Vrije Universiteit Brussel

FACULTY OF SCIENCE AND BIO-ENGINEERING SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

Enhanced Visualisation of Web Links and Annotations

Graduation thesis submitted in partial fulfillment of the requirements for the degree of Master in Applied Computer Science

David Swalus

Promoter: Prof. Dr. Beat Signer
Advisor: Ahmed Tayeh

Academic year 2013-2014

Vrije Universiteit Brussel

FACULTEIT WETENSCHAPPEN EN BIO-INGENIEURSWETENSCHAPPEN
VAKGROEP COMPUTERWETENSCHAPPEN

Enhanced Visualisation of Web Links and Annotations

Afstudeer eindwerk ingediend in gedeeltelijke vervulling van de eisen
voor het behalen van de graad
Master in de Toegepaste Informatica

David Swalus

Promoter: Prof. Dr. Beat Signer
Advisor: Ahmed Tayeh

Academiejaar 2013-2014

Abstract

The World Wide Web is playing an increasingly important role in day to day life. From personal use to big Internet companies, the Web is omnipresent in our modern society. The Web was created in 1989 and publicly available since 1991. Before Web 2.0, webpages were being linked using simple HTML anchor tags. Moreover, web users were passive users. With the introduction of Web 2.0 and the use of some standards such as XLink standard, web users are not passive users any more. They are able to enrich webpages with extra external hyperlinks and annotations. Besides that, with the development of Semantic Web technologies such as RDF and RDFa, web users and developers are able to enrich webpages and their hyperlinks with meaningful machine-readable metadata. Thereby, the Web is very rich of informative hyperlinks and annotations. However, it is rather strange that web browsers have not changed the hyperlinks visualisation since the beginning of the Web. Moreover, metadata attached to hyperlinks and annotations are not exploited to enhance the delivery and visualisation of web content.

Through literature study and critical thinking, we identified a number of important flaws and shortcomings regarding the conventional hyperlinks and annotations visualisation on the Web. A review of some user studies about hyperlink visualisation in addition to a closer look at alternative visualisation approaches and techniques showed that the visualisation of web hyperlinks and annotations can definitely be enhanced. In this thesis we propose the Wysinwis principle, which stands for What You See Is Not What I See. The Wysinwis is built around the web users and their different preferences and surfing attitudes. The Wysinwis principle uses different visualisation techniques for hyperlinks and annotations as well as their attached metadata. As a proof of concept, a prototype has been implemented. Furthermore, to evaluate the concept of the Wysinwis principle and its prototype, a user study has been conducted.

Acknowledgements

First of all I would like to thank my promotor and advisor Ahmed Tayeh for helping me to create this thesis. I am very grateful to him for spending countless days and nights reading and improving my work. Thank you for your guidance and support throughout this long journey.

Special thanks goes out to Prof. Dr. Beat Signer for reviewing my thesis and providing me with his feedback.

I would also like to thank my girlfriend, and soon to be wife, Stéphanie for constantly reading all my drafts. And thank you for looking after our two beautiful children when I was not able to. And mum and dad, I really appreciate you babysitting our children when we were both preoccupied with work.

Thank you my children, Amélie and Quinten, for putting a smile on my face when I was ready to cry.

And last but not least I would like to offer a word of gratitude to all my friends, family, co-workers and acquaintances who have participated in the questionnaire written in my thesis.

Contents

1 Introduction

1.1 Context

The World Wide Web is a hypertext system of interlinked documents and data via the Internet. The World Wide Web is omnipresent in our daily life. It was created at the end of the 80s and has been publicly available since 1991. The Web documents are created with the Hypertext Markup Language (HTML). Already since the beginning of HTML, web developers were able to add hyperlinks to their webpages using the `<a>` tag and `href` attribute. These hyperlinks allow webpages to be interlinked. Later, the HTML standard has gone through several iterations. We are currently at HTML5 and many features have been changed and added. Metadata can be attached to hyperlinks to describe the relations between the interlinked documents. Moreover, typed hyperlinks are supported in the last HTML standards. Furthermore, webpages and their hyperlinks can also be enriched with RDFa [42] metadata.

