{i \mid 1 \leq i \leq \|A\| \wedge A(i) = a\}.$$

Furthermore, $pos^j(a, A)$ denotes position of the j -th occurrence of a in A .

Given the axis A and element index i , a general concept of *neighborhoods* and a *spatial* and *semantic neighborhoods*, which correspond to the aforementioned homonymous concepts, are defined respectively as follows:

$$\nu^{ax}(A, i, \varepsilon) \equiv_{def} \{ \langle B, j \rangle \mid B \in \mathcal{A} \wedge B \neq A \wedge 1 \leq j \leq \|B\| \wedge B(j) \in \nu(A(i), \varepsilon) \},$$

$$\nu_{sp}^{ax}(A, i, \varepsilon_{sp}) \equiv_{def} \{ \langle B, j \rangle \mid B \in \mathcal{A} \wedge B \neq A \wedge 1 \leq j \leq \|B\| \wedge B(j) \in \nu_{sp}(A(i), \varepsilon_{sp}) \},$$

$$\nu_{sem}^{ax}(A, i, \varepsilon_{sem}) \equiv_{def} \{ \langle B, j \rangle \mid B \in \mathcal{A} \wedge B \neq A \wedge 1 \leq j \leq \|B\| \wedge B(j) \in \nu_{sem}(A(i), \varepsilon_{sem}) \}.$$

A relationship of *semantic closeness for axes types* T_1 and T_2 is expressed quantitatively as $0 \leq w(T_1, T_2) \leq 1$ and defined in the MANM. A concept of *semantic neighborhood of an axis type* T with a threshold ε_{sem} is as follows:

$$\nu_{sem}(T, \varepsilon_{sem}) \equiv_{def} \{T' \mid T' \in \mathcal{T} \wedge T' \neq T \wedge w(T, T') \geq \varepsilon_{sem}\}.$$

Spatial axial intersection (*axial intersection*) for i -th element of axis A is a set of pairs $\langle B, j \rangle$ with intersecting axis B and an index j of the equivalent element:

$$\chi(A, i) \equiv_{def} \{ \langle B, j \rangle \mid B \in \mathcal{A} \wedge B \neq A \wedge 1 \leq j \leq \|B\| \wedge B(j) = A(i) \}.$$

$$\chi(A, i) = \nu^{ax}(A, i, \varepsilon), \text{ where } \varepsilon(a, b) \leftarrow a = b.$$

Axial concatenation (*concatenation*) of axes A and B is an operation which joins them into the new axis C such that the first element of B is a successor of the last element of A , $C = A \cdot B$.

Algorithm 6.1: Process of building the MANM

Input : The instance of the WPPS framework (`wppsFramework`).
Configuration of the MANM (`manmConfig`).
A set of types of HTML tags which are considered as readable (`readableTypes`).

Output: An instance of the Multi-Axial Navigation Model (`manm`)

```
1 manm ← BuildEmptyMANM (manmConfig);
2 wppsAPI ← GetAPI (wppsFramework);
3 segmentationTree ← Segmentation (wppsAPI, readableTypes);
4 BuildBasisAxis (segmentationTree, manm);
5 AddSpatialNeighbors (wppsAPI, readableTypes, manm);
6 AddLogicalObjects (wppsFramework, manm);
7 AddSemanticNeighbors (wppsAPI, manm);
8 AddInterfaceAxes (manm);
9 AddStructuralAxes (manm);
```

6.2.4 Ontology

The ontological representation of the MANM [79] is compliant with the description presented in Section 6.2.3. Classes and object properties are presented in Figures 6.8 and 6.9. There are four functional types of axes: basis, interface, structural and content which is used to order named entities (`NEAxis`) and sentences (`SentenceAxis`). Set of axes are grouped into corresponding axis types. An axis is identified by the name of the corresponding axis type and sequence number of the axis within the group of axes of the same type. Additionally, we distinguish two types of axes depending on which type of element of the visual formatting model [260, Sec. 9] constitutes an axis; block-level elements form block axis, whereas inline-level elements form inline axis.

Furthermore, the ontology supports three functional types of objects of a web page content building the similarly named axes: interface (a superclass of an element of the Interface Model, see Section 4.4.3), structural and content elements, where the latter is subdivided into named entities (`NE`) and sentences (`Sentence`). A basic object represents navigable object forming axes of the MANM (i.e., structural element forming a basis axis should be a basic object). Elements have corresponding data properties which specify their type (e.g., `interfaceType`, `structuralType`, `neType`) and textual content (e.g., `stringContent`). A basic object also has an id within the basis axis (`zId`).

6.3 Building the Multi-Axial Navigation Model

The process of building the instance of the MANM is given in Algorithm 6.1. It requires the following objects to be provided: a) an instance of the WPPS framework which conveys functionality necessary for interacting with generated UOM, b) configuration of the MANM (e.g. how the model should be represented in the Jena ontology framework [10]), and c) a set of types of HTML tags which should be considered as readable (i.e., elements which can be represented by text which describe their content). It mainly consists of elements of a web form, alternative text of images, and visible textual content.

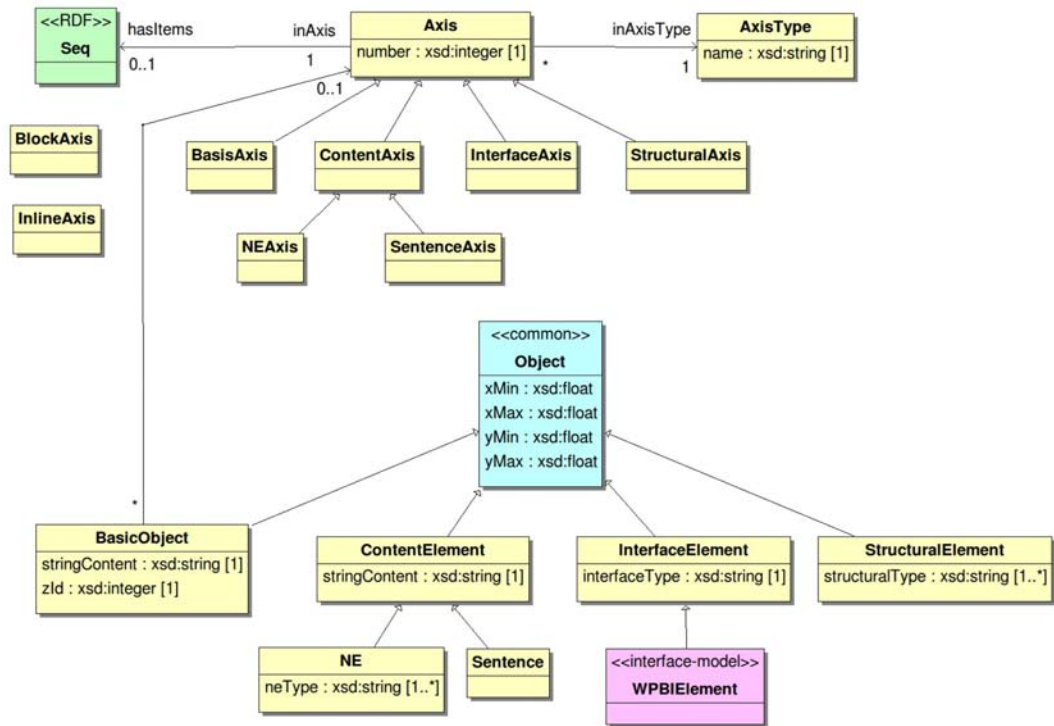
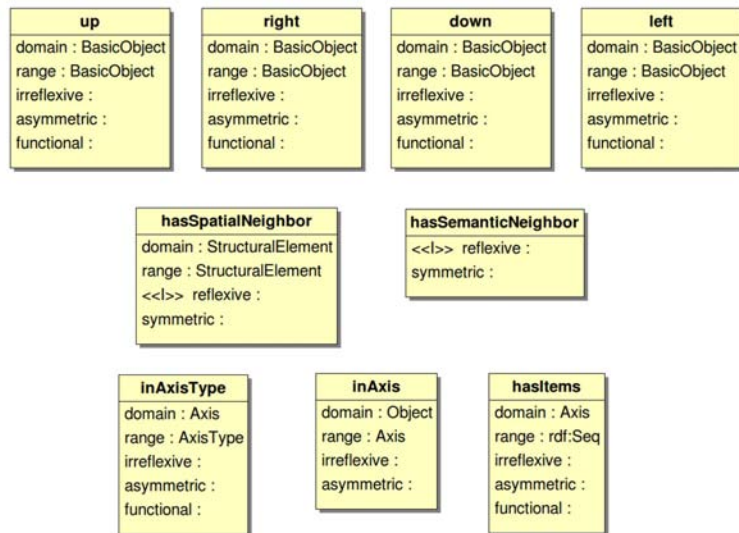


Figure 6.8: Classes of the Multi-Axial Navigation Model



<<|>> - Inherent property

Figure 6.9: Object properties of the Multi-Axial Navigation Model

The first two procedures are building the empty MANM (line 1), which is filled later with assertions corresponding to a certain web page, and acquiring a WPPS API for querying the UOM (line 2). For building the basis axis Z , we propose the use of a segmentation algorithm (e.g., XY-cut [106, 176, 182], VIPS [40, 41], or [104]) to segment a web page based on visual cues on logically consistent parts (line 3) and, in the next procedure, to serialize a segmentation tree reflecting the correct reading order (line 4) [2, 172, 197]. Various experiments by example of Blindzilla2 (see Section 6.5) and interviews of sightless users show the effectiveness of such an approach. Readable elements of this serialization represent the basis axis. Identification of spatial neighbors (line 5) is performed for every readable element based on the value of *spatial closeness* provided (see Section 6.2.3). It has general complexity $O(n^2)$ which can be improved by the application of R-Tree integrated into WPPS. Leveraging predefined wrappers, different objects on a web page can be identified and annotated, providing the user with information regarding the logical objects and logical structures on a web page (line 6). The wrappers applied are also responsible for defining *semantic closeness* between logical objects which is materialized in the MANM as a relationship of *semantic neighbor* (line 7). Identified elements are then serialized by means of axes. Interface axes (line 8) order different types of interface objects, whereas structural axes (line 9) order logical (structural) objects of a web page. As it was experimentally confirmed, interface and structural axes should correspond to the reading order established by the basis axis.

6.4 Methodology of Navigation

*“Begin at the beginning,” the King said gravely,
“and go on till you come to the end: then stop.”*

— Lewis Carroll, *Alice in Wonderland*

Operations from Relational Algebra, First-Order Logics, and functions presented in Section 6.2.3 are used in this section for the sake of formal representation of navigation commands.

An instance of the MANM generated for a web page provides a graph-based structure which comprises necessary information about elements of the model (axes, navigable objects, attributes and relations) ensuring a stable and efficient navigation (see Section 6.2). Taking into account Information Foraging theory [194], cognitive approach to navigation [112, 132, 226], and principles of the user mobility [97, 112], we single out two main concepts: *observation* and *locomotion*.

6.4.1 Observation

Observation plays an important role in the process of orientation. It helps users to recognize their location based on information about surrounding objects. In this case, well-known and recognizable objects (referred to as *landmarks*) are the main reference objects for understanding the user’s whereabouts and planning a navigation path. We define two main types of observation: *in-axial* and *inter-axial*.

In-axial observation allows one to obtain information within the scope of one axis. It includes:

- axis type, axis identifier;
- quantity of passed and residuary items of the axis;
- sequential number of the current item and its type according to the LM;
- features of an axis such as, length, location, coverage, direction, etc. (see Section 6.2.3).

Inter-axial observation supports a user with general information about a relevant web resource (its logical structure, size with the time estimation based on the selected axis, available navigations, etc.) and information about a user's location based on the neighboring axes. The following is information available for a user:

- set of all global axes created for a certain web resource;
- set of all types of global axes created for a certain web resource;
- set of axes within the neighborhood of the current object, and its special cases such as:
 - set of axes intersecting the current axis item $A(i)$: $\pi_1(\chi(A, i))$. (A neighborhood here is defined as the equality to the axis item $A(i)$.) For example, for the second item of the axis “top news titles 1” in Figures 6.4 and 6.6, it is axis “top news 1” and the basis axis;
 - set of axes within the spatial neighborhood of the axis item $A(i)$ with the radius ε_{sp} : $\pi_1(\nu_{sp}^{ax}(A, i, \varepsilon_{sp}))$. For example, (considering only axes going through basic objects in Figure 6.3) for the second object of the news axis this is represented by news title axis;
 - set of axes within the semantic neighborhood of the axis item $A(i)$ with the threshold ε_{sem} : $\pi_1(\nu_{sem}^{ax}(A, i, \varepsilon_{sem}))$. For example (considering only axes going through basic objects in Figure 6.3) for the first item of the comment axis which contains three objects this is represented by news title axis and news axis;
- set of axes within the neighborhood of the current axis, and its particular cases:
 - set of axes X within the semantic neighborhood of the current axis $A \notin X$ with a threshold ε_{sem} : $\sigma_{\pi_1 \neq A} \pi_2(\nu_{sem}(T, \varepsilon_{sem}) \bowtie \pi_{typeOfAxis,1}(\mathcal{A}))$ or $\{B | B \in \mathcal{A} \wedge B \neq A \wedge type(B) \in \nu_{sem}(T, \varepsilon_{sem})\}$;
 - set of axes of the same type as the current axis A : $\{B | B \in \mathcal{A} \wedge B \neq A \wedge type(B) = type(A)\}$. (Neighborhood here is defined as an equality to the axis type of A .) For example, for the axis “navigation menu 1” in Figures 6.4, it is axis “navigation menu 2”;
 - set of axes which share the same feature or set of features, for instance, with the same global direction or similar length, etc.;
- set of axes types related to each other by a certain feature or relationship, and its special case:

- set of axis types within the semantic neighborhood of a type of the current axis A with a threshold $\varepsilon_{sem}: \nu_{sem}(type(A), \varepsilon_{sem})$.

6.4.2 Locomotion

Locomotion is a process of navigation in some environment according to a particular goal and some level of orientation. As in the case of observation, there are two types of locomotion in the MANM: *in-axial* and *inter-axial*.

In-axial locomotion is designed for navigation within one axis. It includes the following operations: go to

- next item (go to the next object of current axis);
- previous item (go to the predecessor of the current object on the current axis);
- first item;
- last item;
- item with specified index (sequential number).

Before presenting operations utilized for the inter-axial locomotion, we give definitions to the following concepts. We define the term **setting new axis** as a movement from the current element to the first element of another axis. The term **changing the axis within the neighborhood** is defined as a locomotion from the current element to its *nearest* element of another axis. The term *nearest* in this thesis is associated with geometric space and particularly with a spatial closeness defined in Section 6.2.3; in general, this term can be related to other spaces, for instance, feature spaces.

We single out the following *abstract inter-axial* movements:

1. Locomotion from the i -th element of axis A to axis B , which splits on:
 - a) Setting a new axis: it is a transition to $B(1)$.
 - b) Changing the axis within the neighborhood (see Figure 6.10): it is a transition to j -th element of the axis $B \neq A$ which has at least one element contained in the neighborhood of $A(i)$ specified by some closeness function ε at that, and $B(j)$ with minimum j is also not farther to $A(i)$ than any other items of axis B . In other words, it is a transition to $j = \mathbf{Q}^*(\mathbf{A}, \mathbf{i}, \mathbf{B}, \mathbf{I}) = \inf(\sigma_{d(A(i), B(\pi_1))=M}(I))$, where $I = \pi_2(\sigma_{\pi_1=B}(\nu^{ax}(A, i, \varepsilon)))$ and $M = G_{\min(d(A(i), B(\pi_1)))}(I)$.
2. Locomotion from the i -th element of axis A to another axis from the neighborhood $\varepsilon(A)$ of A , where $\varepsilon(A)$, which differs by:
 - a) Setting a new axis (see Figure 6.11): it is a transition to the 1-th element of some axis $B \in \varepsilon(A)$, where the element $B(1)$ is not farther to $A(1)$ than the first item of any other axis from the neighborhood. Formally speaking, $B \in \mathbf{Q}^{**}(\mathbf{A}, \mathbf{i}, \varepsilon) = \sigma_{d(A(1), \pi_1(1))=M}(J)$, where $J = \varepsilon(A)$ and $M = G_{\min(d(A(1), \pi_1(1)))}(J)$. If there

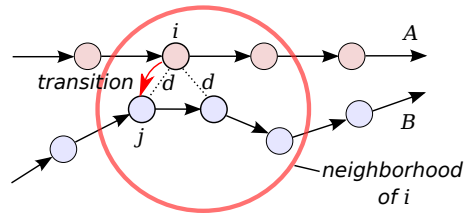


Figure 6.10: Operation of changing axis to the specific axis within the neighborhood of the current item—Inter-axial locomotion from element $A(i)$ with *changing* the axis A to axis B within the neighborhood of element $A(i)$, where $B(j)$ is chosen as a target element

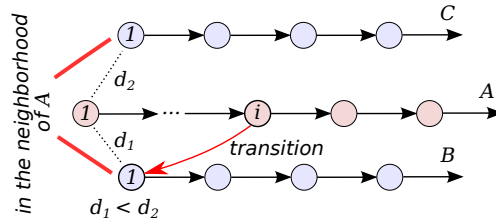


Figure 6.11: Operation of setting a new axis within the neighborhood—Inter-axial locomotion from the axis A to another axis within the neighborhood of A , where $B(1)$ is chosen as a target element

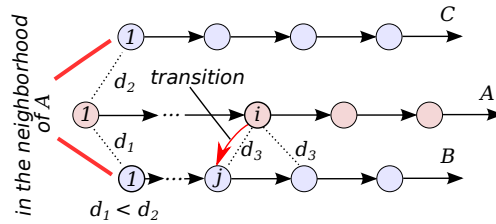


Figure 6.12: Operation of changing axis within the neighborhood—Inter-axial locomotion from the axis A to another axis within the neighborhood of A , where $B(j)$ is chosen as a target element

are several candidate axes X , the one with minimum $pos^1(Y(1), Z)$, where $Y \in X$, is chosen.

- b) Changing the axis within the neighborhood (see Figure 6.12): it is a transition to j -th element of some axis $B \in \varepsilon(A)$, where the element $B(1)$ is not farther to $A(1)$ than the first item of any other axis from the neighborhood; $B(j)$ has the minimum j and is not farther to $A(i)$ than any other items of axis B . To define formally, $j = Q^*(A, i, B, I)$, where $I = \|B\|$, and $B \in Q^{**}(A, i, \varepsilon)$. If there are several candidate axes $X \in \varepsilon(A)$, the one with minimum $pos^1(Y(1), Z)$, where $Y \in X$, is chosen.

Inter-axial locomotion corresponds to the main movements of a blind user through the axes and comprises the following operations:

- corresponding to the first group of abstract inter-axial locomotions:

- set/change axis on the intersection, for *changing* the axis
 $j = \inf(\pi_2(\sigma_{\pi_1=B}(\chi(A, i))))$.
 (This requires consideration of the set of axes intersecting the current axis item);
- set/change axis in the spatial neighborhood of an object, for *changing* the axis
 $\nu^{ax}(A, i, \varepsilon) = \nu_{sp}^{ax}(A, i, \varepsilon_{sp})$.
 (This requires consideration of the set of axes within the spatial neighborhood of the current axis item);
- set/change axis in the semantic neighborhood of an object, for *changing* the axis
 $\nu^{ax}(A, i, \varepsilon) = \nu_{sem}^{ax}(A, i, \varepsilon_{sem})$.
 (This requires consideration of the set of axes within the semantic neighborhood of the current axis item);
- corresponding to the second group of abstract inter-axial locomotions:
 - set/change axis in the semantic neighborhood of axis type,
 $\varepsilon(A) = instances(\nu_{sem}(type(A), \varepsilon_{sem}))$.

It is important to note that a blind user can move to any axis she can obtain via observation operations presented in Section 6.4.1.

The inter-axial navigation (possibility to change axis at the particular object in its spatial or semantic neighborhood) is an important new concept and one of the benefits of our navigation model. It includes both inter-axial observation and locomotion. The reason for changing an axis can be the result of finding an item with stronger information scent and willingness to change the axis from a more generic to a more specific one. For example, changing an axis which goes through top news titles to the axis with top news at some relevant object (see Figure 6.4), or from axis of news categories to the news list axis (see Figure 6.5). In contrast, locomotion from a more specific axis to a more generic one could be caused by very low information scent of objects of a current axis. A user's transitions through axes in either spatial or semantic neighborhoods could mean that a blind user tries to find serialization which better suits her goals and has stronger information scent according to the current position.

The inter-axial navigation allows the user to explore the structure of a document quickly with different contexts and goals and without losing orientation. Since the axis model is a stable network the user can go astray and backtrack to a more familiar location (landmark) at any point. This principle of “no surprises” encourages users to explore freely and without fear of getting lost, while the coarse grain nature of some of the axes also makes it a less time-consuming and therefore less painful experience.

Thus, observation and locomotion are interconnected and provide a basis for scanning and becoming familiar with a web page while ensuring a user's mobility. For example, to read news headlines of the environment category on the main CNN web page by means of the MANM illustrated in Figures 6.4, 6.5, and 6.6, the user can perform the following operations (without the use of the keyword search):

- obtain all types of global axes on a web page,
- choose the navigation axis “categories 1” and read it through,

- choose the navigation axis “categories 2” and read it through until the required category is found,
- read the set of axes within the spatial neighborhood of the current axis item,
- choose the navigation axis “news list 17” and read it through.

It is worth mentioning that instead of switching from “categories 2” to “news list 17,” a blind user can also choose the basis axis and read relevant news headlines. We have demonstrated an example of navigation by means of the MANM without the use of the keyword search to show its effectiveness and general principle of navigation as well as to reflect the reasoning involved.

In contrast, a user’s navigation by means of common screen readers is less intuitive and mainly based on the use of heading tags (if they are provided), links, and linear navigation through DOM tree elements.

6.5 Blindzilla: Implementation

For the approbation of the concepts and methods introduced in this chapter, particularly the MANM (see Section 6.2) and the methodology of navigation (see Section 6.4), and to prove their feasibility and effectiveness, we developed Blindzilla, a set of prototypical tools based on the WPPS framework. All components of Blindzilla are cross-platform and mainly written in Java (JDK 1.7.0); they are plug-ins of the Eclipse RCP (Indigo) framework. The Blindzilla prototypes were successfully tested on different operating systems, such as Ubuntu, Mac OS X, Windows XP, and Windows 7.