With the advent of the Extensible Markup Language (XML) and its link model (XLink), advanced external hyperlinks can be created for webpages. Hence, besides the web developers, normal users are able to create external hyperlinks and annotations for webpages via third-party annotation tools. Therefore, the Web is very rich in metadata and informative hyperlinks and annotations. However, it is rather strange that web browsers did not change

hyperlink visualisation since the beginning of the World Wide Web. They have neglected the rich metadata attached to hyperlinks and they still use the blue underlined text to visualise hyperlinks.

With 2.5[1] billion users on the World Wide Web, each of them with different preferences and attitudes for surfing the Web, we believe that the visualisation of hyperlinks/annotations and their metadata has to be changed to meet all users needs. To tackle the hyperlinks/annotations visualisation shortcomings on the Web, this thesis reviews the state of the art of the visualisation of hyperlinks/annotations on the Web. Then some ideas are proposed for a better visualisation with a proof of concept implementation of the proposed ideas. To check the validity of the proposed solution, an evaluation has been conducted.

1.2 Problem Statement

Web users have a myriad number of characteristics, different needs and different interests. Even though the Web evolved with its standards, web browsers still treat them in the same way. The metadata provided by web developers and normal users is very useful for web users. When the metadata would be visualised, they can filter (parts of) webpages based on this metadata or they can choose to read some webpages or not. However, almost none of the web browsers visualise this metadata or even give a hint to users of its existence.

Another visualisation issue, is the underlined blue visualisation of all hyperlinks. Even though some studies have proven that this visualisation is not suitable for reading webpages content [50, 31, 27], it is still the standard to visualise hyperlinks. The only information browsers give is that there are some parts of the page that are clickable. Important information about the hyperlink such as its type, its target format (e.g. PDF, HTML, Word, anchor of a webpage or image), the protocol that will be used for hyperlink resolution or availability of the target is not visualised.

Last but not least, user-defined annotations are visualised as overlays on webpages with very limited support for filtering and searching through these annotations. Web browsers are not aware of web users' annotations. Indeed, third-party tools have some filtering mechanisms, but a problem arises when there are a lot of annotations on a webpage, because the webpage becomes cluttered and readability issues occur.

[1] http://www.internetworldstats.com/stats.htm

1.3 Research Objectives

The main aim of this thesis is to identify essential aspects that should be considered for a better visualisation of hyperlink/annotations and their attached metadata on the Web. This thesis seeks for a solution that meets the requirements of web users and their diverse preferences and purposes for surfing the Web. Then, the proposed ideas for a better visualisation will be validated by a proof of concept prototype implementation. Moreover, the prototype is evaluated by real users.

1.4 Research Methods

To achieve the aim of this research, the following steps were carried out:

- Review of hyperlink and annotation visualisation on the Web: An in-depth analysis of visualisation mechanisms that are used in the web browsers and third-party annotation tools. Moreover, previous work on hyperlink visualisation has been taken into account.
- Identification of essential aspects for a better visualisation: We propose some ideas for a better visualisation of hyperlink/annotations and their attached metadata on the Web. This solution combines many different visualisation techniques. Moreover, the solution is built around user needs and their user profiles.
- Implementation: As a proof of concept a prototype, has been realised as a Google Chrome Extension. This prototype provides all the necessary functionality for an evaluation by real users.
- An evaluation of the prototype and the proposed visualisation solution: A quantitative and qualitative evaluation of the prototype and the proposed solution has been carried out.

1.5 Thesis Structure

The structure of this thesis is presented as follows:

Chapter 2: Background
The purpose of this chapter is to set the fundamental backgound for some concepts that will be addressed in this thesis.

Chapter 3: Literature Review
A review of hyperlink and annotation visualisation techniques are presented in this chapter. Moreover, some user studies for hyperlink visualisation are discussed.

Chapter 4: Wysinwis
In this chapter, we propose a better visualisation technique for hyperlinks and annotations on the web.

Chapter 5: Prototype
This chapter provides an overvieuw of the implementation of the Wysinwis prototype.

Chapter 6: Evaluation
This chapter presents the user study that has been conducted to evaluate the Wysinwis principle and prototype.

Chapter 7: Conclusions and Future Work
In the final Chapter, we conclude this thesis and highlight some potential future work.

2 Background

In this chapter we introduce some concepts that are essential for our research. We start by giving a historical overview of the Web, we briefly introduce the Semantic Web and end with a short description of web annotations.

2.1 History of the Web

In his famous essay "As we may Think" [8] from 1945, Vannevar Bush described an electromechanical device that can be seen as a prototype for the hypertext systems that we know today. In this essay he presented a hypothetical machine called Memex, in which you can store books and documents in the form of microfilms. Memex stands for Memory Extender or the Portmanteau for Memory and Index. An illustration of the Memex, which looks like a desktop, is depicted in Figure 2.1. To explain the different parts of the Memex we added annotations to the most important parts. In his vision, the user could create trails of links and annotations between documents stored on different microfilms. For this purpose, the desktop consists of two displays and the buttons on the right can be used to save and retrieve the trails.

Influenced by the idea of Bush, Tim Berners-Lee proposed a project in 1989 that we now know as the World Wide Web. The original text of his proposal can be found online [4]. At that time he was working for the European

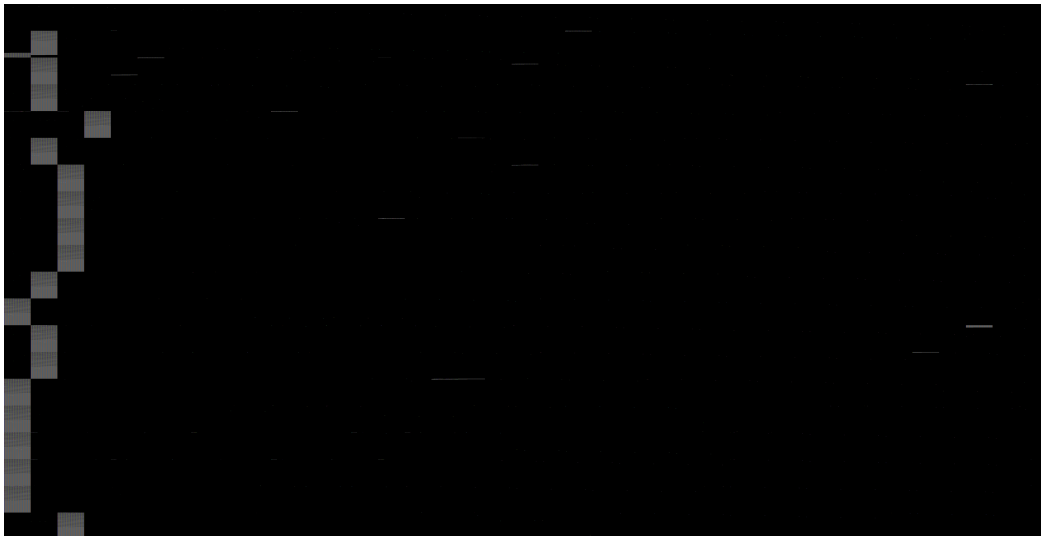

Figure 2.1: The Memex

Organisation for Nuclear Research[1] (CERN) and developed his idea along with Robert Cailliau. Although the HTTP protocol and HTML language were developed by Berners-Lee and Cailliau in 1990, Theodore Nelson had already coined the terms 'hypertext' and 'hypermedia' in 1963.

The main characteristic of hypertext is that it is text which is not linear. Therefore, text can contain links to other text. The term hypermedia is the generalisation of the hypertext concept, which also allows other types of media than text (i.e. video, sound, graphics). In hypermedia, pages are interconnected to each other by hyperlinks. Because hypermedia is the underlying concept for the World Wide Web, hyperlinks form an important part of the current Web. In the next section we will discuss how we can describe documents on the World Wide Web and how they have implemented the hyperlinks.