6.5.1 System Architecture

We present two main Blindzilla implementations: Blindzilla1 and Blindzilla2. Both of them realize the MANM and the navigation methodology conveying an accessible interface due to the use of SWT/JFace libraries. The latter utilizes components of the local operating system for graphical user interfaces that makes them accessible by common screen reading technologies. Blindzilla1 and Blindzilla2 have a Firefox 3.6 (XULRunner of the version 1.9.2) integrated which enables web surfing and interaction with web pages. For interaction with the blind user, the prototypes have speech synthesizers integrated (e.g., FreeTTS, eSpeak, Festival). The general architecture of the Blindzilla prototypes is illustrated in Figure 6.13.

Blindzilla1 operates with the earlier versions of the MANM and was used for the investigation of the navigation model and navigation as well as for their successive amelioration. The improvements were performed according to the recommendations acquired from blind users during the ABBA [236] and the TAMCROW [237] projects. Furthermore, it is based on the earlier version of WPPS (version 1) and integrates different algorithms and approaches developed by a team of the ABBA project [236] for web page analysis, in particular, algorithms for generating different versions of the Physical Model (PM) and the Gestalt Ontology (see Section 4.2). Blindzilla1 comprises three general plug-ins: `at.dbai.blindzilla.builder` (a MANM builder from the PM or the Gestalt Ontology), `at.dbai.blindzilla.navigator` (a navigator through the

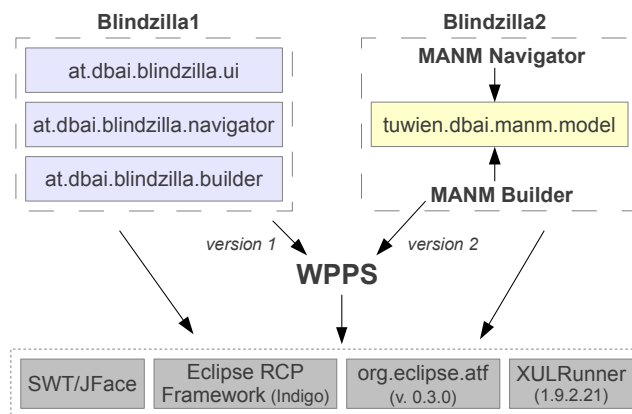


Figure 6.13: Main components of Blindzilla

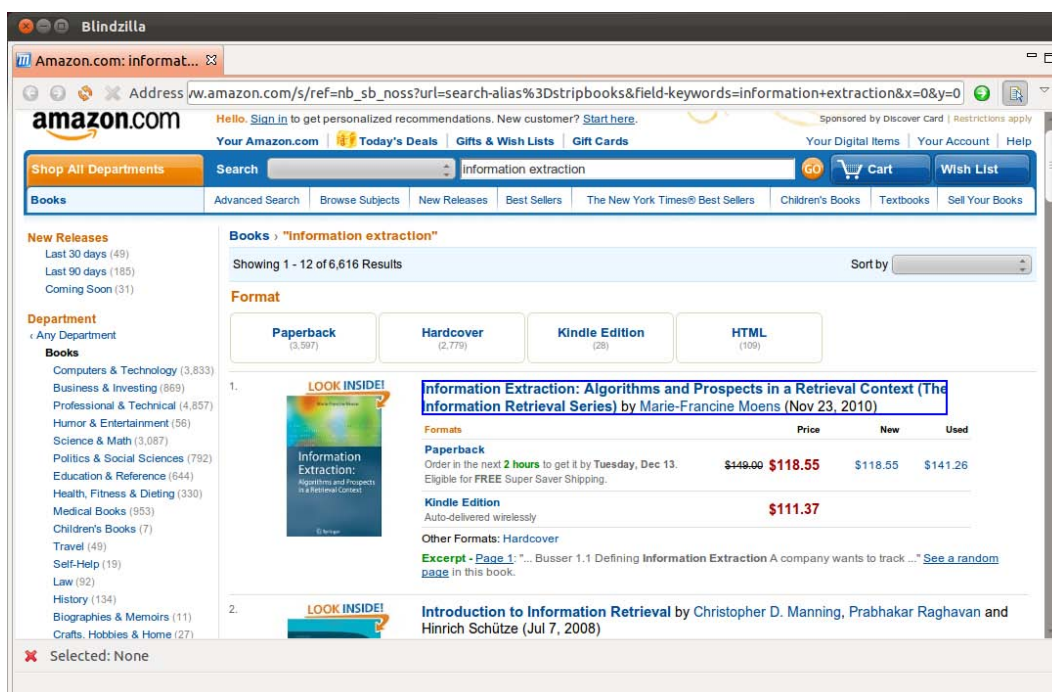


Figure 6.14: Graphical user interface of Blindzilla1

instantiated MANM), and `at.dbai.blindzilla.ui` (an SWT/JFace graphical user interface, based on the Eclipse RCP platform). As demonstrated in Figure 6.14, it has relatively simple interface which is fully accessible with keyboard.

Blindzilla2 realizes the latest version of the MANM and navigation methodology introduced in this thesis and was used in the evaluation presented in Section 6.6. It has full integration with the latest version of the WPPS framework (version 2) and is represented by two prototypes: MANM

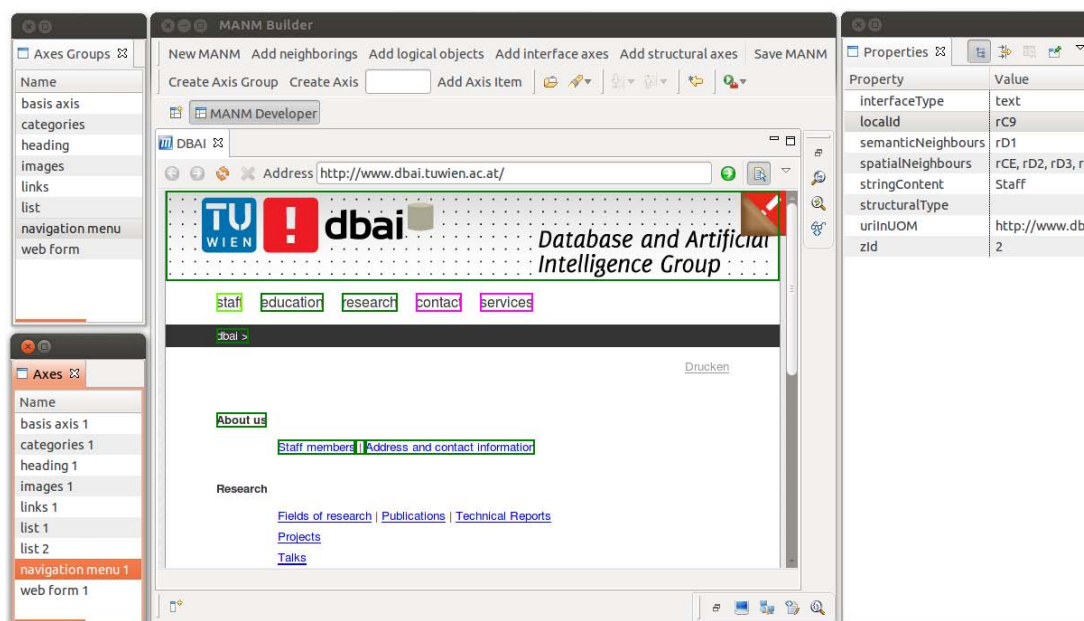


Figure 6.15: Graphical user interface of MANM Builder

Builder (for building the MANM of a web page, see screen shot in Figure 6.15) and MANM Navigator (for navigating through the instance of the MANM). Both prototypes are dependent on the `at.dbai.manm.model` library conveying classes and application programming interface (API) necessary for interacting with the MANM. Like `Blindzilla1`, a MANM Builder is Eclipse RCP based application, integrating `XULRunner` and leveraging ATF plug-ins. In contrast, MANM Navigator is Java tool only based on `at.dbai.manm.model` with accessible SWT/JFace graphical user interface. This interface listens to the keystrokes and enables opening serialized MANMs.

6.5.2 Model Generation Component

In the `Blindzilla1` prototype, the MANM is instantiated by the `at.dbai.blindzilla.builder` plug-in. This process is automatically executed as soon as the web page, requested by the blind user through the integrated web browser, is loaded and rendered. `Blindzilla1` generates basis axis mainly relying on the DOM tree, serializing web page elements according to the depth-first traversal. This specialized web browser generates various axes, including the HTML-based (e.g., headings, links, images, web form elements) and those serializing different types of recognized logical objects (e.g., lists of visually ordered elements, navigation menu, sections).

`Blindzilla2` has a dedicated tool called the MANM Builder for building an instance of the MANM for a given web page (see Figure 6.15). This application allows the developer to instantiate the model for the rendered web page according to the method presented in Section 6.3. With the graphical user interface (GUI) of MANM Builder, the developer can annotate a web page manually and create axes. WPPS API is extensively used in MANM Builder. The basis axis Z

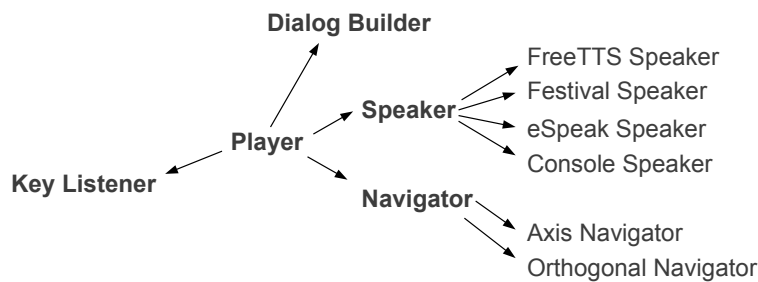


Figure 6.16: Main components of the navigator

with readable elements (i.e. web form elements, alternative text of images, and visible textual content) is generated based on the serialization of leafs of a segmentation tree according to the depth-first traversal. The tree is acquired by the application of the adapted XY-cut algorithm over the web page canvas. By means of an integrated core of the WPPS framework, MANM Builder also enables automatic enrichment of the MANM with spatial neighbors and identification of different logical objects. Information regarding the spatial neighbors of each element of the axis Z as well as recognized interface and logical (structural) objects are materialized in the MANM instance. Extraction of logical objects on a web page is performed by means of invocation of different wrappers leveraging WPPS API. Also, recognized interface and structural (logical) objects are ordered by means of automatically created interface and structural axes respectively.

6.5.3 Navigation Component

The main aspects of the navigation methodology introduced in Section 6.4 are implemented in `at.dbai.blindzilla.navigator` plug-in of `Blindzilla1` and in `MANM Navigator` of `Blindzilla2`. Available keystrokes are presented in Appendix ???. In contrast to `Blindzilla1`, `MANM Navigator` does not generate the MANM “on-the-fly” and does not have a web browser integrated. It is purely a navigation system which operates with the MANM built by the `MANM Builder`. The `MANM Navigator` is an enhanced version of `at.dbai.blindzilla.navigator` with optimized dialogs and algorithms.

In the implementations proposed, we distinguish five main objects (see Figure 6.16): `Navigator`, `Dialog Builder`, `Speaker`, `Key Listener`, and `Player`.

There are two types of **navigators** in `Blindzilla`: `axis navigator` and `orthogonal navigator`. An `axis navigator` conveys functions necessary for in-axial observation and locomotion, for example, to get axis length, quantity of passed items, go to the next item, etc. `Orthogonal navigator` walks over a labeled planar graph with arcs connecting each readable elements on a web page canvas with the four nearest orthogonally visible elements (to the north, east, south, and west). **Dialog Builder** provides all the necessary dialogs and textual representations of web page elements. **Speaker** is responsible for the interaction between computer and the user. It provides the user with necessary audio or textual information according to her interaction leveraging the `Dialog Builder`. `Blindzilla` realizes four types of speakers: `FreeTTS Speaker`, `Festival Speaker`, `eSpeak Speaker`, and `Console Speaker`. The first three speakers generate audio output utilizing different

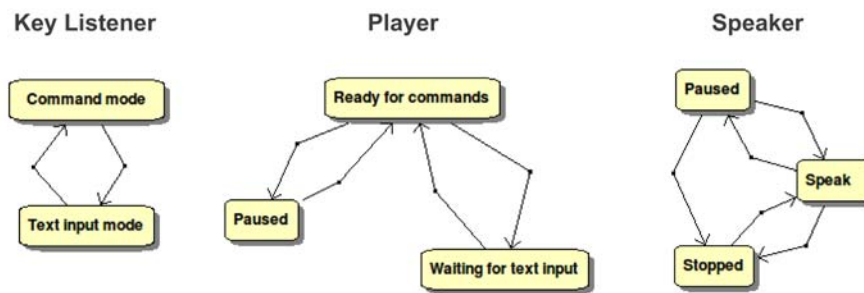


Figure 6.17: The main states of the navigation components

speech synthesizers, whereas the Console Speaker outputs textual information. The FreeTTS Speaker leverages FreeTTS¹ system to render speech and is written entirely in Java and supports only English. Interaction with the system is conducted via Java Speech API, providing a unified interface for text-to-speech engines that implement it. Thus, FreeTTS Speaker has complete integration with FreeTTS and is cross-platform. Festival Speaker using Festival² and eSpeak Speaker operating with eSpeak³ have relatively limited integration with the corresponding text-to-speech engines. This is due to the fact that the utilized engines are written in C++ and C respectively and therefore are invoked outside the Java virtual machine. It is worth mentioning that Festival and eSpeak are cross-platform and support multiple languages including English, German, and Russian. Console Speaker outputs relevant information into the application console that allows the blind user to use their customary screen reader for synthesizing speech. **Key Listener** is a “mediator” between the user and Player; it intercepts key press events and translates them into the commands for Player. **Player** is a central element in the navigation system. It is responsible for realizing the navigation methodology with the help of navigators, for generating textual dialog and textual serialization of the elements to be read by means of Dialog Builder, and for interacting with the user leveraging a certain speaker.

Objects such as Key Listener, Player, and Speaker have states which are worth mentioning. Figure 6.17 depicts the main states of the objects and possible transitions. Key Listener has two main states: “command mode” and “text input mode.” In the command mode, Key Listener interprets key presses into commands passed to Player. In text input mode, Key Listener does not interact with Player and is used to allow the user to interact with a web page, for example, to type text into the editable elements. The transition between these states is performed by the user request. Player in turn has three main states: “ready for commands,” “paused,” and “waiting for text input.” In the first state, Player reacts on user commands and carries out corresponding operations (e.g. reads next element on a certain axis, gives an overview of the neighboring axes within the spatial neighborhood). In the pause mode, Player holds its current state and does not perform any operations. For instance, if the player was playing through the items of a

¹<http://freetts.sourceforge.net/>

²<http://www.cstr.ed.ac.uk/projects/festival/>

³<http://espeak.sourceforge.net/>

certain axis, then when switching into the pause mode by the user, it will discontinue to read any further, and will resume its activity after the transition from this state to the “ready for commands” state. Transition between the first two modes is directly controlled by the user. The third state is activated when the player requires additional parameters from the user, for example, to specify the number of the axis which the user switches to. Speaker has three main states: “speak,” “paused,” and “stopped.” In the speak state, Speaker is synthesizing speech according to the text provided and outputting into the relevant device (i.e. speakers or console). Transitions between these states are controlled by Player. Pause mode interrupts the current audio stream and resumes it when the speaker switches into the speak state. This procedure (i.e. pause-resume) only functions properly for FreeTTS Speaker due to its complete integration with FreeTTS speech synthesizer. The speaker goes into the stop state if either the audio output is complete or if it was forcibly stopped by Player. The latter can happen when the user activates command when audio output is still going, and as a consequence, Player interrupts Speaker.

Figure 6.18 demonstrates an example interaction scenario. For the sake of simplicity, Dialog Builder is not shown on the diagram. However, it is utilized every time Player sends text to Speaker. In the example scenario, Key Listener sends three commands to Player by the user request: set new axis, play axis items, and go up. Reacting on the commands received, Player interacts with Speaker and with one of the navigators available.

6.6 Blindzilla: Evaluation

To evaluate the effectiveness of the proposed MANM (see Section 6.2) and navigation methodology (see Section 6.4), we conducted a user study with five participants who are blind, comparing contemporary screen readers with Blindzilla2 (see Section 6.5) [74]. Each of the volunteers had a certain amount of experience working with screen readers such as JAWS [90], Window-Eyes [178], and Fire Vox [47] at the time of the experiment. These tools have different levels of popularity among the blind community [296]. Presented in Section B.2, JAWS is the most commonly used software according to different inquiries, for example, [296]. Although Window-Eyes provides similar functionality, it is used less often. In general, the cross-platform Fire Vox is used quite seldom and is most popular among Linux users. For the purpose of the evaluation, the volunteers also learnt the API of MANM Navigator and where able to use it.

We defined a set of navigation scenarios that we believe capture a realistic range of typical usage behavior within varying levels of complexities. As such, web pages of different genres such as weblog, on-line shops, news, and search engines were considered. Overall, we defined six tasks⁴ and presented them to the testing group. These tasks are as follows: For a given page⁵

- Dog blog: Find and read the latest blog post;
- Cat blog: Find and read the second newest blog post;
- Amazon 1: Find the cheapest book listed on the web page;

⁴The tasks given to the blind users were formulated together with the participants of the ABBA project [236].

⁵<http://www.dbai.tuwien.ac.at/staff/fayzrakh/blindzilla/ds/blindzillaEval2012.zip>.

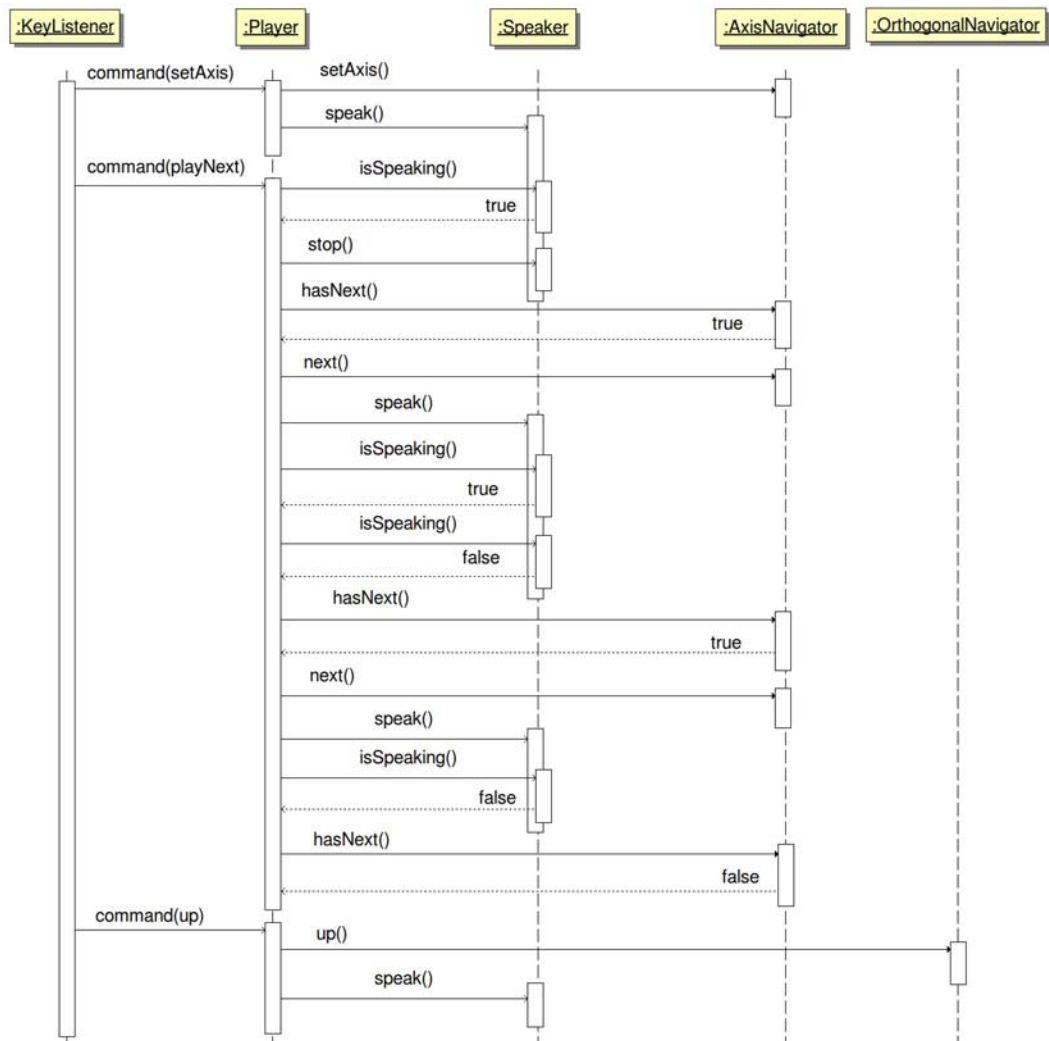


Figure 6.18: Sequence diagram of the navigation component of Blindzilla demonstrating an interaction example

- Amazon 2: Find and read the first customer review;
- CNN: Find and read the news teasers in the “business” category;
- Google: Read the titles of web resources on the result page.

Dog blog and cat blog are web pages of a weblog genre, rich with images which are not compliant with the accessibility standards. Amazon 1 and 2 are web pages of the popular on-line shop, CNN is a popular news website, and all of them have relatively complex interfaces with the active use of AJAX technology. Google is a good example of the search engine.

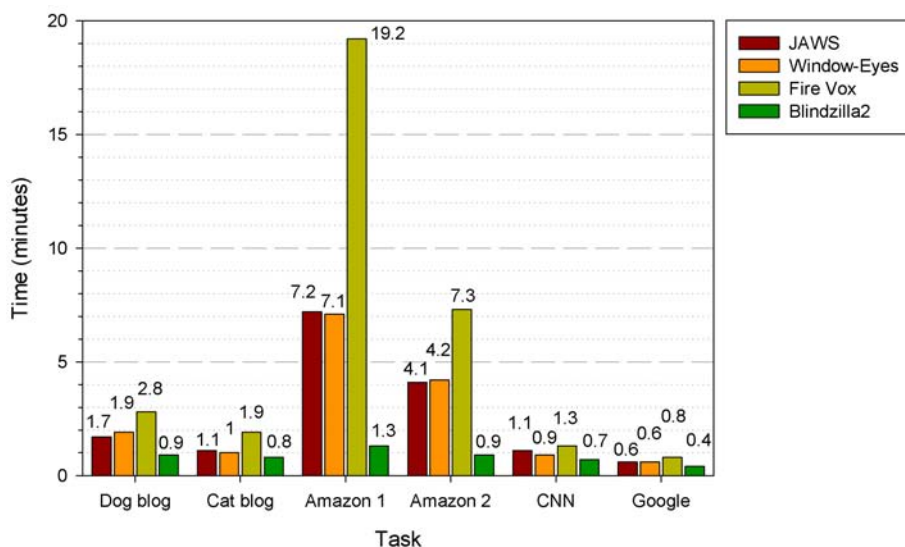


Figure 6.19: Results of the evaluation of the efficiency of using the proposed Multi-Axial Navigation Model and navigation methodology relative to contemporary screen readers

For each web page considered, the main HTML-based axes available in today’s screen readers (e.g., axes of headers, links, images) were automatically generated in the MANM Builder, as well as certain axes serializing various types of logical objects recognized on a web page (such as titles, categories, sections, dates, prices, etc.). Some of the logical objects were manually annotated on a web page, for example, list of products on Amazon 1 page or sections on Amazon 2. A generated basis axis Z reflects the layout of a web page leveraging the adapted XY-cut algorithm for the content segmentation.

Before the experiment, each participant was asked to choose one of the proposed screen readers they were familiar with and would use during the entire study. Since the participants were not familiar with the user interface of the MANM Navigator, we suggested to two participants that they should use it for the experiment. At the end, their average results were estimated. The testing group was also asked to avoid using keyword searches. All the screen reading tools had the voice rate equal to 200 words per minute. We chose time spent for fulfilling a task as the basis for evaluating the efficiency of the navigation model and the methodology proposed. The results of the experiment are summarized in Figure 6.19.

Depending on the task, the quantitative results show that the various screen reading systems have different efficiency. In particular, the participants encountered severe obstacles on the Amazon 1 task, which took them on average from 7 to 19 minutes to complete with the use of third-party tools. The reason for this poor performance is that the Amazon 1 web page contains a lot of elements of different types, and it was difficult for the blind users to distinguish products with the price located in different parts of the web page content from the rest. Furthermore, all the web pages except Google are not compliant with the accessibility guidelines. For example, they do not provide alternative text for images, they use table tags for molding the web page layout,

and they misuse HTML paragraphs and HTML headings. Blind users are quite sensitive to these issues due to the fact that most of them prefer to have images properly described [29] and the structure of a web page well organized with HTML headings [292–294]. Thus, the participants used different navigation strategies [37] (except keyword search). However, the most common were heading, list, table and paragraph navigation as well as sequential navigation with reading the text line by line. Due to the misuse of HTML tags the participants had to use sequential navigation reading through the page quite often, which explains why some tasks (especially Amazon 1 and Amazon 2) took them so much time to complete. The Blindzilla2 prototype fared better due to the possibility of navigating through the logical elements of a web page, in particular, the web and domain specific. This important advantage enables the user to think in terms of logical representation and logical objects instead of the various serializations of a web page’s source code and HTML elements.

In general, the Blindzilla2 prototype yielded the best performance across all tasks. On average, tasks with the use of the Blindzilla2 prototype were carried out 3.61 times faster than with the use of other screen readers, and 2.74 times faster than JAWS 13.0. The average value is computed according to the following formula:

$$\frac{\sum_{i=1}^N \sum_{j \in S} t_{i,j} / (\|S\| \cdot t_{i,\text{blindzilla}})}{N},$$

where $N = 6$ is the number of tasks, S is a set of the screen readers compared against Blindzilla2, $t_{i,j}$ is a time spend for the task i using the screen reader j .

In the interview, the participants shared their opinion regarding the navigation system and the Blindzilla2 API provided. The models built by the Blindzilla2 corresponded to the domain of the web page, and this factor was crucial in the success of the users of Blindzilla2. Furthermore, some of the axes went through logical objects. Using the keystrokes, the participants were able to acquire a list of available structural (logical) axes and thus, obtain an overview of a web page and choose an axis most relevant to the current task. We would like to note that for the inter-axis navigation, users mainly used the operation of *changing axis on the intersection*. This operation was leveraged, for example, when the users navigated through more general axis (or so-called “faster” axis, for example, which goes through titles of blog posts) and after finding the required element or element with stronger information scent, they changed it to the more detailed (or so-called “slower” axis, for example, which goes through all elements of the blog content). While carrying out the CNN task, for instance, one of the users, after choosing axis of news categories and moving to the required category, changed it to the basis axis for reading the relevant news titles of the current category. Thus, the MANM together with the navigation methodology allows the user to navigate the content more adequately within their goals and web page domain. We believe this factor is crucial for ensuring user mobility [97].

6.7 Discussion

In this chapter, we presented the Multi-Axial Navigation Model (MANM) (see Section 6.2), navigation methodology (see Section 6.4) and Blindzilla prototypes (see Section 6.5) implementing them. The effectiveness of the proposed models and methods was demonstrated in the

evaluation conducted with a small group of blind users utilizing examples of several web pages with predefined axes (see Section 6.6). Blindzilla proved its efficiency in contrast to the common screen readers and yielded the best performance across all tasks. On average, tasks with the use of Blindzilla were carried out more than 3 times faster than with the use of other screen readers and this confirms the importance of further research necessary in this direction.

Our main goal is to maximize the benefits from the one-dimensional navigation within the multidimensional information space of a web page. Blind users lack visual cues which enable sighted users to almost instantly build a relevant mental model of a web page and thus, easily find and read interesting information. Blind users always lack the context and more generic picture of a web page content. Therefore, MANM, which can be built both automatically and by means of manual annotations, provides blind users with source code independent serializations and navigation mechanisms for getting overview, context, and various trails for effective and efficient navigation.

The MANM and navigation methodology improve a blind user's mobility in the following aspects introduced in [97, 112] (see Section 6.1.2):

Mobility objects:

(Cues, obstacles, memories) The MANM incorporates different types of objects adopted from the Unified Ontological Model (UOM) and thus, reduces the number of "obstacles" and increases the number of "cues". The presence of various types of objects describing interface and logical components of a web page helps to identify various landmarks which support orientation within the web page environment.

(Out-of-view) The presence of various navigation axes covering different parts of a web page content ensure more efficient access to different navigable objects and therefore enables a decrease in the amount of "out-of-view" objects.

Mobility techniques:

(Preview and Probing) This aspect is accessible by various navigation methods which a blind user can perform including observation and locomotion. For example, to obtain objects and axes within semantic and spatial neighborhoods, the user can utilize in-axial and inter-axial navigations as well as orthogonal two-dimensional navigation (based on orthogonal visibility of objects, see Figure 4.16 on page 120 and Figure 4.17).

(External Memory) A variety of different ways of navigation over an instance of the MANM as well as regularity and static character of the network of axes enable a blind user to remember and easily recognize her whereabouts and therefore be able to plan further navigation without probing.

The proposed concepts also ensure effective application of the mobility principles. Regarding the *information flow*, for example, different axes with different levels of refinement of the web page content (e.g., axes connecting news groups, news headlines, or news content) provide a blind user with different speed and information details in reading a relevant web page. This provides the possibility to choose a certain axis detailed enough to understand the current context. *Granularity* is related to the regularity and static character of the network of axes (principle of *regularity*) which enable a blind user to easily find familiar objects and whereabouts. This also provides the

opportunity to build a mental model relevant to a web page (principle of *memory*). Furthermore, *egocentricity* and *spatial principles* are also taken into account providing sightless users with a deictic frame of reference [50,205] which gives her the possibility to orient herself relative to her current position, which is very natural in real word navigation.

Conclusion and Future Work

Accessibility of web resources for different groups of people in society is a very important aim which has been addressed within the field of Web Accessibility. In this thesis however, we specifically focus on the challenges of sightless people who encounter numerous problems while interacting with web pages due to a lack of the most important constituent of a web page—visual presentation.

When investigating contemporary approaches, methods, and tools ameliorating accessibility of web pages for blind users, we realized the inalienability of Web Accessibility within the research fields of Web Page Understanding (WPU) and Web Information Extraction (WIE), which are both referred to as Web Page Processing (WPP). Therefore, the aim and objectives posed within this thesis are directly related to the aforementioned areas of research (see Section 1.3).

7.1 Results

We divide the results obtained in this thesis into those making a direct contribution to WPP and those enhancing accessibility of a web page.

7.1.1 Web Page Processing

A. The *extensive analysis*, conducted in Section 2.4 of various models and structures for document representation used in WPP, indicated the importance of considering the rendered state of a web page for developing effective and robust web page processing (WPP) methods. In particular, the consideration of qualitative visual features on a web page canvas provides the developer with broader opportunities of applying methods of different fields of study, be it Document Understanding (DU), spatial reasoning, spatial cognition, Computer Graphics, or Computer Vision. Furthermore, in contrast to various ways of spelling the visual representation into X/HTML, the presence of a greatly smaller amount of different spatial configurations of web objects and visual design patterns ensures the robustness of methods operating on the rendered web page (see Sections 2.4.8 and 5.4.2). Consequently, the effectiveness is ensured by the analysis

of semantically relevant features of a web page which are considered by sighted users when reading a web page. It is also important to note that the semantic gap between source code and visual representation is growing even larger over time due to the constant development of new technologies. Unfortunately, methods leveraging visual cues are less efficient due to the time required for rendering the DOM tree together with CSS Object Model (CSSOM) as well as generation of the required web page model. Thus, the developer has to find a trade-off between robustness and effectiveness on the one hand and efficiency on the other hand.

Therefore, this analysis provides us with necessary knowledge regarding the existing web page representations and relevant WPP methods as well as conclusions which underlie the models and WPP approaches proposed within this thesis.

B. Based on the *extensive analysis* of web page representations, we investigated and identified various spatial relations and features of elements populating pages' canvases (see Chapter 3) and gave them qualitative and quantitative expressions. The most commonly used qualitative spatial relations were specified and expressed by means of ***Two-Dimensional Interval Relations with Centering (2DIRC)*** (see Section 3.10.1). This ensures the elimination of invalid combinations of basic relations between elements within the relevant *qualitative model* and dramatical decrease of various combinations of pairwise joint basic relations (by the factor of 32). Due to the peculiarities of the rectangular shape of basic web page elements, the ***RCC8 topology was refined*** with 33 supplementary relations (see Section 3.6.2). These 33 specific relations together with NTPP, NTPP⁻, and EQUAL are jointly exhaustive and pairwise disjoint (JEPD) and thus, enable the description of all possible spatial configurations of blocks merely based on the connection relation C. Another important innovation is a ***fuzziness in the interval relations*** which is able to deal with certain peculiarities and inaccuracy in the arrangement of elements on a web page canvas (see Section 3.10.2). Furthermore, assorted identified spatial relations were evaluated over the ***WPPS-HTML-DS1 dataset*** [72] presented in this thesis (see Section 3.11). Various statistical characteristics acquired during the *evaluation* reflect different features of web pages, such as regularities in the layout and the presence of Manhattan or near-Manhattan layouts and therefore the predominance of orthogonal directions. In addition, there is the presence of more than 15% of elements which are not rendered on the relevant page's canvas and should be eliminated from the visual-based WPP methods. [73, 81]

All the concepts mentioned are used for modeling a web page structure and layout by means of both quantitative and qualitative characteristics reflecting correspondence between them.

C. We proposed a ***web page conceptualization*** which represents various aspects of a web page in the form of layers considered in different processes and tasks (see Figure 4.3 on page 106). To benefit from visual (i.e., layout, interface objects, font and other visual features) and technical (i.e., the source code, CSSOM, and DOM tree) layers of a web page, we have elaborated the ***Unified Ontological Model (UOM)*** [77, 80, 82, 151] (see Chapters 3 and 4). The UOM is based on the models used within the fields of WPP and DU and consists of two main sub-models: the Physical Model (PM) and Logical Model (LM). The former also consists of sub-models which provide relevant constructs for modeling technical, geometric and interface layers of web pages. Specifically, the ***Block-based Geometric Model (BGM)*** [73, 81] (see Section 4.4.2), sub-model

of the PM, representing the geometric layer models various spatial relations between *structural web page elements* introduced in Chapter 3. By means of domain ontologies, the LM can be used to interpret knowledge acquired during WPP. In particular, it allows developers to model different data structures of logical objects as well as web specific and domain specific objects. In its practical application within the scope of the ABBA [236] and TAMCROW [237] projects, the UOM demonstrated its sufficient expressiveness in describing different aspects of a web page used in various tasks of web page analysis.

D. We also introduced the following concepts in this thesis:

- The WPP was defined as a *process of gradual generation of different models of web page representation*: first, the instantiation of the PM according to the DOM trees and CSSOMs; second, instantiation of the UOM as a process of WPU and WIE; and third, its transformation into other representations required for the integration into external systems [71, 76, 77] (see Section 5.1).
- There are declarative (e.g., the Datalog \pm , H μ L ε X, and simplified web pattern matching languages) and imperative approaches (e.g., languages such as Java, C#, C++, and Perl) used in the implementation of WPP methods. Each of these approaches has its advantages and disadvantages. Thus, to benefit from both approaches we proposed a *bridged adapter* software design pattern [76, 77] (see Section 5.2.4) for implementing object-oriented abstraction for ontological modes. It also enables the development of the software which can automatically adapt its implementation according to certain modifications within the subject model and modes of computing the required information. All operations performed on the object-oriented level are transmitted into the ontology and all the modifications on the ontology level are accessible for the object-oriented abstraction.

Leveraging these concepts and approaches, we developed a *Web Page Processing System (WPPS)* [75–77] (see Section 5.3) which demonstrated the feasibility and effectiveness of the UOM proposed as well as its relevant concepts and methods. In particular, WPPS performs the analysis of a web page according to the proposed workflow for the WPP. Furthermore, by utilizing the bridged adapter design pattern, it supports various *configurations* of the UOM which specify types of objects, attributes, and relations to be automatically instantiated in the PM, modes of obtaining the requested information (e.g., querying the ontology or computing “on-the-fly” based on other information presented in the ontology), inference rules to be applied, consideration of the fuzziness, fragment of a web page which should be considered for the analysis, the use of RDF containers (when it is required to take into account the order of objects), and much more (see Section 5.3.2). The WPPS demonstrated its effectiveness and efficiency in developing wrappers (see Section 5.4) and was also used as a framework for developing other systems [145] (e.g., *ObjIdent*, a web page annotation tool for computing features [84] relevant for the basic web object identification, see Section 5.6). In the detailed *evaluation* of WPPS in Section 5.5, we demonstrated different configurations by example of two queries and discussed efficiency and the use of relevant wrappers, providing the developer with the necessary information for choosing the most appropriate configuration for WPP.

It is important to note that the WPPS leverages an *R-tree* index for querying the containment and intersection relationships. Furthermore, as we found out in practice, during the PM instantiation WPPS omits about 37.7% of web page elements on average, which are invisible or not visually perceptible by the user. This makes interaction with an instance of the UOM more efficient.

7.1.2 Web Accessibility

E. We conducted a *survey* of 95 participants who are blind. As discussed in Appendix B, this survey reflects the present state and relevant problems within the field of Web Accessibility. We believe that this survey could also motivate researchers of web accessibility and web authors who work collaboratively to focus on resolving the problems that have been identified. In particular, this survey has revealed that there is poor accessibility of websites in certain genres and a huge discrepancy between the perception of a web page by sighted and sightless users and their mental models respectively.

F. During the analysis of various typhlot technology tools (see Section 2.1) as well as blind users' interaction with them (see Section 2.2), we identified *two main problems*: 1) most web pages do not follow accessibility standards and lack semantic annotations, which is crucial for the DOM tree based screen reading technologies; 2) navigation is mainly represented by locomotions through DOM tree elements of a certain type. In some senses, this approach forces blind users to grasp the logical structure of a web page and the role of different elements based on the DOM tree serialized by the screen reading tool. Thus, scanning and understanding a web page becomes quite a difficult process for blind users.

The first problem is addressed within this thesis as a web page processing problem, which can be approached by means of relevant tools, methods, and models developed in this thesis (see Section 7.1.1). The second problem is considered from the viewpoint of one-dimensional navigation, caused by the use of aural channel and Braille display, through the multidimensional information space of a web page (see Section 6.1).

G. The *Multi-Axial Navigation Model (MANM)* [21, 83, 85, 151] proposed in this thesis provides blind users with different serializations (referred to as axes) of the web page content, reflecting spatial and semantic relationships on the set of navigable objects and axes (see Section 6.2). The MANM conveys yet another representation of a web page (independent from the DOM tree and visual representation) which can be used for modeling effective navigation trails with annotated logical and interface objects to help blind users to easily and correctly understand the content. The MANM incorporates objects from the UOM provided by methods implemented in the WPPS framework and utilizes orthogonal visibility relations from the BGM for orthogonal navigation. We also presented various characteristics and features of axes which can be used to assess the correctness of the access generated as well as the corresponding generation algorithm.

H. For the MANM proposed, we developed a *navigation methodology* [21, 74, 83, 85] specifying basic interactions which can be performed with the MANM (see Section 6.4). Navigation methods

were defined in accordance to principles presented in Information Foraging theory, cognitive approach to navigation, and principles of user mobility. As such, they ensure efficient and deliberate navigation relevant to the web page's logical structure.

I. The feasibility of the proposed model and methodology is demonstrated by the *Blindzilla* prototypes which were developed based on the WPPS framework. Blindzilla implements a set of WPP methods to identify certain types of objects and build different navigation axes. [74]

Thus, the MANM and navigation methodology proposed are used to obtain maximum performance from interaction with the computer by blind users who are limited due to an absence of visual perception.

J. The proposed model, navigation methodology and tools were evaluated with the help of a small group of blind users. Blindzilla proved its effectiveness in contrast to common screen readers and yielded the best performance across all tasks. On average, tasks with the use of Blindzilla were carried out more than 3 times faster than with the use of other screen readers. [74]

7.2 Future Work

The results acquired in this thesis as well as the concepts and lines of research introduced deserve further studies. In line with the results discussed in Section 7.1, we believe that future work should focus mainly on Web Page Processing and Web Accessibility.

7.2.1 Web Page Processing

Unified Ontological Model of a Web Page

The UOM formalizes certain aspects of web pages reflected in the web pages' conceptual model. This ensures the possibility of applying different methods of fields of research and conducting an exhaustive analysis of different representations. Thus, the UOM should be studied further and other layers of the web page conceptual model should be incorporated. Among the visual layers, the most interesting layer is the Gestalt layer which refers to various Gestalt laws such as law of proximity, similarity, closure, and symmetry. These laws and concepts such as ding, saliency, foreground and background give additional meaning to web page elements. They provide the ability to distinguish more important elements which require a user's attention and to identify different groups of elements and their relation with other groups. The formalization of the Gestalt layer can be a big step towards the development of more robust algorithms reflecting certain mental processes performed by humans during pattern recognition. Thus, visual layers should be further researched as well as their relation to Computer Vision.

We believe it is very important that the logical layers be further investigated and various web specific and domain specific objects be defined in publicly available ontologies. The use of such datasets with Linked Data technology can dramatically improve accessibility of web pages both for humans and computers.

Object-Oriented Abstraction for Ontological Models

The problem of providing object-oriented representation of the application domain for a given ontology is one of the most important challenges considered in model-driven engineering. Unfortunately, contemporary approaches do not satisfy the requirements presented in Section 5.2.2 which are necessary for a seamless integration of declarative and object-oriented paradigms. The solution to this problem, represented by the *bridged adapter* software design pattern, requires the developer to manually define class interfaces, hierarchies, and constraints as well as the application programming interface (API) for interaction with individuals within the ontology. In order to make substantial contributions to model-driven engineering, further research should target the challenge of automatic generation of the object-oriented abstraction and relevant API according to the requirements defined in this thesis. For example, to be adaptive to 1) different inference rules and reasoning mechanisms applied, 2) different sets of concepts instantiated in the ontology, and 3) different modes of obtaining the required information. Moreover, the object-oriented abstraction should be always synchronized with the ontology. This research should also consider problems of instantiating a model which is unknown in advance but is compliant with certain requirements or possess certain properties.

This research study can dramatically increase the efficiency of model-driven engineering and the robustness of developed software.

WPPS

Further enhancement of the WPPS framework should include the optimization of selector and processing functions of the WPPS framework and introduction of more class methods which will make interaction with the UOM and development of WPP methods even more efficient. It is important to integrate other ontology frameworks to have a possibility to choose the most efficient for the specific WPPS configuration and web page processing method developed. Furthermore, the WPPS configuration file can be amplified by the specification of different membership functions (e.g., triangular function, trapezoidal function, and Gaussian function) as well as application of the concept of uncertainty to different features including color, font, and text content.

Another interesting challenge consists of the integration of knowledge bases represented by means of formalisms differing from RDF and OWL, for example, Datalog \pm [43] or **H μ L ϵ X** [173]. This will allow the application of alternative approaches for WPP.

Furthermore, limitations of the application of various membership functions (reflecting inaccuracies on a web page layout) for computing qualitative spatial relations and R-tree should be further investigated.

7.2.2 Web Accessibility

Multi-Axial Navigation Model and Navigation

The MANM enables navigation through basic navigable elements providing the user with different paths and transitions. We see the potential development of the MANM in building axes through complex objects and dynamically generating axes for reading their content. These aspects and transitions should be well specified, intuitive for blind users, and not complicate their navigation.

One promising solution includes the generation of web pages' abstractions which consist of different granularity. For example, the most abstract representation will generate axes which go through the most salient and important elements, such as titles and subtitles. Another more detailed abstraction will possess axes which cover certain parts of a web page consisting of content with relatively higher information gains. The most detailed abstraction will possess axes which contain items corresponding to tokens of a relevant object. Research on this aspect will provide blind users with different representations and gives them the opportunity to choose the most convenient and relevant representations for a certain task.

To optimize information search on a web page, we also see further research opportunities in developing summary generation methods taking into account peculiarities of a web page layout or methods eliminating part of the content, and thus imitating a skim reading. The first approach is directly related to methods introduced within the field of Natural Language Processing, whereas the second approach requires a detailed analysis of a web page layout and identification of the most prominent structural parts of a web page which should be read. In regards to the second approach, for example, skimming a table entails reading only the most important and salient parts according to specified constraints (e.g., time or number of items).

Blindzilla

Further work on the Blindzilla prototypes should include the development of the end-to-end solution for blind users with different collections of WPP methods corresponding to certain website genres and tasks as well as methods for genre identification. Another functional enhancement should be related to the possibility of sharing both instances of the MANM and navigation trails with a certain web community and be able to replay these navigation trails.

Queries for Spatial Relations Analysis

This section provides a list of SPARQL queries which were used in the analysis of spatial relations between elements of web pages' layout—blocks (see Section 3.11). Types of spatial relationships, such as topological, direction, distance, and alignment, are considered (see Sections A.1, A.2, A.3, and A.4). SPARQL queries were evaluated against ontologies generated for web pages from the WPPS-HTML-DS1 dataset [72]. The schematic representation of the experimental ontology is illustrated in Figure A.1. A block has a nonzero area and corresponds to the Definition 3.5 on page 67. For querying the ontologies Jena [10] and SPARQL 1.1 [282] were used.

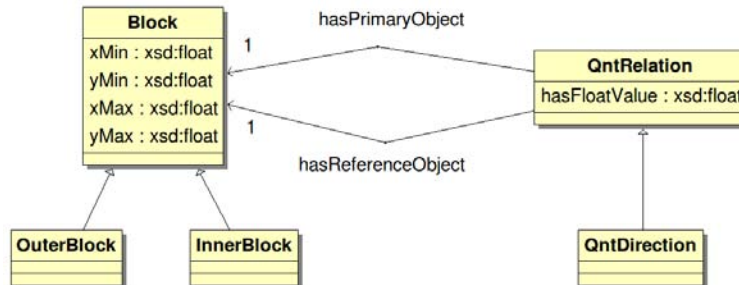


Figure A.1: Concepts and properties of the experimental ontology used in the analysis of web pages' spatial relations

A.1 RCC8

For the analysis of topology of the web page layout, the RCC8 relations, such as DC, EC, PO, TPP, NTPP, and EQUAL, between outer blocks were considered (see Listings A.1, A.2, A.3, A.4, A.5, and A.6).

Listing A.1: Number of DC relations on a web page

```
1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (
15 # do not consider symmetric relations
16 xsd:string(?prim) > xsd:string(?ref)
17 # block is on the visible part of the canvas
18 && ?xMaxPrim >0 && ?yMaxPrim > 0
19 && ?xMaxRef >0 && ?yMaxRef > 0
20 &&
21 ( ?xMaxPrim < ?xMinRef || ?xMinPrim > ?xMaxRef
22 || ?yMaxPrim < ?yMinRef || ?yMinPrim > ?yMaxRef )
23 )
24 }
```

Listing A.2: Number of EC relations on a web page

```
1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (
15 # do not consider symmetric relations
16 xsd:string(?prim) > xsd:string(?ref)
17 # block is on the visible part of the canvas
18 && ?xMaxPrim >0 && ?yMaxPrim > 0
19 && ?xMaxRef >0 && ?yMaxRef > 0
20 &&
21 ( (?xMaxPrim = ?xMinRef
22 && ?yMaxPrim >= ?yMinRef && ?yMinPrim <= ?yMaxRef)
23 || (?xMinPrim = ?xMaxRef
24 && ?yMaxPrim >= ?yMinRef && ?yMinPrim <= ?yMaxRef)
25 || (?yMaxPrim = ?yMinRef
26 && ?xMaxPrim >= ?xMinRef && ?xMinPrim <= ?xMaxRef)
27 || (?yMinPrim = ?yMaxRef
28 && ?xMaxPrim >= ?xMinRef && ?xMinPrim <= ?xMaxRef) )
29 )
30 }
```

Listing A.3: Number of PO relations on a web page

```
1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (
15 # do not consider symmetric relations
16 xsd:string(?prim) > xsd:string(?ref)
17 # blocks are on the visible part of the canvas
18 && ?xMaxPrim > 0 && ?yMaxPrim > 0
19 && ?xMaxRef > 0 && ?yMaxRef > 0

21 &&
22 # primary and reference blocks should have common points
23 ?xMinPrim < ?xMaxRef && ?xMaxPrim > ?xMinRef
24 && ?yMinPrim < ?yMaxRef && ?yMaxPrim > ?yMinRef
25 # primary block should have points outside the reference block
26 && ( ?xMinPrim < ?xMinRef || ?yMinPrim < ?yMinRef
27 || ?xMaxPrim > ?xMaxRef || ?yMaxPrim > ?yMaxRef )
28 # reference block should have points outside the primary block
29 && ( ?xMinPrim > ?xMinRef || ?yMinPrim > ?yMinRef
30 || ?xMaxPrim < ?xMaxRef || ?yMaxPrim < ?yMaxRef )
31 )
32 }
```

Listing A.4: Number of TPP relations on a web page

```
1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (
15 ?prim != ?ref
16 # blocks are on the visible part of the canvas
17 && ?xMaxPrim > 0 && ?yMaxPrim > 0
18 && ?xMaxRef > 0 && ?yMaxRef > 0
19 &&
20 # primary block inside reference block or equal
21 ?xMinPrim >= ?xMinRef && ?yMinPrim >= ?yMinRef
22 && ?xMaxPrim <= ?xMaxRef && ?yMaxPrim <= ?yMaxRef
23 # one of the border should coincide
24 && ( ?xMinPrim = ?xMinRef || ?yMinPrim = ?yMinRef
25 || ?xMaxPrim = ?xMaxRef || ?yMaxPrim = ?yMaxRef )
```

```

26 # reference block should have points outside the primary block
27 && ( ?xMinPrim > ?xMinRef || ?yMinPrim > ?yMinRef
28 || ?xMaxPrim < ?xMaxRef || ?yMaxPrim < ?yMaxRef )
29 )
30 }

```

Listing A.5: Number of NTPP relations on a web page

```

1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (
15 ?prim != ?ref
16 # blocks are on the visible part of the canvas
17 && ?xMaxPrim > 0 && ?yMaxPrim > 0
18 && ?xMaxRef > 0 && ?yMaxRef > 0
19 &&
20 # primary block inside reference block
21 ?xMinPrim > ?xMinRef && ?yMinPrim > ?yMinRef
22 && ?xMaxPrim < ?xMaxRef && ?yMaxPrim < ?yMaxRef
23 )
24 }

```

Listing A.6: Number of EQUAL relations on a web page

```

1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (
15 # do not consider symmetric relations
16 xsd:string(?prim) > xsd:string(?ref)
17 # blocks are on the visible part of the canvas
18 && ?xMaxPrim > 0 && ?yMaxPrim > 0
19 && ?xMaxRef > 0 && ?yMaxRef > 0
20 &&
21 # primary block equals reference block
22 ?xMinPrim = ?xMinRef && ?yMinPrim = ?yMinRef
23 && ?xMaxPrim = ?xMaxRef && ?yMaxPrim = ?yMaxRef
24 )
25 }

```


A.2 Quantitative Directions

Listing A.7 presents a SPARQL query used in the analysis of quantitative directions between outer blocks. `QntDirection` was defined between the blocks which were positioned at a distance not exceeding 50px from each other.

Listing A.7: Number of different groups of quantitative directions on a web page

```
1 SELECT ?anglrnd (count(?anglrnd) as ?cnt)
2 WHERE {
3   ?dir com:hasPrimaryObject ?prim.
4   ?dir com:hasReferenceObject ?ref.
5   ?dir rdf:type qntb:QntDirection.
6   ?prim rdf:type go:OuterBlock.
7   ?ref rdf:type go:OuterBlock.
8   ?dir com:hasFloatValue ?anglvalGroup.

10  ?prim qntb:xMin ?xMinPrim.
11  ?prim qntb:yMin ?yMinPrim.
12  ?prim qntb:xMax ?xMaxPrim.
13  ?prim qntb:yMax ?yMaxPrim.

15  ?ref qntb:xMin ?xMinRef.
16  ?ref qntb:yMin ?yMinRef.
17  ?ref qntb:xMax ?xMaxRef.
18  ?ref qntb:yMax ?yMaxRef.
19  LET (?anglrnd := fn:round(?anglvalGroup))
20  FILTER (
21    # block is on the visible part of the canvas
22    && ?xMaxPrim > 0 && ?yMaxPrim > 0
23    && ?xMaxRef > 0 && ?yMaxRef > 0
24  )
25 }
26 GROUP BY ?anglrnd
27 ORDER BY ?anglrnd
```

A.3 Quantitative Distances Between Border Projections

Listing A.8 illustrates a SPARQL query applied in the analysis of quantitative distances by example of a distance between left border projections of the blocks.

Listing A.8: Number of different groups of quantitative distances between blocks' left borders

```
1 SELECT ?dif (count(?dif) as ?cnt)
2 WHERE {
3   ?prim qntb:xMin ?xMinPrim.
4   ?prim qntb:yMin ?yMinPrim.
5   ?prim qntb:xMax ?xMaxPrim.
6   ?prim qntb:yMax ?yMaxPrim.

8   ?ref qntb:xMin ?xMinRef.
9   ?ref qntb:yMin ?yMinRef.
10  ?ref qntb:xMax ?xMaxRef.
11  ?ref qntb:yMax ?yMaxRef.
12  LET (?dif := fn:abs(fn:round(?xMinPrim - ?xMinRef)) )
13  FILTER (
14    # block is on the visible part of the canvas
```

```

15  && ?xMaxPrim >0 && ?yMaxPrim > 0
16  && ?xMaxRef >0 && ?yMaxRef > 0
17  &&
18  ?dif > 0 # exclude aligned elements
19  )
20 }
21 GROUP BY ?dif
22 ORDER BY DESC(?cnt)

```

A.4 Alignment relations

For the analysis of geometry of the web page layout, alignment relations, such as LA, CV, RA, TA, CH, and BA, between outer blocks were considered (see Listings A.9, A.10, A.11, A.12, A.13, and A.14).

Listing A.9: Number of LA relations on a web page

```

1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (
15 # do not consider symmetric relations
16 xsd:string(?prim) > xsd:string(?ref)
17 # block is on the visible part of the canvas
18 && ?xMaxPrim >0 && ?yMaxPrim > 0
19 && ?xMaxRef >0 && ?yMaxRef > 0
20 &&
21 ?xMinPrim = ?xMinRef
22 )
23 }

```

Listing A.10: Number of CV relations on a web page

```

1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (

```

```

15 # do not consider symmetric relations
16 xsd:string(?prim) > xsd:string(?ref)
17 # block is on the visible part of the canvas
18 && ?xMaxPrim >0 && ?yMaxPrim > 0
19 && ?xMaxRef >0 && ?yMaxRef > 0
20 &&
21 ((?xMaxPrim + ?xMinPrim) / 2) = ((?xMaxRef + ?xMinRef) / 2)
22 )
23 }

```

Listing A.11: Number of RA relations on a web page

```

1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (
15 # do not consider symmetric relations
16 xsd:string(?prim) > xsd:string(?ref)
17 # block is on the visible part of the canvas
18 && ?xMaxPrim >0 && ?yMaxPrim > 0
19 && ?xMaxRef >0 && ?yMaxRef > 0
20 &&
21 ?xMaxPrim = ?xMaxRef
22 )
23 }

```

Listing A.12: Number of TA relations on a web page

```

1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (
15 # do not consider symmetric relations
16 xsd:string(?prim) > xsd:string(?ref)
17 # block is on the visible part of the canvas
18 && ?xMaxPrim >0 && ?yMaxPrim > 0
19 && ?xMaxRef >0 && ?yMaxRef > 0
20 &&
21 ?yMinPrim = ?yMinRef
22 )
23 }

```

Listing A.13: Number of CH relations on a web page

```
1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (
15 # do not consider symmetric relations
16 xsd:string(?prim) > xsd:string(?ref)
17 # block is on the visible part of the canvas
18 && ?xMaxPrim > 0 && ?yMaxPrim > 0
19 && ?xMaxRef > 0 && ?yMaxRef > 0
20 &&
21 ((?yMaxPrim + ?yMinPrim) / 2) = ((?yMaxRef + ?yMinRef) / 2)
22 )
23 }
```

Listing A.14: Number of BA relations on a web page

```
1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (
15 # do not consider symmetric relations
16 xsd:string(?prim) > xsd:string(?ref)
17 # block is on the visible part of the canvas
18 && ?xMaxPrim > 0 && ?yMaxPrim > 0
19 && ?xMaxRef > 0 && ?yMaxRef > 0
20 &&
21 ?yMaxPrim = ?yMaxRef
22 )
23 }
```

Listing A.15: Number of NA relations on a web page

```
1 SELECT (count(*) as ?qnt)
2 WHERE {
3 ?prim rdf:type go:OuterBlock.
4 ?prim qntb:xMin ?xMinPrim.
5 ?prim qntb:yMin ?yMinPrim.
6 ?prim qntb:xMax ?xMaxPrim.
7 ?prim qntb:yMax ?yMaxPrim.
```

```

9 ?ref rdf:type go:OuterBlock.
10 ?ref qntb:xMin ?xMinRef.
11 ?ref qntb:yMin ?yMinRef.
12 ?ref qntb:xMax ?xMaxRef.
13 ?ref qntb:yMax ?yMaxRef.
14 FILTER (
15   # do not consider symmetric relations
16   xsd:string(?prim) > xsd:string(?ref)
17   # block is on the visible part of the canvas
18   && ?xMaxPrim >0 && ?yMaxPrim > 0
19   && ?xMaxRef >0 && ?yMaxRef > 0
20   &&
21   !(
22     ?xMinPrim = ?xMinRef
23     ||
24     ((?yMaxPrim + ?yMinPrim) / 2) = ((?yMaxRef + ?yMinRef) / 2)
25     ||
26     ?xMaxPrim = ?xMaxRef
27     ||
28     ?yMinPrim = ?yMinRef
29     ||
30     ((?xMaxPrim + ?xMinPrim) / 2) = ((?xMaxRef + ?xMinRef) / 2)
31     ||
32     ?yMaxPrim = ?yMaxRef
33   )
34 )
35 }

```


A Survey of Blind Users of the Web

There is no typical screen reader user... but we can learn much about typical behavior.

— Jared Smith & Jonathan Whiting, *The Legend of the Typical Screen Reader User*, WebAIM, 2009

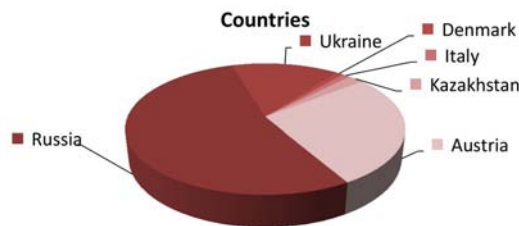
This survey, which can also be found in [78], was produced and conducted by the author of this thesis from June 1, 2013 to August 31, 2013. Questions were formulated based on the research of the ABBA [236] and TAMCROW [237] projects of TU Vienna, of which the author was also involved in. Moreover, some important aspects which served as preconditions for carrying out this inquiry are mentioned in our previous work [21, 83]. The survey reflects the present state and relevant problems within the field of Web Accessibility along with major works (with subjective and objective evaluations) such as [29, 56, 212, 226, 292–296]. In contrast to existing inquiries, we highlight the problem of blind users familiarizing themselves with new web pages and the accessibility of web pages of different genres. It is worth mentioning that the majority of questions are subjective and thus reflect the personal opinions of blind users regarding relevant issues. As a result of the responses acquired, we believe that this survey could motivate the researchers of web accessibility and web authors who work collaboratively to be able to resolve the problems that have been identified.

In the questionnaire provided to the sightless web users, we first asked them general questions regarding their gender, age, and blindness (see Section B.1). Then we asked respondents about using computer and assistive technology and their opinion regarding the complexity of contemporary web pages (see Section B.2). We also posed questions about visiting new web pages and the difference between becoming familiar with new web pages independently versus with assistance (see Section B.3). We emphasized the main challenges of navigating a web page’s content and identification of its main logical components such as the navigation menu and main content (see Sections B.4 and B.5). Furthermore, we are concerned with the accessibility of contemporary web pages of different genres which provide different services, such as: search engines (see Section B.6.1), social media websites (see Section B.6.2), news

websites (see Section B.6.3), web forums (see Section B.6.4), on-line shops (see Section B.6.5), and weblogs (see Section B.6.6).

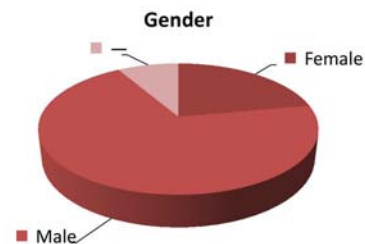
B.1 Demographics of Respondents

A total of 95 participants who are blind took part in the survey. The majority of participants are from Russia, Austria, and Ukraine (see Figure B.1). Most of the respondents attend different organizations such as Russian Community of Blinds¹ Blinden- und Sehbehindertenverband Österreich² (BSVÖ), and different community groups with support from the local libraries, for example, Saint Petersburg State Library for the Blind³ and Perm Krai Library for the Blind⁴ in Russia.



Country	% of Respondents
Russia	53.7%
Austria	27.4%
Ukraine	14.7%
Kazakhstan	2.1%
Denmark	1.1%
Italy	1.1%

Figure B.1: Country of respondents



Gender	% of Respondents
Male	69.5%
Female	22.1%
—	8.4%

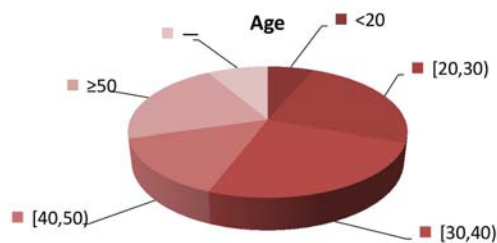
Figure B.2: Gender of respondents

¹<http://www.vos.org.ru/>

²<http://www.blindenverband.at/>, <http://www.braille.at/>

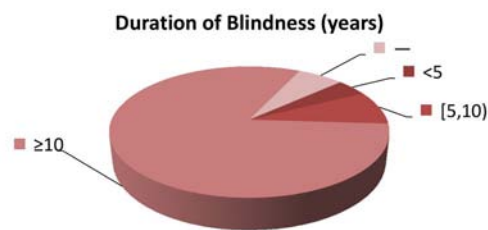
³<http://gbs.spb.ru/>

⁴<http://permksbs.ru/>



Age (years)	% of Respondents
<20	6.3%
[20, 30)	24.2%
[30, 40)	25.3%
[40, 50)	14.7%
≥50	21.1%
—	8.4%

Figure B.3: Age of respondents



Response (years)	% of Respondents
<5	4.2%
[5, 10)	8.4%
≥10	81.1%
—	6.3%

Figure B.4: For how many years have you been blind (or unable to see text and graphics on web pages)?

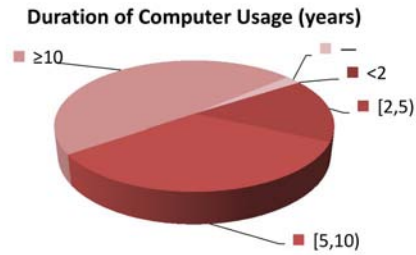
The participants are comprised of a variety of ages (see Figure B.3), from 15 to 79. An overwhelming majority have been blind for more than ten years (see Figure B.4) while 33.7% of respondents are blind from birth and 40.0% lost their vision before the age of five.

B.2 Expertise and Preferences⁵

All the respondents are computer-literate and the majority (82.1%) have been using the computer for at least five years (see Figure B.5). This allows us to consider them as intermediate or even advanced users. Furthermore, 89.5% use the computer every day. Windows (94.3%) and therefore Windows-based screen readers, such as JAWS (71.6%) and NVDA (50.5%), are extremely popular among blind users (see Figures B.6 and B.7). It is important to note that the users surveyed are satisfied with the screen reader of their choice (see Figure B.8). The necessity of good audio output is confirmed by the fact that 72.6% of users exclusively rely on the screen reader and do not use a Braille display (see Figure B.9). This is mainly due to the high cost of the Braille display.

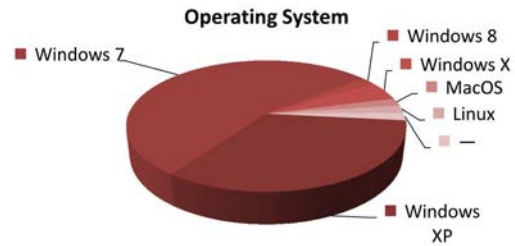
73.7% of participants have been using the Web for at least five years (see Figure B.10), 78.9% surf the Web every day, and 13.7% using the Web a few times a week but not every day. The complexity of web pages is primarily assessed as average by respondents (rate 3 within the interval [1, 5], see Figure B.11). We believe that this rate should not differ much from the opinion of sighted users due to the presence of differing complexities of web pages in general.

⁵Some questions presented in this section are similar to the questions presented by WebAIM in [296]. The results acquired by us do not contradict the results reported by WebAIM.



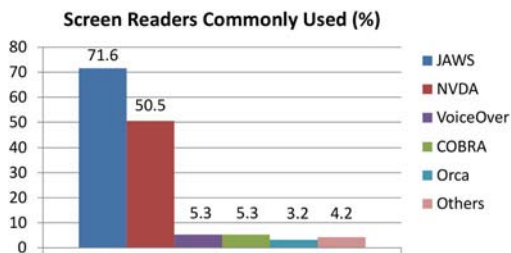
Response (years)	% of Respondents
<2	0%
[2, 5)	15.8%
[5, 10)	33.7%
≥10	48.4%
—	2.1%

Figure B.5: How many years have you been using the computer being blind?



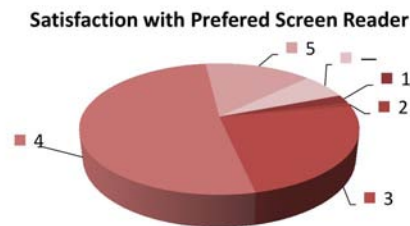
Response	% of Respondents
Windows 8	1.9%
Windows 7	54.3%
Windows XP	33.3%
Windows X (version was not specified)	4.8%
GNU/Linux	1.9%
Mac OS X	1.9%
—	1.9%

Figure B.6: Which operating system do you mostly use?



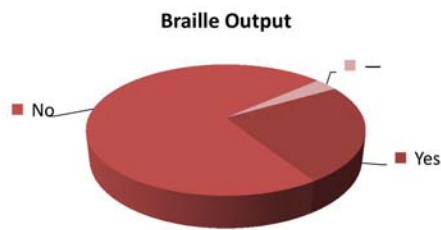
Response	% of Respondents
JAWS	71.6%
NVDA	50.5%
VoiceOver	5.3%
COBRA	5.3%
Orca	3.2%
Others	4.2%
—	2.1%

Figure B.7: Which screen reader do you mostly use? Specify several screen readers if you use them equally as often



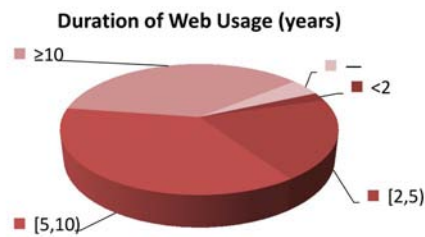
Response	% of Respondents
1	2.1%
2	1.1%
3	24.2%
4	51.6%
5	14.7%
—	6.3%

Figure B.8: How satisfied are you with the screen readers you mostly use? (Choose from 1 to 5: 1 if you are absolutely not satisfied with the screen reader, 5 if you are completely satisfied.)



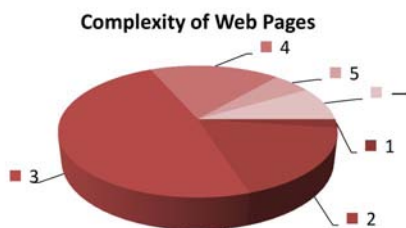
Response	% of Respondents
Yes	24.2%
No	72.6%
—	3.2%

Figure B.9: Do you use a Braille display with your screen reader?



Response (years)	% of Respondents
<2	2.1%
[2, 5)	20.0%
[5, 10)	36.8%
≥10	36.8%
—	4.2%

Figure B.10: How many years have you been using the Web being blind?



Response	% of Respondents
1	2.1%
2	17.9%
3	48.4%
4	17.9%
5	5.3%
—	8.4%

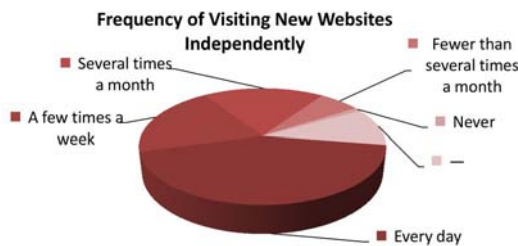
Figure B.11: What is your general opinion regarding the complexity of web pages? (Choose from 1 to 5: 1 if web pages are very easy to understand in general, 5 if web pages perceptually are very complicated.)

B.3 Becoming Familiar with New Web Pages

As Figures B.12 and B.13 show, 43.2% of respondents get to know new websites independently on a daily basis, and only 4.2% navigate through the content of early unseen websites with the help of a sighted assistant on a daily basis. Interestingly, only 23.2% of respondents assert that familiarizing themselves with new websites with sighted assistant is easier than independent learning while 28.4% have the opposite opinion (see Figure B.14). According to our interview of individual respondents, the assistant can explain to the blind user the logical structure of a web page, its content, and help to find necessary information. On the other hand, the assistant relies

on visual cues and uses terms related to the description of visual and spatial characteristics of a web page's elements and its layout, which is neither accessible nor useful for blind users. A blind user's mental model, defined as a result of the screen reader operating mainly over the DOM tree, differs greatly from the mental model of a sighted user. Thus, information about spatial allocation of web objects and visual features (for example, that relevant text is to the right of an image with a certain content) is not useful for the blind or its use is quite limited (some screen readers, like JAWS, can provide information based on the computed CSS attributes, e.g., color of the text and background, font style, size). As one of the respondents explained, the best assistance a blind user can obtain in searching for necessary information on a web page is from another blind user, who knows the web page well and uses the same software.

There were many people who did not answer the questions in this section. Afterwards, it was discovered that some respondents were not able to choose a specific answer while others, unfortunately, did not understand the question.



Response	% of Respondents
Every day	43.2%
A few times a week but not every day	20.0%
Several times a month	18.9%
Fewer than several times a month	6.3%
Never	1.1%
—	10.5%

Figure B.12: How often do you familiarize yourself with new websites (unaided)?



Response	% of Respondents
Every day	4.2%
A few times a week but not every day	1.1%
Several times a month	11.6%
Fewer than several times a month	30.5%
Never	31.6%
—	21.1%

Figure B.13: How often do you familiarize yourself with new websites with the assistance of a sighted person?

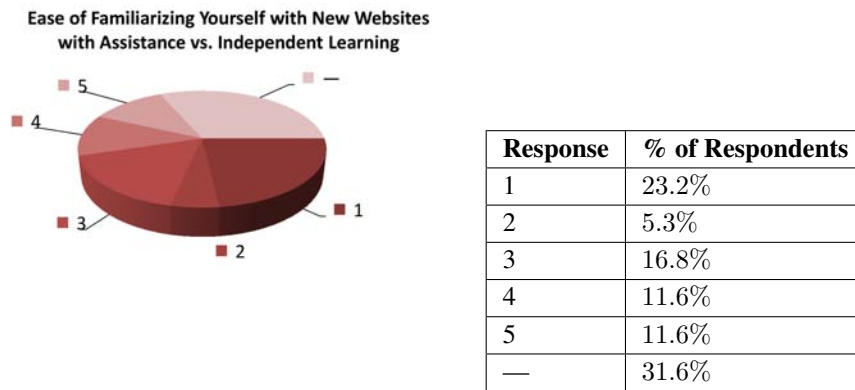


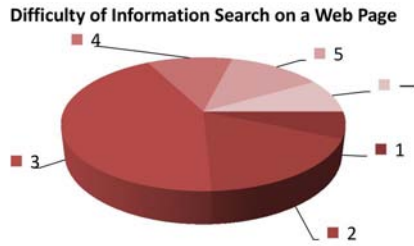
Figure B.14: To what extent is familiarizing yourself with new websites with the help of your preferred sighted assistant easier than independent learning? (Choose from 1 to 5: 1 if independent learning is greatly easier than learning with your assistant, 5 if familiarizing yourself with new websites with the assistant is greatly easier than independent learning.)

B.4 General Navigation

67.4% of respondents report that it is not easy to find relevant information on a web page (see Figure B.15), highlighting the weak accessibility of contemporary web pages. According to results illustrated in Figure B.16, there are different opinions regarding the images and their influence on web page navigation. This is due to the fact that, on the one hand, today's web pages have a lot of images without properly assigned alternative text and thus cannot be understood and leveraged by sightless users (this fact is also confirmed in [29]), making web pages even more cumbersome. However, on the other hand, images with alternative text can be used by blind users as landmarks and thus speed up the navigation on a web page, especially during the next visit. For example, blind users can often find the main navigation menu near the logotype. For specific web forums, blind users can also remember that a content of the post is close to the profile picture with a specific title attached. Thus, these images can be used later for fast access to relevant content. Images are also used to figure out the boundaries of the target content.

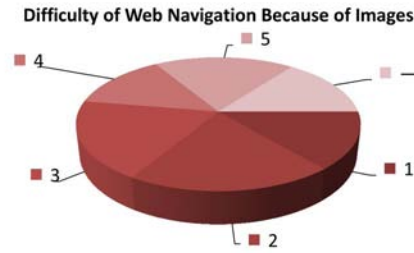
Regarding Figures B.17 and B.18, sightless users can identify navigation menus and main content, however, they have to use much more effort than sighted persons. Blind people usually define navigation menus as a certain list of links which allows them to go through the main sections of a website and they expect to find it at the beginning of the web page serialized by the screen reader. Therefore, if such a list is located quite far away from the beginning of the serialized sequence of web page elements, it is very likely to not be found.

The main content of a web page is usually searched by the blind via navigation through the HTML heading tags and, if there is no such tags or they are misused, blind users mainly have to read through the page and rely on the textual content.



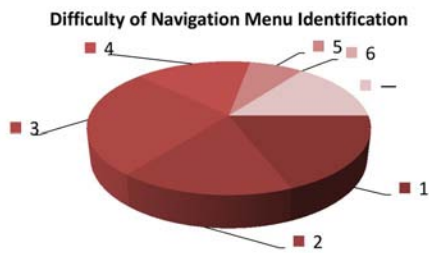
Response	% of Respondents
1	6.3%
2	17.9%
3	43.2%
4	11.6%
5	12.6%
—	8.4%

Figure B.15: How difficult is it for you to find desired information on a typical web page that contains it? For example, to find the price of a book on a web page which has the list of books with prices. (Choose from 1 to 5: 1 if to find such information is very easy, 5 if to find such information is very difficult.)



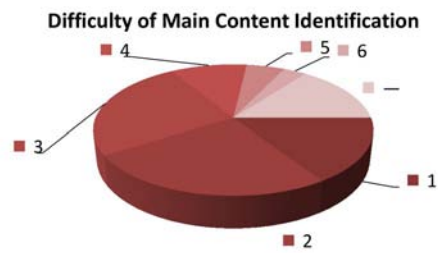
Response	% of Respondents
1	13.7%
2	20.0%
3	18.9%
4	13.7%
5	18.9%
—	14.7%

Figure B.16: How complicated do images make navigation on a web page in general? (Choose from 1 to 5: 1 if image content does not disturb the navigation at all, 5 if it is hard to navigate when image content occurs on a web page.)



Response	% of Respondents
1	18.9%
2	16.8%
3	26.3%
4	15.8%
5	7.4%
6	0.0%
—	14.7%

Figure B.17: How difficult is it for you to distinguish an area with the navigation menu on a typical web page from the rest during your navigation on the page? (Choose from 1 to 5: 1 if it is very simple, 5 if it is very difficult, 6 if it is impossible or almost impossible.)

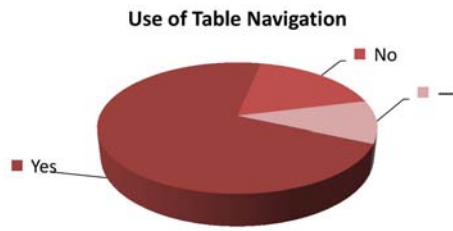


Response	% of Respondents
1	15.8%
2	25.3%
3	25.3%
4	10.5%
5	5.3%
6	3.2%
—	14.7%

Figure B.18: How difficult is it for you to distinguish the main content on a typical web page from the rest during your navigation on the page? (Choose from 1 to 5: 1 if it is very simple, 5 if it is very difficult, 6 if it is impossible or almost impossible.)

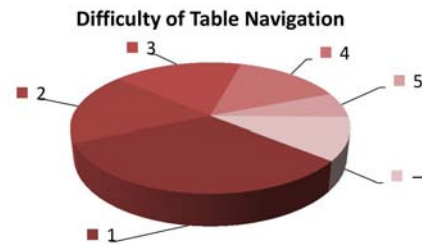
B.5 Navigation Through Tables and Lists

Most of the respondents use table and list navigations and are quite satisfied with their simplicity (see Figures B.19, B.20, B.21, and B.22). However, it is worth mentioning that screen readers navigate through HTML tables and HTML lists, which, unfortunately, are very often used to lay out a web page's content [146].



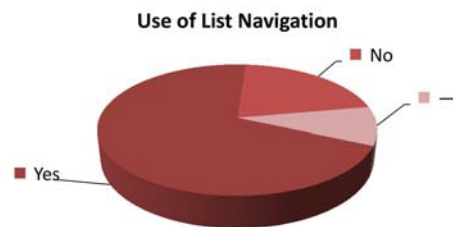
Response	% of Respondents
Yes	71.6%
No	17.9%
—	10.5%

Figure B.19: Do you use navigation through the content of a table? For example, to go from row to row, from column to column, or from cell to cell



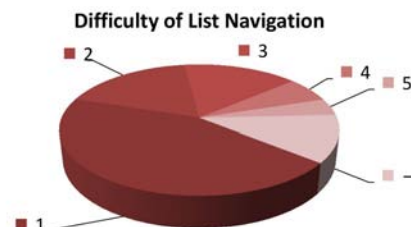
Response	% of Respondents
1	32.6%
2	17.9%
3	17.9%
4	14.7%
5	6.3%
—	10.5%

Figure B.20: What is the level of difficulty in navigating through the content of a table? (Choose from 1 to 5: 1 if it is very simple, 5 if it is very difficult.)



Response	% of Respondents
Yes	69.5%
No	21.1%
—	9.5%

Figure B.21: Do you use navigation through the content of a list? For example, to go from item to item of the list



Response	% of Respondents
1	44.2%
2	17.9%
3	15.8%
4	6.3%
5	4.2%
—	11.6%

Figure B.22: What is the level of difficulty in navigating through the content of a list? (Choose from 1 to 5: 1 if it is very simple, 5 if it is very difficult.)

B.6 Accessibility of Web Pages by Genres

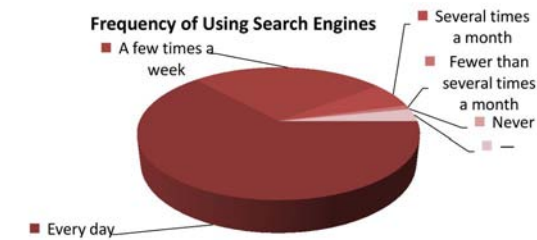
Search engines are the most essential component of web surfing. The survey concluded that 63.2% of respondents use search engines every day (see Figure B.23) and 76.8% of users are satisfied with their simple (or very simple) interface (see Figure B.24). Several cooperators of the ABBA project [236], who are blind, primarily use Wikipedia⁶ for information search and will refer to search engines only if required information cannot be found there. This is because all web pages of Wikipedia have a unified and relatively simple interface that allow blind users to easily find and navigate to the relevant content on the target web page of Wikipedia. According to our survey, Wikipedia is not as popular as search engines. Only 24.2% of respondents use it several times a week (or every day, see Figure B.25) while 54.7% use only search engines for their information search (see Figure B.27). However, most of the respondents (62.1%) report that the interface of Wikipedia is simple to use (or very simple, see Figure B.26). We believe these results are due to the fact that search engines, in contrast to Wikipedia, can answer more complex and diverse requests and index a much bigger set of information resources. Wikipedia, in turn, mainly conveys encyclopedia knowledge and requires the user to set more precise definition of the query string.

A total of 27.4% use social media websites every day and 50.5% use it several times a week (or every day, see Figure B.28), regardless of the complexity of their interface (40.0% believe that it is complex, see Figure B.29). News websites are also quite popular among blind people: 22.1% visit them every day and 46.3% visit them several times a week (or every day, see Figure B.30). However, not many people report any difficulties with the interface of news web pages (27.4% believe that it is difficult or very difficult and 50.5% believe that it is at least not simple, see Figure B.31). The use of web forums, on-line shops, and weblogs are less common in the blind community. For example, 9.5% of respondents informed us that they visit web forums every day and 27.4% visit them several times a week (or everyday, see Figure B.32). 34.7% report that navigation through the post on web forums is difficult (or very difficult) and 51.6% believe that it is at least not simple, see Figure B.33. Only 6.3% visit on-line shops several times a week (or everyday, see Figure B.34). The biggest group of respondents (26.3%) believe that the task of finding a desired product is of average complexity (see Figure B.35). A small minority of respondents (3.2%) visit weblogs several times a week (or everyday, see Figure B.36) and the biggest group of respondents who replied assert that it is very difficult to navigate through posts in weblogs (11.6%, see Figure B.37). In addition, 18.9% say that it is difficult (or very difficult) to navigate through posts while 61.1% did not answer this question.

It is important to note that for some questions the biggest group of respondents did not provide any answer. We ascertained two main reasons for this: One is that some participants did not have enough time to answer all the questions of the questionnaire and another reason is that some people did not understand certain terminology, such as weblogs or web forum, and thought that it would be reasonable to skip the question.

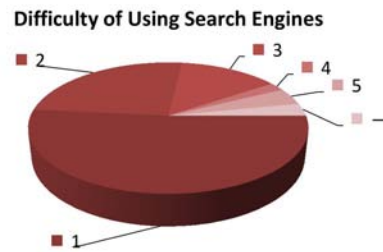
⁶<http://www.wikipedia.org/>

B.6.1 Information Search



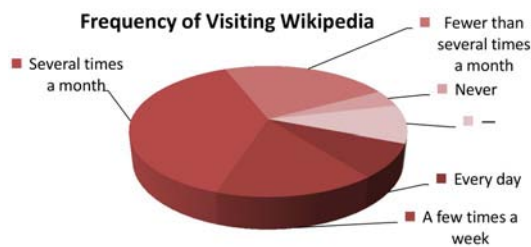
Response	% of Respondents
Every day	63.2%
A few times a week but not every day	26.3%
Several times a month	6.3%
Fewer than several times a month	1.1%
Never	0.0%
—	3.2%

Figure B.23: How often do you use search engines like Google, Yahoo or Bing?



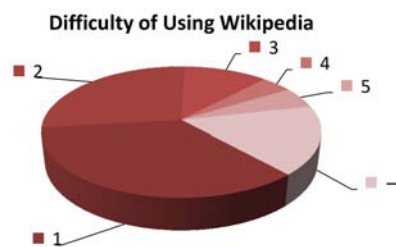
Response	% of Respondents
1	51.6%
2	25.3%
3	13.7%
4	2.1%
5	4.2%
—	3.2%

Figure B.24: How difficult is it for you to use search engines? (Choose from 1 to 5: 1 if it is very simple, 5 if it is very difficult.)



Response	% of Respondents
Every day	8.4%
A few times a week but not every day	15.8%
Several times a month	38.9%
Fewer than several times a month	23.2%
Never	4.2%
—	9.5%

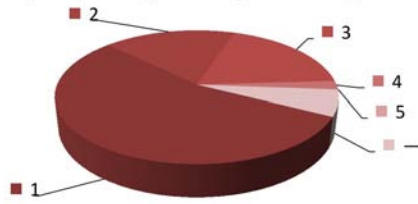
Figure B.25: How often do you visit Wikipedia?



Response	% of Respondents
1	34.7%
2	27.4%
3	11.6%
4	4.2%
5	5.3%
—	16.8%

Figure B.26: How difficult is it for you to use Wikipedia? (Choose from 1 to 5: 1 if it is very simple, 5 if it is very difficult.)

Frequency of Using Search Engines vs. Wikipedia

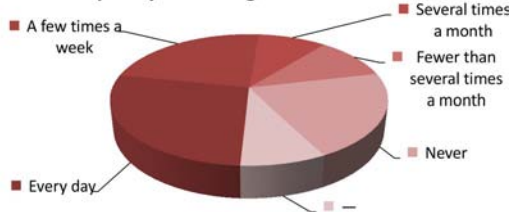


Response	% of Respondents
1	54.7%
2	17.9%
3	18.9%
4	2.1%
5	0.0%
—	6.3%

Figure B.27: For information searches, do you use a search engine or Wikipedia more often? (Choose from 1 to 5: 1 if you only use a search engine, 3 if you use a search engine and Wikipedia equally often, 5 if you only use Wikipedia.)

B.6.2 Social Media

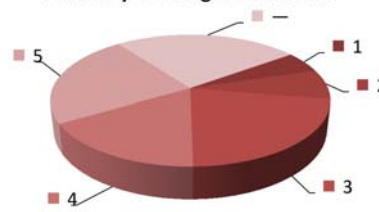
Frequency of Visiting Social Media



Response	% of Respondents
Every day	27.4%
A few times a week but not every day	23.2%
Several times a month	9.5%
Fewer than several times a month	10.5%
Never	21.1%
—	8.4%

Figure B.28: How frequently do you visit social media websites (for example, Facebook, VKontakte, or Twitter)?

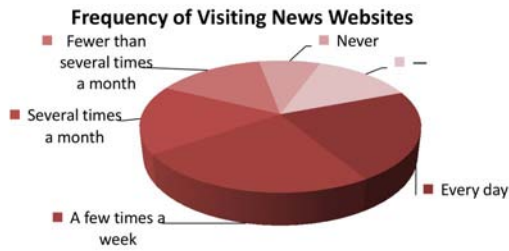
Difficulty of Using Social Media



Response	% of Respondents
1	5.3%
2	7.4%
3	22.1%
4	16.8%
5	23.2%
—	25.3%

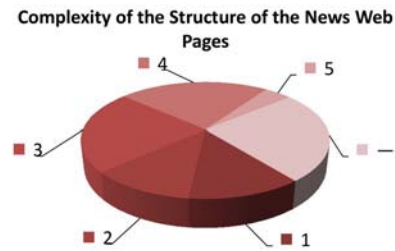
Figure B.29: How difficult is it for you to work with the interface of a typical social media website? (Choose from 1 to 5: 1 if it is very simple, 5 if it is very difficult.)

B.6.3 News Websites



Response	% of Respondents
Every day	22.1%
A few times a week but not every day	24.2%
Several times a month	16.8%
Fewer than several times a month	14.7%
Never	8.4%
—	13.7%

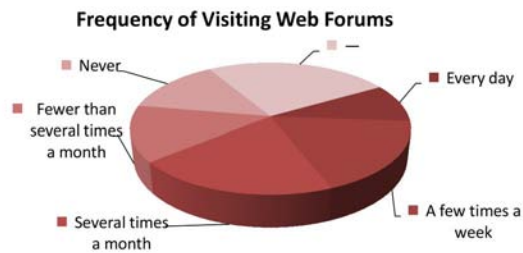
Figure B.30: How often do you read news articles on news websites?



Response	% of Respondents
1	12.6%
2	11.6%
3	23.2%
4	23.2%
5	4.2%
—	25.3%

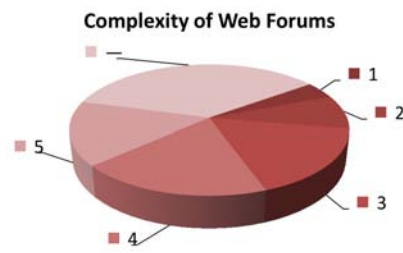
Figure B.31: How well are news web pages structured from your point of view? For example, can you distinguish different categories of news (such as business or sport) on the main web page and navigate through the categories? Can you distinguish a content of the news article from the rest? (Choose from 1 to 5: 1 if a typical news web page provides quite a poor structure, 5 if a typical news web page has quite a convenient structure.)

B.6.4 Web Forums



Response	% of Respondents
Every day	9.5%
A few times a week but not every day	17.9%
Several times a month	20.0%
Fewer than several times a month	13.7%
Never	13.7%
—	25.3%

Figure B.32: How often do you visit web forums?



Response	% of Respondents
1	4.2%
2	8.4%
3	16.8%
4	18.9%
5	15.8%
—	35.8%

Figure B.33: How difficult is it to navigate through the posts on a typical web forum? (Choose from 1 to 5: 1 if navigation on the typical web forum is straightforward, 5 if navigation is very complicated.)

B.6.5 On-line Shops

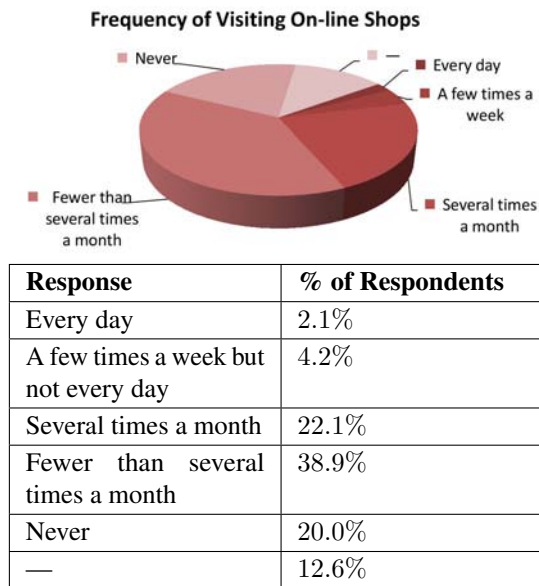


Figure B.34: How often do you visit on-line shops?

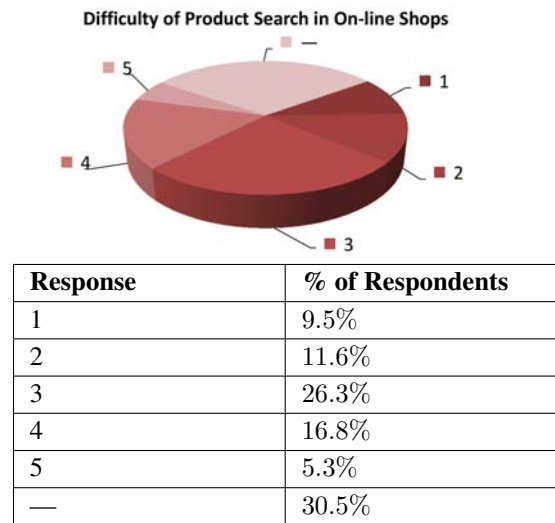
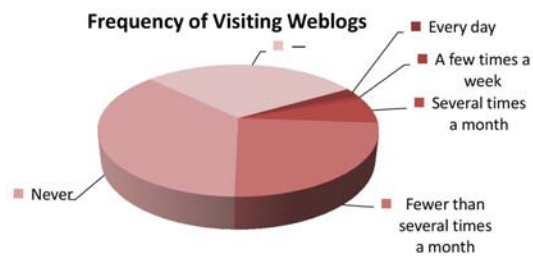


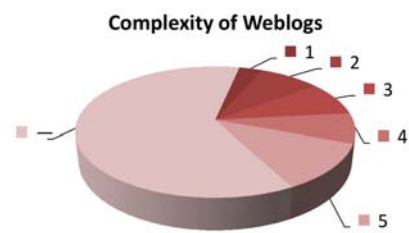
Figure B.35: How difficult is it to find the desired product in a typical on-line shop? (Choose from 1 to 5: 1 if it is very simple, 5 if it is very hard.)

B.6.6 Weblogs



Response	% of Respondents
Every day	2.1%
A few times a week but not every day	1.1%
Several times a month	6.3%
Fewer than several times a month	24.2%
Never	36.8%
—	29.5%

Figure B.36: How often do you visit weblogs? (For example, Live Journal.)



Response	% of Respondents
1	3.2%
2	8.4%
3	8.4%
4	7.4%
5	11.6%
—	61.1%

Figure B.37: How difficult is it for you to navigate through the posts in a typical weblog? (Choose from 1 to 5: 1 if navigation on the typical weblog is straightforward, 5 if navigation is very complicated.)

Bibliography

- [1] B. Adelberg. NoDoSE—a tool for semi-automatically extracting structured and semistructured data from text documents. *ACM SIGMOD Record*, 27(2):283–294, June 1998.
- [2] M. Aiello, C. Monz, L. Todoran, and M. Worring. Document understanding for a broad class of documents. *International Journal on Document Analysis and Recognition*, 5(1):1–16, Nov. 2002.
- [3] M. Aiello, I. E. Pratt-Hartmann, and J. F. van Benthem, editors. *Handbook of spatial logics*. Springer, Dordrecht, The Netherlands, 2007.
- [4] M. Aiello and A. M. W. Smeulders. Thick 2D relations for document understanding. *Information Sciences*, 167(1-4):147–176, Dec. 2004.
- [5] C. Alexandraki, A. Paramythis, N. Maou, and C. Stephanidis. Web accessibility through adaptation. *Computers Helping People with Special Needs*, 3118:302–309, 2004.
- [6] S. P. Algur and P. S. Hiremath. Visual clue based extraction of web data from flat and nested data records. In *Proceedings of International Conference on Management of Data (COMAD 2006)*, pages 207–210, Dehli, India, 2006.
- [7] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, Nov. 1983.
- [8] American Foundation for the Blind. American Foundation for the Blind (AFB). <http://www.afb.org>. Date accessed: 10.12.2012.
- [9] G. Antoniou and F. van Harmelen. *A semantic web primer*. The MIT Press, London, second edition, 2008.
- [10] Apache. Jena, version 2.7.4. <http://jena.apache.org/>, 2012.
- [11] Apple. VoiceOver, version 3. <http://www.apple.com/accessibility/voiceover/>, 2010.
- [12] A. Arasu and H. Garcia-Molina. Extracting structured data from web pages. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, pages 337–348, New York, 2003. ACM.
- [13] C. Asakawa and T. Itoh. User interface of a home page reader. In *Proceedings of the Third international ACM Conference on Assistive Technologies (Marina del Rey, California, United States, April 15 - 17, 1998)*, pages 149–156, New York, NY, 1998. ACM Press.
- [14] F. Baader, I. Horrocks, and U. Sattler. Description Logics as Ontology Languages for the Semantic Web. *Mechanizing Mathematical Reasoning*, 2605:228–248, 2005.
- [15] P. Balbiani, J.-F. Condotta, and L. F. n. Del Cerro. A model for reasoning about bidimensional temporal relations. In A. G. Cohn, L. Schubert, and S. C. Shapiro, editors, *In Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98)*, pages 124–130. Morgan Kaufmann, 1998.
- [16] P. Balbiani, J.-F. Condotta, and L. F. n. Del Cerro. A new tractable subclass of the rectangle algebra. In *Proceedings of the 16th international joint conference on Artificial intelligence (IJCAI-99)*, volume 1, pages 442–447, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.
- [17] P. Balbiani, J.-F. Condotta, and L. F. n. Del Cerro. Tractability Results in the Block Algebra. *Journal of Logic and Computation*, 12(5):885–909, Oct. 2002.
- [18] P. Bartalos and M. Bielikova. An approach to object-ontology mapping. In *Proceedings of the 2nd IFIP Central and East European Conference on Software Engineering Techniques (CEE-SET 2007)*, pages 9–16. Slovak

University of Technology, 2007.

- [19] A. Bashmakov and I. Bashmakov. *Intelligent information technologies (Russian)*. Bauman MSTU, Moscow, 2005.
- [20] BAUM Retec AG. COBRA, version 10. <http://www.baum.de/cms/en/cobra10/>, 2013.
- [21] R. Baumgartner, R. R. Fayzrakhmanov, W. Holzinger, B. Krüpl, M. C. Göbel, D. Klein, and R. Gattringer. Web 2.0 vision for the blind. In *Proceedings of Web Science Conference 2010 (WebSci'10), Raleigh, USA, 26–27 April, 2010*, pages 1–8, 2010.
- [22] R. Baumgartner, S. Flesca, and G. Gottlob. The Elog Web Extraction Language. In R. Nieuwenhuis and A. Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 2001)*, volume 2250, pages 548–560, Havana, Cuba, 2001. Springer.
- [23] R. Baumgartner, O. Frölich, and G. Gottlob. The Lixto systems applications in business intelligence. *The Semantic Web: Research and Applications*, 4519:16–26, 2007.
- [24] S. Bechhofer, S. Harper, and D. Lunn. SADie: Semantic Annotation for Accessibility. *The Semantic Web*, 4273:101–115, 2006.
- [25] M. K. Bergman. The Deep Web: Surfacing hidden value. *Journal of Electronic Publishing (JEP)*, 7(1):1–17, 2001.
- [26] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifiers (URI): Generic Syntax: RFC 2396. Rfc 2396, RFC Editor, 1998.
- [27] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, May 2001.
- [28] P. A. Bernstein and L. M. Haas. Information integration in the enterprise. *Communications of the ACM - Enterprise Information Integration and Other Tools for Merging Data*, 51(9):72–79, 2008.
- [29] J. P. Bigham, A. C. Cavender, J. T. Brudvik, J. O. Wobbrock, R. E. Lander, and R. E. Ladner. WebinSitu: A comparative analysis of blind and sighted browsing behavior. In *Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility (Assets '07)*, pages 51–58, New York, 2007. ACM.
- [30] J. P. Bigham, R. S. Kaminsky, R. E. Ladner, O. M. Danielsson, and G. L. Hempton. WebInSight: Making Web Images Accessible. In *Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility (Assets '06)*, pages 181–188, New York, NY, 2006. Acm.
- [31] J. P. Bigham, T. Lau, and J. Nichols. Trailblazer: enabling blind users to blaze trails through the web. In *Proceedings of the 13th International Conference on Intelligent User Interfaces*, volume 09, pages 177–186, Sanibel Island, Florida, USA, 2009. ACM: New York, NY, USA.
- [32] J. P. Bigham, C. M. Prince, and R. E. Ladner. Addressing performance and security in a screen reading web application that enables accessibility anywhere. *2008 Eighth International Conference on Web Engineering*, pages 273–284, July 2008.
- [33] J. P. Bigham, C. M. Prince, and R. E. Ladner. WebAnywhere: a screen reader on-the-go. In *Proceedings of the 2008 International Cross-Disciplinary Conference on Web Accessibility*, pages 73–82, New York, 2008. ACM Press.
- [34] M. H. Blackmon, P. G. Polson, M. Kitajima, and C. Lewis. Cognitive walkthrough for the web. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '02)*, pages 463–470, New York, 2002. ACM.
- [35] Y. Borodin. *Bridging the Web Accessibility Divide*. Phd thesis, Stony Brook University, 2009.
- [36] Y. Borodin, F. Ahmed, M. A. Islam, Y. Puzis, V. Melnyk, S. Feng, I. V. Ramakrishnan, and G. Dausch. Hearsay: a new generation context-driven multi-modal assistive web browser. In *Proceedings of the 19th international conference on World wide web (WWW'10)*, pages 1233–1236, Raleigh, NC, USA, 2010. ACM New York, NY, USA.
- [37] Y. Borodin, J. P. Bigham, G. Dausch, and I. Ramakrishnan. More than meets the eye: A survey of screen-reader browsing strategies. In *Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A '10)*, page Article No. 13. ACM, 2010.
- [38] M. Brambring. Mobility and orientation processes of the blind. In D. H. Warren and E. R. Strelow, editors, *Electronic Spatial Sensing for the Blind*, pages 493–508, Lancaster, 1984. Nijhoff.

- [39] D. Buttler, L. Liu, and C. Pu. A fully automated object extraction system for the World Wide Web. *Proceedings 21st International Conference on Distributed Computing Systems*, pages 361–370, 2001.
- [40] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma. Extracting content structure for web pages based on visual representation. In *Proceedings of the 5th Asia-Pacific web conference on Web technologies and applications*, pages 406–417, Xian, China, 2003. Springer-Verlag.
- [41] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma. VIPS : A vision-based page segmentation algorithm. Technical report msr-tr-2003-79, Microsoft Research Asia, Beijing, 2003.
- [42] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma. Block-based web search. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 456–463, Sheffield, United Kingdom, 2004. ACM.
- [43] A. Cali, G. Gottlob, T. Lukasiewicz, and A. Pieris. Datalog+/-: A family of languages for ontology querying. *Datalog Reloaded*, 6702:351–368, 2011.
- [44] M. E. Califf and R. J. Mooney. Relational learning of pattern-match rules for information extraction. In *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence (AAAI '99/IAAI '99)*, pages 328–334, Menlo Park, 1999. American Association for Artificial Intelligence.
- [45] C.-H. Chang, M. Kayed, M. R. Girgis, and K. F. Shaalan. A Survey of Web Information Extraction Systems. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1411–1428, Oct. 2006.
- [46] C.-H. Chang and S.-C. Lui. IEPAD: information extraction based on pattern discovery. In V. Y. Shen, N. Saito, M. R. Lyu, and M. E. Zurko, editors, *Proceedings of the 10th international conference on World Wide Web*, pages 681–688, Hong Kong, Hong Kong, 2001. ACM.
- [47] C. L. Chen. Fire Vox. <http://www.firevox.clcworld.net/>, 2013.
- [48] C. L. Chen and T. V. Raman. AxsJAX: a talking translation bot using google IM: bringing web-2.0 applications to life. In *Proceedings of the 2008 international cross-disciplinary conference on Web accessibility (W4A)*, pages 54–56, New York, New York, USA, 2008. ACM Press.
- [49] B. L. Clarke. A calculus of individuals based on connection. *Notre Dame Journal of Formal Logic*, 22(3):204–218, July 1981.
- [50] E. Clementini, P. Di Felice, and D. Hernández. Qualitative representation of positional information. *Artificial Intelligence*, 95(2):317–356, 1997.
- [51] A. G. Cohn. Qualitative spatial representation and reasoning techniques. In G. Brewka, C. Habel, and B. Nebel, editors, *KI-97: Advances in Artificial Intelligence*, volume 1303, pages 1–30. Springer Berlin, Berlin, Germany, May 1997.
- [52] A. G. Cohn and S. M. Hazarika. Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae*, 46(1-2):1–29, 2001.
- [53] A. G. Cohn and J. Renz. Qualitative spatial representation and reasoning. In F. Harmelen, V. Lifschitz, and B. Porter, editors, *Handbook of Knowledge Representation*, chapter Chapter 13, pages 551–596. Elsevier, London, UK, first edit edition, 2008.
- [54] M. Cosulschi, N. Constantinescu, and M. Gabroveanu. Classification and comparison of information structures from a web page. *Annals of University of Craiova, Math. Comp. Sci. Ser.*, 31:109–121, 2004.
- [55] J. Cowie and W. Lehnert. Information Extraction. *Communications of the ACM*, 39(1):80–91, 1996.
- [56] K. P. Coyne and J. Nielsen. *Beyond alt text: Making the web easy to use for users with disabilities*. Nielsen Norman Group, Fremont, 2001.
- [57] V. Crescenzi, G. Mecca, and P. Merialdo. RoadRunner: Towards automatic data extraction from large web sites. In *Proceedings of the 27th International Conference on Very Large Data Bases*, pages 109–118, Rome, Italy, 2001. Morgan Kaufmann Publishers Inc.
- [58] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, C. Ursu, M. Dimitrov, M. Dowman, N. Aswani, I. Roberts, Y. Li, A. Shafiri, and A. Funk. Developing language processing components with GATE version 5 (a user guide). A user guide Gate 2, The University of Sheffield, 2009.
- [59] A. Dieberger. A city metaphor to support navigation in complex information spaces. *Spatial Information Theory. A Theoretical Basis for GIS*, 1329:53–67, Dec. 1997.

- [60] Dolphin. SuperNova Screen Reader, version 13.02. <http://www.yourdolphin.com/>, 2012.
- [61] T. Dönz. *Extracting Text Coordinates and Segmenting Tables in the Visual Representation of Web Pages*. Bachelor thesis, Vienna University of Technology, 2006.
- [62] E. C. Dragut, T. Kabisch, C. Yu, and U. Leser. A hierarchical approach to model web query interfaces for web source integration. In *Proceedings of the VLDB Endowment*, pages 325–336. VLDB Endowment, 2009.
- [63] P. Duygulu and V. Atalay. A hierarchical representation of form documents for identification and retrieval. *International Journal on Document Analysis and Recognition*, 5(1):17–27, Nov. 2002.
- [64] M. J. Egenhofer and R. D. Franzosa. Point-set topological spatial relations. *International Journal of Geographical Information Systems*, 5(2):161–174, 1991.
- [65] M. Y. Erlewine. Ubiquity : Designing a Multilingual Natural Language Interface Features of a Natural Syntax. In *SIGIR Workshop on Information Access in a Multilingual World*, page 4, Boston, Massachusetts USA, 2009.
- [66] A. Faaborg and H. Lieberman. A goal-oriented web browser. *Proceedings of the SIGCHI conference on Human Factors in computing systems - CHI '06*, page 751, 2006.
- [67] S. Faulkner. WAI ARIA landmark role tests. <http://www.html5accessibility.com/tests/landmarks.html>. Date accessed: 15.05.2013, 2011.
- [68] R. R. Fayzrakhmanov. *Development of ontology-based web form recognition methods in meta-search systems (Russian)*. Master thesis, Perm State Technical University, 2008.
- [69] R. R. Fayzrakhmanov. RegExpTokenizer. <http://code.google.com/p/reg-exp-tokenizer/>, 2008.
- [70] R. R. Fayzrakhmanov. Css drawing order detection. <https://code.google.com/p/css-drawing-order-detection/>, 2011.
- [71] R. R. Fayzrakhmanov. Information extraction from web pages based on their visual representation. In A. Harth and N. Koch, editors, *Proceedings of the 11th International Conference on Web Engineering (ICWE'11), PhD Symposium, Paphos, Cyprus, 20–24 June, 2011*, pages 342–346, Heidelberg, 2011. Springer.
- [72] R. R. Fayzrakhmanov. WPPS-HTML-DS1 dataset. <http://www.dbai.tuwien.ac.at/staff/fayzrakh/wpps/datasets/WPPS-HTML-DS1.zip>, 2011.
- [73] R. R. Fayzrakhmanov. A blocks-based geometric model of web pages for automatic processing and information extraction. *Science and Business: Development Ways*, 9(15):56–64, 2012.
- [74] R. R. Fayzrakhmanov. A new approach for ensuring web accessibility for the sightless users based on enhancing navigation characteristics of web pages (Russian). *Ingenerniy Vestnik Dona*, 23(4-2):10, 2012.
- [75] R. R. Fayzrakhmanov. Web Page Processing System. <http://www.dbai.tuwien.ac.at/staff/fayzrakh/wpps/>, 2012.
- [76] R. R. Fayzrakhmanov. WPPS: A framework for web page processing. In X. S. Wang, I. Cruz, A. Delis, and G. Huang, editors, *In Proceedings of the 13th International Conference on Web Information Systems Engineering (WISE'2012), Demo Session, Paphos, Cyprus, 28–30 November, 2012*, pages 800–803. Springer, 2012.
- [77] R. R. Fayzrakhmanov. WPPS: A novel and comprehensive framework for web page understanding and information extraction. In B. White and P. Isaías, editors, *Proceeding of the International Conference IADIS WWW/Internet, Madrid, Spain, 18–21 October, 2012*, pages 19–26, Madrid, 2012. IADIS Press.
- [78] R. R. Fayzrakhmanov. A Survey of Blind Users of the Web 01.06.2013–31.08.2013 (<http://www.dbai.tuwien.ac.at/staff/fayzrakh/wa/survey2013/>), 2013.
- [79] R. R. Fayzrakhmanov. The Multi-Axial Navigation Model for Improving Web Page Accessibility (<http://www.dbai.tuwien.ac.at/staff/fayzrakh/wa/MANM2.zip>), 2013.
- [80] R. R. Fayzrakhmanov. The Unified Ontological Model (UOM2) formalizing some aspects of the web page conceptualization (<http://www.dbai.tuwien.ac.at/staff/fayzrakh/wpps/UOM2.zip>), 2013.
- [81] R. R. Fayzrakhmanov, E. V. Dolgova, and R. A. Fayzrakhmanov. Modeling of Information Representation in the Tasks of Web Page Processing and Web Information Extraction (Russian). *Vestnik of the Izhevsk State Technical University*, (2):176–179, 2011.
- [82] R. R. Fayzrakhmanov, M. C. Göbel, W. Holzinger, B. Krüpl, and R. Baumgartner. A unified ontology-based web page model for improving accessibility. In *Proceedings of the 19th international conference on World Wide Web (WWW'2010), Raleigh, USA, April 26–30, 2010*, pages 1087–1088, New York, 2010. ACM.

- [83] R. R. Fayzrakhmanov, M. C. Göbel, W. Holzinger, B. Krüpl, A. Mager, and R. Baumgartner. Modelling web navigation with the user in mind. In *Proceedings of the International Cross Disciplinary Conference on Web Accessibility (W4A'2010)*, Raleigh, USA, 26–27 April, 2010, pages 1–4, New York, 2010. ACM.
- [84] R. R. Fayzrakhmanov, C. Herzog, and I. Kordomatis. Web objects identification for web automation: Objects and their features. Technical report (dbai-tr-2013-80), Institute of Information Systems, Vienna University of Technology, Vienna, 2013.
- [85] R. R. Fayzrakhmanov, B. Krüpl, and W. Holzinger. Multiaxial navigation system for improving web accessibility. In *Proceedings of the 2nd International Internet Conference on Innovative Technologies: Theory, Tools, Implementation (INNOTECH'2010)*, Perm, 2010. Perm State Technical University.
- [86] Federal Agency on Technical Regulation and Metrology. GOST P 52872-2007. Internet resources. Accessibility requirements for the visually handicapped. National Standard of Russian Federation. Gost, Federal Agency on Technical Regulation and Metrology, 2007.
- [87] E. Ferrara, P. D. Meo, G. Fiumara, and R. Baumgartner. Web data extraction, applications and techniques: A survey. *CoRR*, abs/1207.0(June):20, 2012.
- [88] R. Fikes, P. Hayes, and I. Horrocks. OWL-QL — A language for deductive query answering on the semantic. KSL Technical Report 03-14. Ksl technical report 03-14, Knowledge Systems Laboratory, Stanford University, Stanford, CA, 2003.
- [89] A. U. Frank. Qualitative spatial reasoning about cardinal directions. In *Proceedings of the 7th Austrian Conference on Artificial Intelligence*, pages 157–167, 1991.
- [90] Freedom Scientific. JAWS for Windows Screen Reading Software, version 14. <http://www.freedomscientific.com/products/fs/jaws-product-page.asp>, 2012.
- [91] T. Furche, G. Gottlob, G. Grasso, X. Guo, G. Orsi, and C. Schallhart. OPAL: Automated form understanding for the deep web. In *Proceedings of the 21st international conference on World Wide Web*, pages 829–838, New York, 2012. ACM.
- [92] T. Furche, G. Gottlob, G. Grasso, C. Schallhart, and A. Sellers. OXPath: a language for scalable, memory-efficient data extraction from Web applications. In *Proceedings of the VLDB Endowment*, pages 1016–1027. VLDB Endowment, 2011.
- [93] W. Gatterbauer and P. Bohunsky. Table extraction using spatial reasoning on the CSS2 visual box model. In *Proceedings of the 21st National Conference on Artificial Intelligence*, volume 2, pages 1313–1318, Boston, Massachusetts, USA, 2006. AAAI Press.
- [94] W. Gatterbauer, P. Bohunsky, M. Herzog, B. Krüpl, and B. Pollak. Towards domain-independent information extraction from web tables. In *Proceedings of the 16th international conference on World Wide Web - WWW '07*, pages 71–80, Banff, Alberta, Canada, 2007. ACM Press.
- [95] W. Gatterbauer, B. Krüpl, W. Holzinger, and M. Herzog. Web information extraction using eupeptic data in web tables. In *Proceedings of the 1st International Workshop on Representation and Analysis of Web Space (RAWS 2005)*, pages 41–48, Prague, Czech Republic, 2005. VSB - Technical University of Ostrava.
- [96] A. Geyer-Schulz. *Fuzzy rule-based expert systems and genetic machine learning*. Physica-Verlag, Heidelberg, 1995.
- [97] C. Goble, S. Harper, and R. Stevens. The travails of visually impaired web travellers. In *Proceedings of the Eleventh ACM on Hypertext and Hypermedia (HYPERTEXT '00)*, pages 1–10, New York, New York, USA, 2000. ACM Press.
- [98] G. Gottlob, C. Koch, R. Baumgartner, M. Herzog, and S. Flesca. The Lixto data extraction project: Back and forth between theory and practice. In *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS '04)*, pages 1–12, New York, 2004. ACM.
- [99] B. C. Grau, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneider, and U. Sattler. OWL 2: The next step for OWL. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(4):309–322, 2008.
- [100] M. Grobe. RDF, Jena, SparQL and the 'Semantic Web'. In *Proceedings of the 37th Annual ACM SIGUCCS Fall Conference (SIGUCCS '09)*, pages 131–138, St. Louis, MO, 2009. ACM.
- [101] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

- [102] X.-D. Gu, J. Chen, W.-Y. Ma, and G.-L. Chen. Visual based content understanding towards web adaptation. *Adaptive Hypermedia and Adaptive Web-Based Systems*, 2347:164–173, 2002.
- [103] N. Guarino. Formal ontology and information systems. In *Proc. of International Conference On Formal Ontology In Information Systems*, pages 3–15, Trento, Italy, 1998. IOS Press; Amsterdam.
- [104] H. Guo, J. Mahmud, Y. Borodin, A. Stent, and I. V. Ramakrishnan. A general approach for partitioning web page content based on geometric and style information. *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, 2:929–933, Sept. 2007.
- [105] A. Guttman. R-trees: a dynamic index structure for spatial searching. In *Proceedings ACM SIGMOD Conference on Management of Data*, pages 47–57, Boston, Massachusetts, USA, 1984.
- [106] J. Ha, R. M. Haralick, and I. T. Phillips. Recursive X-Y cut using bounding boxes of connected components. In *Proceedings of the Third International Conference on Document Analysis and Recognition*, volume 2, pages 952–955, Montreal, Canada, 1995. IEEE Computer Society.
- [107] V. Haarslev, R. Möller, and M. Wessel. Querying the Semantic Web with Racer + nRQL. In S. Bechhofer, V. Haarslev, C. Lutz, and R. Moeller, editors, *Proceedings of the KI-2004 International Workshop on Applications of Description Logics (ADL'04)*, pages 1–10. CEUR, 2004.
- [108] V. Hakkoymaz. A specification model for temporal and spatial relations of segments in multimedia presentations. *Journal Of Digital Information Management*, 8(2):136–146, 2010.
- [109] J. Hammer, J. Mchugh, and H. Garcia-Molina. Semistructured data: the TSIMMIS experience. In *In Proceedings of the 1st East-European Symposium on Advances in Databases and Information Systems (ADBIS)*, pages 1–8, St. Petersburg, Russia, 1997.
- [110] R. M. Haralick. Document image understanding: geometric and logical layout. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '94)*, pages 385 – 390. IEEE Computer Society, 1994.
- [111] S. Harper, C. Goble, and R. Stevens. A pilot study to examine the mobility problems of visually impaired users travelling the Web. *ACM SIGCAPH Computers and the Physically Handicapped*, 1(68):10–19, Sept. 2000.
- [112] S. Harper, C. Goble, and R. Stevens. Web mobility guidelines for visually impaired surfers. *Journal of Research and Practice in Information Technology*, 33(1):30–41, 2001.
- [113] S. Harper and Y. Yeliz, editors. *Web accessibility: A foundation for Research*. Springer, London, 2008.
- [114] T. Hassan. Object-level document analysis of PDF files. In *Proceedings of the 9th ACM symposium on Document engineering - DocEng '09*, pages 47–55, New York, New York, USA, 2009. ACM Press.
- [115] T. Hassan. User-guided wrapping of PDF documents using graph matching techniques. In *Proceedings of the 2009 10th International Conference on Document Analysis and Recognition*, pages 631–635. IEEE Computer Society, July 2009.
- [116] T. Hassan. *User-guided information extraction from print-oriented documents*. Phd thesis, Vienna University of Technology, 2010.
- [117] T. Hassan and R. Baumgartner. Table recognition and understanding from PDF files. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition*, volume 2, pages 1143–1147, Paraná, Argentina, 2007. IEEE Computer Society.
- [118] X. He, D. Cai, J.-R. Wen, W.-Y. Ma, and H.-J. Zhang. Clustering and searching WWW images using link and page layout analysis. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 3(2):1–33, May 2007.
- [119] J. Hendler, N. Shadbolt, W. Hall, T. Berners-Lee, and D. Weitzner. Web science: an interdisciplinary approach to understanding the Web. *Communications of the ACM*, 51(7):60–69, 2008.
- [120] D. Hernández, E. Clementini, and P. Di Felice. Qualitative distances. *Spatial Information Theory. A Theoretical Basis for GIS. LNCS*, 988:45–57, 1995.
- [121] C. Herzog, I. Kordomatis, W. Holzinger, R. R. Fayzrakhmanov, and B. Krüpl-Sypien. Feature-based object identification for web automation. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing (SAC' 13)*, pages 742–749, Coimbra, 2013. ACM.
- [122] P. S. Hiremath and S. P. Algur. Extraction of data from web pages: a vision based approach. *International Journal of Computer and Information Science and Engineering*, 3:50–59, 2009.

- [123] P. S. Hiremath and S. P. Algur. Extraction Of flat and nested data records from web pages. *International Journal on Computer Science and Engineering*, 2(1):36–45, 2010.
- [124] P. S. Hiremath, S. S. Benchalli, S. P. Algur, and R. V. Udupadi. Mining data regions from web pages. In *Proceedings of the International Conference on Management of Data (COMAD 2005)*, pages 130–138, Hyderabad, India, 2005.
- [125] A. Hogue and D. Karger. Thresher: automating the unwrapping of semantic content from the World Wide Web. In A. Ellis and T. Hagino, editors, *Proceedings of the 14th international conference on World Wide Web*, pages 86–95, New York, 2005. ACM.
- [126] W. Holzinger, B. Krüpl, and M. Herzog. Using ontologies for extracting product features from web pages. In *Proceedings of the 5th International Conference on The Semantic Web (ISWC'06)*, pages 286–299. Springer, 2006.
- [127] W. Holzinger and B. Krüpl-Sypien. Gestalt Ontology. Personal communication, 2009.
- [128] C.-N. Hsu and M.-T. Dung. Generating finite-state transducers for semi-structured data extraction from the Web. *Information Systems*, 23(8):521–538, Dec. 1998.
- [129] N. Hurst, W. Li, and K. Marriott. Review of automatic document formatting. In *Proceedings of the 9th ACM symposium on Document engineering (DocEng '09)*, pages 99–108, Munich, Germany, 2009. ACM Press.
- [130] IBM. IBM developer guidelines. <http://www-03.ibm.com/able/guidelines/>. Date accessed: 10.12.2012.
- [131] J.-S. R. Jang, C.-T. Sun, and E. Mizutani. *Neuro-fuzzy and soft computing: A computational approach to learning and machine intelligence*. Prentice Hall, first edit edition, 1997.
- [132] S. Jul and G. W. Furnas. Navigation in electronic worlds. *ACM SIGCHI Bulletin*, 29(4):44–49, 1997.
- [133] A. Kalyanpur, D. J. Pastor, S. Battle, and J. Padget. Automatic mapping of OWL ontologies into Java. In *Proceedings of the 16th International Conference of Software Engineering and Knowledge Engineering*, pages 98–103, 2004.
- [134] V. Kashyap, C. Bussler, and M. Moran. *The Semantic Web. Semantics for data and services on the Web*. Springer, Berlin, 2008.
- [135] M. Kaye and C.-H. Chang. FiVaTech: page-level Web data extraction from template pages. *IEEE Transactions on Knowledge and Data Engineering*, 22(2):249–263, Feb. 2010.
- [136] J. Keith. *DOM Scripting: Web design with JavaScript and the Document Object Model*. Springer, New York, the USA, 1st edition, 2005.
- [137] B. Kelly, D. Sloan, L. Phipps, H. Petrie, and F. Hamilton. Forcing standardization or accommodating diversity? A framework for applying the WCAG in the real world. In *Proceedings of the 2005 International Cross-Disciplinary Workshop on Web Accessibility (W4A)*, pages 46–54. ACM Press, 2005.
- [138] R. Khare and Y. An. An empirical study on using hidden markov model for search interface segmentation. In *Proceedings of the 18th ACM conference on Information and knowledge management (CIKM '09)*, page 17, New York, 2009. ACM Press.
- [139] M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. *Journal of the ACM (JACM)*, 42(4):741–843, 1995.
- [140] T. Kistler and J. Marais. WebL—a programming language for the Web. In *Proceedings of the seventh international conference on World Wide Web*, pages 259–270. Elsevier, 1998.
- [141] S. Klink, A. Dengel, and T. Kieninger. Document structure analysis based on layout and textual features. In *In Proceedings of International Workshop on Document Analysis Systems (DAS2000)*, pages 99–111. IAPR, 2000.
- [142] J. Kocibova, K. Klos, O. Lehecka, M. Kudelka, and V. Snasel. Web page analysis: experiments based on discussion and purchase web patterns. In *2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology Workshops (WI-IATW '07)*, pages 221–225, Washington, Nov. 2007. IEEE Computer Society.
- [143] K. Koffka. *Principles of Gestalt psychology*. Harcourt, Brace & Co., New York, 1 edition edition, 1935.
- [144] J. Kong, K. Zhang, and X. Zeng. Spatial graph grammars for graphical user interfaces. *ACM Transactions on Computer-Human Interaction*, 13(2):268–307, June 2006.
- [145] I. Kordomatis, C. Herzog, R. R. Fayzrakhmanov, B. Krüpl-Sypien, W. Holzinger, and R. Baumgartner. Web

- object identification for web automation and meta-search. In *Proceedings of the 3rd International Conference on Web Intelligence, Mining and Semantics (WIMS '13)*, pages 1–12, New York, 2013. ACM.
- [146] M. Kovacevic, M. Diligenti, M. Gori, and V. Milutinovic. Visual adjacency multigraphs – a novel approach for a web page classification. In *In Proceedings of the Workshop on Statistical Approaches to Web Mining (SAWM'2004)*, pages 38–49, Pisa, Italy, 2004. University of Pisa.
- [147] P. Kremen and Z. Kouba. Ontology-driven information system design. *IEEE Transactions on Systems, Man, and Cybernetics. Part C: Applications and Reviews*, 42(3):334–344, 2012.
- [148] M. Krishnamoorthy, G. Nagy, and S. Seth. Syntactic segmentation and labeling of digitized pages from technical journals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(7):737–747, 1993.
- [149] B. Krüpl and M. Herzog. Visually guided bottom-up table detection and segmentation in web documents. In *Proceedings of the 15th international conference on World Wide Web - WWW '06*, pages 933–934, New York, New York, USA, 2006. ACM Press.
- [150] B. Krüpl, M. Herzog, and W. Gatterbauer. Using visual cues for extraction of tabular data from arbitrary HTML documents. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, pages 1000–1001, Chiba, Japan, 2005. ACM.
- [151] B. Krüpl-Sypien, R. R. Fayzrakhmanov, W. Holzinger, M. Panzenböck, and R. Baumgartner. A versatile model for web page representation, information extraction and content re-packaging. In M. Hardy and F. W. Tompa, editors, *In Proceedings of the 11th ACM Symposium on Document Engineering (DocEng2011), Mountain View, USA, 19–22 September, 2011*, pages 129–138, New York, 2011. ACM.
- [152] S. Kuhlins and R. Tredwell. Toolkits for generating wrappers. *Objects, Components, Architectures, Services, and Applications for a Networked World*, 2591/2003:184–198, 2003.
- [153] N. Kushmerick, D. Weld, and R. Doorenbos. Wrapper induction for information extraction. In *Proceedings of the 15th International Conference on Artificial Intelligence (IJCAI)*, pages 729–735. University of Washington, 1997.
- [154] P. Ladkin. Models of axioms for time intervals. In *Proceedings of the 6th National Conference on Artificial Intelligence*, volume 1, pages 234–239. Morgan Kaufmann, 1987.
- [155] A. H. F. Laender, B. Ribeiro-Neto, and A. S. da Silva. DEByE - Data extraction by example. *Data & Knowledge Engineering*, 40(2):121–154, 2002.
- [156] A. H. F. Laender, B. A. Ribeiro-Neto, A. S. da Silva, and J. S. Teixeira. A brief survey of web data extraction tools. *ACM SIGMOD Record*, 31(2):84, June 2002.
- [157] T. Lau, J. Cerruti, G. Manzato, M. Bengualid, J. P. Bigham, and J. Nichols. A conversational interface to web automation. In *Proceedings of the 23rd annual ACM symposium on User interface software and technology (UIST '10)*, pages 229–238. ACM, 2010.
- [158] B. Leporini. Google news: how user-friendly is it for the blind? In *Proceedings of the 29th ACM International Conference on Design of Communication (SIGDOC '11)*, pages 241–248. ACM, 2011.
- [159] G. Leshed, E. M. Haber, T. Matthews, and T. Lau. CoScripter: automating & sharing how-to knowledge in the enterprise. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*, pages 1719–1728, Florence, 2008. ACM Press.
- [160] L. Li, Y. Liu, A. Obregon, and M. Weatherston. Visual segmentation-based data record extraction from web documents. In *Proceedings of 2007 IEEE International Conference on Information Reuse and Integration*, pages 502–507, Las Vegas, IL, Aug. 2007. IEEE.
- [161] X. Li and P. A. Ng. A document classification and extraction system with learning ability. In *Proceedings of the Fifth International Conference on Document Analysis and Recognition*, pages 197–200, Bangalore, India, 1999. IEEE Computer Society.
- [162] G. Ligozat. A new proof of tractability for ORD-Horn relations. In *Proceedings of the 13th National Conference on Artificial Intelligence*, volume 1, pages 395–401. AAAI Press and MIT Press, 1996.
- [163] B. Liu. *Web data mining*. Springer, Berlin, 2nd edition, 2008.
- [164] B. Liu, R. Grossman, and Y. Zhai. Mining data records in Web pages. In L. Getoor, T. Senator, P. Domingos, and C. Faloutsos, editors, *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 601 – 606, New York, NY, USA, 2003. ACM.

- [165] B. Liu and Y. Zhai. NET - a system for extracting web data from flat and nested data records. *Web Information Systems Engineering - WISE 2005*, 3806:487–495, 2005.
- [166] W. Liu and X. Meng. Vision-based web data records extraction. In *In Proceedings of the 9th SIGMOD International Workshop on Web and Databases (SIGMOD-WebDB2006)*, page 6, Chicago, Illinois, USA, 2006. ACM.
- [167] W. Liu, X. Meng, and W. Meng. ViDE: A Vision-Based Approach for Deep Web Data Extraction. *IEEE Transactions on Knowledge and Data Engineering*, 22(3):447–460, Mar. 2010.
- [168] D. Lunn, S. Harper, and S. Bechhofer. Combining SADie and AxsJAX to improve the accessibility of web content. In *Proceedings of the 2009 International Cross-Disciplinary Conference on Web Accessibility (W4A)*, pages 75–78, New York, 2009. ACM Press.
- [169] P. Luo, J. Fan, S. Liu, F. Lin, Y. Xiong, and J. Liu. Web article extraction for web printing: a DOM+visual based approach. In *Proceedings of the 9th ACM Symposium on Document Engineering*, pages 66–69, New York, NY, USA, 2009. ACM.
- [170] C. Lutz, F. Baader, E. Franconi, D. Lembo, R. Möller, R. Rosati, U. Sattler, B. Suntisrivaraporn, and S. Tessaris. Reasoning support for ontology design. In B. C. Grau, P. Hitzler, C. Shankey, and E. Wallace, editors, *Proceedings of the Second International Workshop OWL: Experiences and Directions*, pages 1–10, 2006.
- [171] J. U. Mahmud, Y. Borodin, and I. V. Ramakrishnan. Csurf: A context-driven non-visual web-browser. In *Proceedings of the 16th international conference on World Wide Web*, pages 31–40, New York, New York, USA, 2007. ACM Press.
- [172] D. Malerba and M. Ceci. Learning to order: A relational approach. *Mining Complex Data*, 4944:209–223, 2008.
- [173] M. Manna, E. Oro, M. Ruffolo, M. Alviano, and N. Leone. The HiLeX system for semantic information extraction. *Transactions on Large-Scale Data- and Knowledge-Centered Systems (TLDKS)*, 7100:91–125, 2012.
- [174] B. McBride. The Resource Description Framework (RDF) and its Vocabulary Description Language RDFS. In S. Staab and R. Studer, editors, *The Handbook on Ontologies in Information Systems*. Springer, 2003.
- [175] R. R. Mehta, P. Mitra, and H. Karnick. Extracting semantic structure of web documents using content and visual information. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, pages 928–929, Chiba, Japan, 2005. ACM.
- [176] J.-L. Meunier. Optimized XY-cut for determining a page reading order. In *Proceedings of the Eighth International Conference on Document Analysis and Recognition*, volume 1, pages 347–351. IEEE Computer Society, 2005.
- [177] G. Miao, J. Tatemura, W.-P. Hsiung, A. Sawires, and L. E. Moser. Extracting data records from the web using tag path clustering. In *Proceedings of the 18th international conference on World Wide Web - WWW '09*, pages 981–990. ACM, 2009.
- [178] G. Micro. Window-Eyes, version 8.0. <http://www.gwmicro.com/window-eyes/>, 2012.
- [179] R. Mitkov, editor. *The Oxford handbook of computational linguistics*. Oxford University Press, New York, New York, USA, 2005.
- [180] T. Miyoshi and A. Murata. A method to evaluate properness of GUI design based on complexity indexes of size, local density, alignment, and grouping. In *Proceedings of 2001 IEEE International Conference on Systems, Man and Cybernetics*, volume 1, pages 221–226, Tucson, AZ, USA, 2001. IEEE.
- [181] A. Mukerjee and G. Joe. A qualitative model for space. In *In proceedings of the Eight National Conference on Artificial Intelligence (AAAI'90)*, volume 2, pages 721–727, New York, NY, USA, 1990.
- [182] G. Nagy and S. Seth. Hierarchical representation of optically scanned documents. In *In Proceedings of the 7th International Conference on Pattern Recognition*, pages 347–349, Montreal, Canada, 1984. IEEE Computer Society.
- [183] I. Navarrete and G. Sciavicco. Spatial Reasoning with Rectangular Cardinal Direction. In *Proceedings of the ECAI 2006 Workshop on Spatial and Temporal Reasoning*, pages 1–9, 2006.
- [184] B. Nebel and H.-J. Bürckert. Reasoning about temporal relations: a maximal tractable subclass of Allen's interval algebra. *Journal of the ACM (JACM)*, 42(1):43–66, 1995.

- [185] C. Nédellec and A. Nazarenko. Ontology and information extraction: A necessary symbiosis. In P. Buitelaar, P. Cimiano, and B. Magnini, editors, *Ontology Learning from Text: Methods, Evaluation and Applications*, pages 155–170. IOS Press, 2005.
- [186] H. Nguyen, T. Nguyen, and J. Freire. Learning to extract form labels. *Proceedings of the VLDB Endowment*, 1(1):684–694, 2008.
- [187] Z. Nie, J.-R. Wen, and W.-Y. Ma. Webpage understanding: beyond page-level search. *ACM SIGMOD Record*, 37(4):48–54, 2008.
- [188] NV Access. NVDA, version 2012.3. <http://www.nvda-project.org/>, 2012.
- [189] E. Oro and M. Ruffolo. SILA: A spatial instance learning approach for deep web pages. In *Proceedings of the 20th ACM international conference on Information and knowledge management (CIKM '11)*, pages 2329–2332, New York, 2011. ACM.
- [190] E. Oro, M. Ruffolo, and S. Staab. SXPath: Extending XPath towards spatial querying on web documents. *Proceedings of the VLDB Endowment*, 4(2):129–140, 2010.
- [191] F. S. Parreiras, C. Saathoff, T. Walter, T. Franz, and S. Staab. APIs à gogo: Automatic Generation of Ontology APIs. In *2009 IEEE International Conference on Semantic Computing (ICSC '09)*, pages 342–348. Ieee, Sept. 2009.
- [192] J. Pérez, M. Arenas, and C. Gutierrez. Semantics and complexity of SPARQL. *ACM Transactions on Database Systems (TODS)*, 34(3), 2009.
- [193] H. Petrie, F. Hamilton, and N. King. Tension, what tension?: Website accessibility and visual design. In *Proceedings of the 2004 International Cross-Disciplinary Workshop on Web Accessibility (W4A '04)*, pages 13–18. ACM Press, 2004.
- [194] P. Pirolli and S. K. Card. Information foraging. *Psychological Review*, 106(4):1–84, 1999.
- [195] D. Poole and A. Mackworth. *Artificial intelligence: Foundations of computational agents*. Cambridge University Press, New York, 2010.
- [196] A. Popko and A. Kamynin. On Russian GOST of the accessibility of Internet resources for visually impaired people (Russian). *Information Society*, 1(1):73–79, 2010.
- [197] N. Prangnawarat. *Realizing web accessibility based on understanding the visual structure of grocery web portals*. PhD thesis, Vienna University of Technology, 2009.
- [198] Y. Puzis, Y. Borodin, F. Ahmed, and I. V. Ramakrishnan. An intuitive accessible web automation user interface. In *Proceedings of the International Cross-Disciplinary Conference on Web Accessibility (W4A '12)*, page Article No. 41, New York, New York, USA, 2012. ACM Press.
- [199] Y. Puzis, Y. Borodin, R. Puzis, and I. Ramakrishnan. Predictive web automation assistant for people with vision impairments. In *Proceedings of the 22nd international conference on World Wide Web (WWW '13)*, pages 1031–1040. ACM, 2013.
- [200] G. M. A. Rahaman, M. M. Hossain, M. A. Arif, E. Chowdhury, and S. Debnath. Mining structured objects (data records) based on maximum region detection by text content comparison from website. *International Journal of Electrical and Computer Sciences (IJECS-IJENS)*, 10(02):22–28, 2010.
- [201] T. V. Raman. Emacspeak—a speech interface. In *Proceedings of the SIGCHI conference on Human factors in computing systems common ground (CHI '96)*, pages 66–71, New York, 1996. ACM Press.
- [202] T. V. Raman. Emacspeak, version 38.0. <http://emacspeak.sourceforge.net/>, 2013.
- [203] D. A. Randell, Z. Cui, and A. G. Cohn. A Spatial Logic based on Regions and Connection. In B. Nebel, C. Rich, and W. Swartout, editors, *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, pages 165–176, Los Altos, 1992. Morgan Kaufmann.
- [204] J. Renz and B. Nebel. On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the region connection calculus. *Artificial Intelligence*, 108(1-2):69–123, 1999.
- [205] G. Retz-Schmidt. Various views on spatial prepositions. *AI Magazine*, 9(2):95–105, 1988.
- [206] J. Roach. The Rectangle Placement Language. *21st Design Automation Conference Proceedings*, pages 405–411, 1984.
- [207] P. Roth, L. Petrucci, T. Pun, and A. Assimacopoulos. Auditory browser for blind and visually impaired users.

- In *CHI '99 Extended Abstracts on Human Factors in Computing Systems, Pittsburgh, Pennsylvania*, pages 218–219, New York, 1999. ACM.
- [208] Royal National Institute of Blind People. Royal National Institute of Blind People (RNIB). <http://www.rnib.org.uk>. Date accessed: 10.12.2012.
- [209] S. Sarawagi. Information Extraction. *Foundations and Trends in Databases*, 1(3):261–377, 2008.
- [210] Serotek. System Access To Go. <http://www.satogo.com/en/>, 2013.
- [211] D. Shea and M. E. Holzschlag. *The Zen of CSS Design: Visual Enlightenment for the Web*. Peachpit Press, USA, 2005.
- [212] V. I. Shvetsov and M. A. Roschina. *Computer typhlotechnology in social integration of people with deep visual impairments (Russian)*. State University of Nizhny Novgorod, Nizhny Novgorod, 2007.
- [213] V. I. Shvetsov and M. A. Roschina. Accessibility of Internet resources for sightless users as a factor for providing them with access to the open education (Russian). *Open Education*, 1(1):124–128, 2010.
- [214] K. Simon and G. Lausen. ViPER: augmenting automatic information extraction with visual perceptions. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 381–388, Bremen, Germany, 2005. ACM New York, NY, US.
- [215] R. Singh, A. Lahoti, and A. Mukerjee. Interval-algebra based block layout analysis and document template generation. In *Workshop on Document Layout Interpretation and its Applications*, page 4, 1999.
- [216] S. Skiadopoulos and M. Koubarakis. Composing cardinal direction relations. In C. S. Jensen, M. Schneider, B. Seeger, and V. J. Tsotras, editors, *Advances in Spatial and Temporal Databases*, pages 299–317. Springer, 2001.
- [217] T. R. Smith and K. K. Park. Algebraic approach to spatial reasoning. *International Journal of Geographical Information Systems*, 6(3):177–192, 1992.
- [218] S. Soderland, D. Fisher, J. Aseltine, and W. Lehnert. Crystal: Inducing a conceptual dictionary. In C. Mellish, editor, *Proceedings of the 14th international joint conference on Artificial intelligence (IJCAI'95)*, pages 1314–1319, San Francisco, 1995. Morgan Kaufmann.
- [219] A. Spengler and P. Gallinari. Document structure meets page layout: loopy random fields for web news content extraction. In *Proceedings of the 10th ACM symposium on Document engineering (DocEng '10)*, pages 151–160, Manchester, United Kingdom, 2010. ACM: New York, NY, USA.
- [220] S. Staab and R. Studer, editors. *Handbook on Ontologies*. Springer, Berlin, Germany, 2004.
- [221] S. Staab, T. Walter, G. Gröner, and F. S. Parreiras. Model driven engineering with ontology technologies. *Reasoning Web. Semantic Technologies for Software Engineering*, 6325:62–98, 2010.
- [222] A. Stent, I. V. Ramakrishnan, and G. Yang. Hearsay: enabling audio browsing on hypertext content. In *Proceedings of the 13th international conference on World Wide Web (WWW'2004)*, pages 80–89. ACM: New York, NY, USA, 2004.
- [223] C. Stephanidis. Adaptive techniques for universal access. *User Modeling and User-Adapted Interaction*, 11(1-2):159–179, 2001.
- [224] R. Studer, V. R. Benjamins, and D. Fensel. Knowledge engineering: Principles and methods. *Data & Knowledge Engineering*, 25(1–2):161–197, 1998.
- [225] H. Takagi, C. Asakawa, K. Fukuda, and J. Maeda. Accessibility designer: visualizing usability for the blind. In *Proceedings of the 6th international ACM SIGACCESS conference on Computers and accessibility (Assets '04)*, pages 177–184, New York, 2004. ACM.
- [226] H. Takagi, S. Saito, K. Fukuda, and C. Asakawa. Analysis of navigability of Web applications for improving blind usability. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 14(3):37, Sept. 2007.
- [227] Y. Y. Tang, M. Cheriet, J. Liu, J. N. Said, and C. Y. Suen. Document analysis and recognition by computers. In C. H. Chen, L. F. Pau, and P. S. P. Wang, editors, *Handbook of Pattern Recognition and Computer Vision*, pages 579–612. World Scientific Publishing Company, Singapore, Republic of Singapore, 2 sub edit edition, 1999.
- [228] Y. Y. Tang, S.-W. Lee, and C. Y. Suen. Automatic document processing: A survey. *Pattern Recognition*, 29(12):1931–1952, Dec. 1996.
- [229] S. Tessaris, E. Franconi, T. Eiter, C. Gutierrez, S. Handschuh, M.-C. Rousset, and R. A. Schmidt, editors.

Reasoning Web. Semantic technologies for information systems. Springer, Berlin, 2009.

- [230] J. Thatcher, P. Bohman, M. Burks, S. L. Henry, B. Regan, S. Swierenga, M. D. Urban, and C. D. Waddell. *Constructing accessible web sites.* Glasshaus, 2002.
- [231] The Apache Software Foundation. Jena Ontology API. <http://jena.apache.org/documentation/ontology/>. Date accessed: 15.12.2012, 2013.
- [232] The Apache Software Foundation. Reasoners and rule engines: Jena inference support. <http://jena.apache.org/documentation/inference/>. Date accessed: 15.12.2012, 2013.
- [233] The United States Government. Section 508: The Road to Accessibility. Rehabilitation Act. <http://www.section508.gov>. Date accessed: 05.03.2012, 1998.
- [234] M. F. Theofanos and J. Redish. Bridging the gap: Between accessibility and usability. *Interactions*, 10(6):36–51, 2003.
- [235] Y.-F. Tseng and H.-Y. Kao. The mining and extraction of primary informative blocks and data objects from systematic web pages. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 370–373. IEEE Computer Society, 2006.
- [236] TU Wien. ABBA — Advanced barrier-free browser accessibility. FFG Fit-IT Project 819563, 2009–2010. <http://www.dbai.tuwien.ac.at/proj/ABBA/>.
- [237] TU Wien. TAMCROW — Task mining and crowd sourcing. FFG Fit-IT Project 829614, 2011–2012. <http://www.dbai.tuwien.ac.at/proj/tamcrow/>.
- [238] W3C. Web Accessibility Initiative (WAI). <http://www.w3.org/WAI/>. Date accessed: 10.12.2012.
- [239] W3C. World Wide Web Consortium (W3C). <http://www.w3.org/>. Date accessed: 10.12.2012.
- [240] W3C. Document Object Model (DOM) Level 1 Specification (Version 1.0). W3C Recommendation 1 October 1998. Recommendation, W3C, 1998.
- [241] W3C. HTML 4.01 Specification. W3C Recommendation 24 December 1999. Recommendation, W3C, 1999.
- [242] W3C. Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation 22 February 1999. Recommendation, W3C, 1999.
- [243] W3C. XML path language (XPath) version 1.0. W3C Recommendation 16 November 1999. Recommendation, W3C, 1999.
- [244] W3C. Document Object Model (DOM) Level 2 Traversal and Range Specification (Version 1.0). W3C Recommendation 13 November 2000. Recommendation, W3C, 2000.
- [245] W3C. CSS3 Module: Line. W3C Working Draft 15 May 2002. Working draft, W3C, 2002.
- [246] W3C. XHTML 1.0 The Extensible HyperText Markup Language (Second Edition). A Reformulation of HTML 4 in XML 1.0. W3C Recommendation 26 January 2000, revised 1 August 2002. Recommendation, W3C, 2002.
- [247] W3C. Document Object Model (DOM) Technical Reports. <http://www.w3.org/DOM/DOMTR>. Date accessed: 05.03.2012, 2004.
- [248] W3C. OWL Web Ontology Language Guide. W3C Recommendation 10 February 2004. Recommendation, W3C, 2004.
- [249] W3C. OWL Web Ontology Language Overview. W3C Recommendation 10 February 2004. Recommendation, W3C, 2004.
- [250] W3C. RDF Primer. W3C Recommendation 10 February 2004. Recommendation, W3C, 2004.
- [251] W3C. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation 10 February 2004. Recommendation, W3C, 2004.
- [252] W3C. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. W3C Member Submission 21 May 2004. Member submission, W3C, 2004.
- [253] W3C. Voice Extensible Markup Language (VoiceXML) Version 2.0. W3C Recommendation 16 March 2004. Recommendation, W3C, 2004.
- [254] W3C. CSS Basic Box Model. W3C Working Draft 9 August 2007. Working draft, W3C, 2007.
- [255] W3C. CSS Grid Positioning Module Level 3. W3C Working Draft 5 September 2007. Working draft, W3C, 2007.

- [256] W3C. Extensible Markup Language (XML) 1.0 (Fifth Edition). W3C Recommendation 26 November 2008. Recommendation, W3C, 2008.
- [257] W3C. SPARQL Query Language for RDF. W3C Recommendation 15 January 2008. Recommendation, W3C, 2008.
- [258] W3C. Web Content Accessibility Guidelines (WCAG) 2.0. W3C Recommendation 11 December 2008. Recommendation, W3C, 2008.
- [259] W3C. Accessible Rich Internet Applications (WAI-ARIA) 1.0. W3C Candidate Recommendation 18 January 2011. Candidate recommendation, W3C, 2011.
- [260] W3C. Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification. W3C Recommendation 07 June 2011. Recommendation, W3C, 2011.
- [261] W3C. CSS Lists and Counters Module Level 3. W3C Working Draft 24 May 2011. Working draft, W3C, 2011.
- [262] W3C. CSS Multi-Column Layout Module. W3C Candidate Recommendation 12 April 2011. Candidate recommendation, W3C, 2011.
- [263] W3C. CSS Template Layout Module. W3C Working Draft 29 November 2011. Working draft, W3C, 2011.
- [264] W3C. CSSOM View Module. W3C Working Draft 4 August 2011. Working draft, W3C, 2011.
- [265] W3C. Authoring Tool Accessibility Guidelines (ATAG) 2.0. W3C Working Draft 10 April 2012. Working draft, W3C, 2012.
- [266] W3C. CSS Backgrounds and Borders Module Level 3. W3C Candidate Recommendation 24 July 2012. Candidate recommendation, W3C, 2012.
- [267] W3C. CSS Basic User Interface Module Level 3 (CSS3 UI). W3C Working Draft 17 January 2012. Working draft, W3C, 2012.
- [268] W3C. CSS Box Alignment Module Level 3. W3C Working Draft 12 June 2012. Working draft, W3C, 2012.
- [269] W3C. CSS Exclusions and Shapes Module Level 3. W3C Working Draft 3 May 2012. Working draft, W3C, 2012.
- [270] W3C. CSS Flexible Box Layout Module. W3C Candidate Recommendation, 18 September 2012. Candidate recommendation, W3C, 2012.
- [271] W3C. CSS Regions Module Level 3. W3C Working Draft 23 August 2012. Working draft, W3C, 2012.
- [272] W3C. CSS Speech Module. W3C Candidate Recommendation 20 March 2012. Candidate recommendation, W3C, 2012.
- [273] W3C. CSS Transforms. W3C Working Draft 11 September 2012. Working draft, W3C, 2012.
- [274] W3C. CSS Transitions. W3C Working Draft 3 April 2012. Working draft, W3C, 2012.
- [275] W3C. CSS Values and Units Module Level 3. W3C Candidate Recommendation 28 August 2012. Candidate recommendation, W3C, 2012.
- [276] W3C. Media Queries. W3C Recommendation 19 June 2012. Recommendation, W3C, 2012.
- [277] W3C. OWL 2 Web Ontology Language Document Overview (Second Edition). W3C Recommendation 11 December 2012. Recommendation, W3C, 2012.
- [278] W3C. OWL 2 Web Ontology Language New Features and Rationale (Second Edition). W3C Recommendation 11 December 2012. Recommendation, W3C, 2012.
- [279] W3C. OWL 2 Web Ontology Language Primer (Second Edition). W3C Recommendation 11 December 2012. Recommendation, W3C, 2012.
- [280] W3C. OWL 2 Web Ontology Language Profiles (Second Edition). W3C Recommendation 11 December 2012. Recommendation, W3C, 2012.
- [281] W3C. OWL 2 Web Ontology Language RDF-Based Semantics (Second Edition). W3C Recommendation 11 December 2012. Technical report, W3C, 2012.
- [282] W3C. SPARQL 1.1 Query Language. W3C Proposed Recommendation 08 November 2012. Proposed recommendation, W3C, 2012.
- [283] W3C. The Rule Markup Initiative, <http://ruleml.org/>. (Accessed April 13, 2013), 2012.
- [284] W3C. User Agent Accessibility Guidelines (UAAG) 2.0. W3C Working Draft 23 May 2013. Working draft,

- W3C, 2012.
- [285] W3C. W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes. W3C Recommendation 5 April 2012. Recommendation, W3C, 2012.
- [286] W3C. WebDriver. W3C Working Draft 10 July 2012. Working draft, W3C, 2012.
- [287] W3C. RIF Basic Logic Dialect (Second Edition). W3C Recommendation 5 February 2013. Recommendation, W3C, 2013.
- [288] H. Walischewski. Automatic knowledge acquisition for spatial document interpretation. In *Proceedings of the Fourth International Conference on Document Analysis and Recognition (ICDAR'97)*, pages 243–247, Washington, DC, USA, 1997. IEEE Computer Society.
- [289] H. Walischewski. Learning and interpretation of the layout of structured documents. In G. Brewka, C. Habel, and B. Nebel, editors, *KI-97: Advances in Artificial Intelligence. 21st Annual German Conference on Artificial Intelligence*, volume 1303, pages 409–412, Freiburg, Germany, 1997. Springer.
- [290] J. Wang and F. H. Lochovsky. Data extraction and label assignment for web databases. In G. Hencsey, B. White, Y.-F. R. Chen, L. Kovács, and S. Lawrence, editors, *Proceedings of the 12th international conference on World Wide Web*, pages 187–196, New York, 2003. ACM.
- [291] C. Ware. *Information visualization: perception for design*. Morgan Kaufmann Publishers Inc., San Francisco, 2nd editio edition, 2004.
- [292] T. Watanabe. Experimental evaluation of usability and accessibility of heading elements. *Disability and Rehabilitation. Assistive Technology*, 4(4):236–247, 2009.
- [293] Web Accessibility in Mind (WebAIM). Screen Reader User Survey #2 Results (<http://webaim.org/projects/screenreadersurvey2/>), 2009.
- [294] Web Accessibility in Mind (WebAIM). Survey of Preferences of Screen Readers Users (<http://webaim.org/projects/screenreadersurvey/>), 2009.
- [295] Web Accessibility in Mind (WebAIM). Screen Reader User Survey #3 Results (<http://webaim.org/projects/screenreadersurvey3/>), 2010.
- [296] Web Accessibility in Mind (WebAIM). Screen Reader User Survey #4 Results (<http://webaim.org/projects/screenreadersurvey4/>), 2012.
- [297] M. Weisen, N. King, and H. Petrie. The accessibility of museum web sites: Results from an English Investigation and international comparisons. In J. Trant and D. Bearman, editors, *Proceedings of Museums and the Web 2005*, page 9, Vancouver, British Columbia, Canada, 2005. Archives & Museum Informatics.
- [298] B. White. The Implications of Web 2.0 on Web Information Systems. *Web Information Systems and Technologies*, 1:3–7, 2007.
- [299] P. Xiang, X. Yang, and Y. Shi. Web page segmentation based on Gestalt theory. In *IEEE International Conference on Multimedia and Expo*, pages 2253–2256. IEEE, July 2007.
- [300] Y. Yesilada. Web page segmentation: A review. Technical Report March, University of Manchester and Middle East Technical University Northern Cyprus Campus, Nicosia, Cyprus, 2011.
- [301] Y. Yesilada, S. Harper, and S. Eraslan. Experiential transcoding: an EyeTracking approach. In *Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility*, page 4. ACM, 2013.
- [302] Y. Yesilada, S. Harper, C. Goble, and R. Stevens. DANTE: annotation and transformation of web pages for visually impaired users. In S. Feldman and M. Uretsky, editors, *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 490–491. ACM, 2004.
- [303] M. Zajicek, C. Powell, C. Reeves, and J. Griffiths. Web browsing for the visually impaired. In *Computers and assistive technology ICCHP '98 Vienna, Budapest, 31 August - 4 September 1998*), pages 161–169, Vienna, 1998. OCG.
- [304] Y. Zhai and B. Liu. Web data extraction based on partial tree alignment. In *Proceedings of the 14th international conference on World Wide Web (WWW '05)*, pages 76–85, Japan, 2005. ACM Press.
- [305] Z. Zhang, B. He, and K. C.-C. Chang. Understanding web query interfaces: best-effort parsing with hidden syntax. In A. C. König, S. Dessoloch, P. Valduriez, and G. Weikum, editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 107–118, Paris, France, 2004. ACM.

- [306] H. Zhao, W. Meng, Z. Wu, V. Raghavan, and C. Yu. Fully automatic wrapper generation for search engines. In *Proceedings of the 14th international conference on World Wide Web - WWW '05*, page 66, New York, New York, USA, 2005. ACM Press.
- [307] H. Zhao, W. Meng, and C. Yu. Automatic extraction of dynamic record sections from search engine result pages. In *Proceedings of the 32nd international conference on Very large data bases (VLDB '06)*, pages 989–1000. VLDB Endowment, 2006.
- [308] P. Zhong and J. Chen. A generalized hidden Markov model approach for web information extraction. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 709–718. IEEE Computer Society, Dec. 2006.
- [309] J. Zhu, Z. Nie, J.-R. Wen, B. Zhang, and W.-Y. Ma. 2D conditional random fields for web information extraction. In *Proceedings of the 22nd International Conference on Machine learning (ICML '05)*, pages 1044–1051, Bonn, Germany, 2005. ACM Press.
- [310] J. Zhu, Z. Nie, J.-R. Wen, B. Zhang, and W.-Y. Ma. Simultaneous record detection and attribute labeling in web data extraction. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '06)*, volume 1, pages 494–503, Philadelphia, PA, USA, 2006. ACM.
- [311] J. Zou, D. Le, and G. R. Thoma. Combining DOM tree and geometric layout analysis for online medical journal article segmentation. In *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries (JCDL '06)*, page 119, Chapel Hill, NC, USA, 2006. ACM Press.

Acronyms

2DIR	Two-Dimensional Interval Relations
2DIRC	Two-Dimensional Interval Relations with Centering
API	application programming interface
ATAG	Authoring Tool Accessibility Guidelines
BGM	Block-based Geometric Model
BOM	Browser Object Model
CAD	Computer-Aided Design
CSS	Cascading Style Sheets
CSSOM	CSS Object Model
CWA	closed world assumption
DL	Description Logics
DLP	Description Logic Programs
DOM	Document Object Model
DU	Document Understanding
GIS	Geographical Information Systems
GUI	graphical user interface
IE	Information Extraction
JEPD	jointly exhaustive and pairwise disjoint
LM	Logical Model
MANM	Multi-Axial Navigation Model

NLP	Natural Language Processing
OWA	open world assumption
OWL	Web Ontology Language
PM	Physical Model
QltBGM	Qualitative Block-based Geometric Model
QntBGM	Quantitative Block-based Geometric Model
RCC	Region Connection Calculus
RCR	Rectangular Cardinal Relations
RDF	Resource Description Framework
RDFS	RDF Schema
RIF	Rule Interchange Format
RuleML	Rule Markup Language
SGG	Spatial Graph Grammar
StrBGM	Structural Block-based Geometric Model
SVG	Scalable Vector Graphics
SWRL	Semantic Web Rules Language
TBRR	Thick Boundary Rectangle Relations
UAAG	User Agent Accessibility Guidelines
UNA	unique name assumption
UOM	Unified Ontological Model
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
WAI	Web Accessibility Initiative
WAI-ARIA	Accessible Rich Internet Applications
WCAG	Web Content Accessibility Guidelines
WDE	Web Data Extraction
WIE	Web Information Extraction
WPP	Web Page Processing
WPPS	Web Page Processing System
WPU	Web Page Understanding