2.1.1 HTML

Berners-Lee and Cailliau developed a markup language called the HyperText Markup Language (HTML) to describe documents on the World Wide Web. The article "HTML Tags" [5] was the first publicly available description of HTML. The HTML language was based on the Standard Generalised Markup Language (SGML) [40]. Initially, there were only a few tags available in HTML, but since the very beginning there was a support for the creation

[1] Homepage of CERN: `http://home.web.cern.ch/`

of links; otherwise it would not be a markup language to build hypermedia-documents. The syntax of the language looks like the code in Listing 2.1.

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>Hello World</title>
    </head>
    <body>
        <h1>Hello World</h1>
        <p>
            A simple paragraph in HTML.
        </p>
    </body>
</html>
```

Listing 2.1: Hello World example in HTML

The simplest way to create hyperlinks in HTML is by using the anchor (`<a>`) tag together with the `href` attribute, see Listing 2.2.

```
<a href=destination-URI>This is a link</a>
```

Listing 2.2: A Hyperlink in HTML

After HTML's initial version, updates were made to it to add other tags and to review some of the existing ones. Originally there was no official party that maintained HTML. At that time browser developers (i.e. Microsoft or Netscape) implemented a version of HTML, but the meaning of the specified tags was interpreted differently by different browser vendors. They even added extra tags to HTML, which are only supported by their own browsers. Often this issue is referenced as the First Browser War [48, 3]. Because of the Browser War, authors of webpages were obliged to choose for a specific browser to support. In 1994, Tim Berners-Lee founded a consortium, called the World Wide Web Consortium[2] (W3C), to prevent the lack of compatibility between web browsers. The goal of the W3C was and is to push through a set of core principles and components to all the vendors.

At the moment, the last version of HTML is HTML5, which is the first version that is not based on SGML, but on XML (Extensible Markup Language) [40]. This version extends HTML with other features, most of them related to multimedia content (i.e. SVG, MathML or video). Next to these additional tags, more strict parsing rules were put in place. The HTML5 specification also depicts some new APIs [41] (i.e. Drag and Drop, client-side storage). The strict rules help developers to debug their website and the most important benefit is that an HTML document becomes a valid restricted tree.

[2]Home page World Wide Web Consortium: `http://www.w3.org/`

To format a webpage, there exists an HTML `style` attribute or we can use Cascading Style Sheets (CSS). With both techniques, it is possible to change the rendering of a webpage and therefore also the visualisation of hyperlinks. In the literature, several ideas for the visualisation of hyperlinks are proposed. The best known way by users to distinguish a link from a normal text on a webpage is the underlined blue text for the visualisation of hyperlinks. An example of this kind of hyperlink visualisation is www.google.be.

We will discuss more techniques for the visualisation of hyperlinks together with their benefits and drawbacks later in Chapter 3.

Thanks to the improvements made to HTML along with the development of web-related APIs, the Web itself has also evolved. In the next section we discuss the evolution of the World Wide Web to a Semantic Web. In the last section of this chapter, we discuss the concept of annotations. We will see that hyperlinks are in fact a subtype of the much broader concept of annotations.

2.2 The Semantic Web

When the World Wide Web started, webpages were documents with data maintained only by the webmaster. It was a Web of documents linked together by the use of hyperlinks. Since 2004, the term Web 2.0 became popular. It was not a new version of the Web, but a term to indicate the Social Web. Users of the Web are no longer passive viewers, but content creators. They use social network sites, create blogs or use web applications.

The current version of the Web is Web 3.0, also called the Semantic Web. The idea is to create a 'Web of data' instead of a 'Web of documents'. The goal is that data that has a meaning for humans can also be processed by computers. Therefore, data must be described by adding machine-readable metadata. Computers could then use all the metadata to derive new facts, or to detect contradictions. The architecture of the Semantic Web is often represented by a layered diagramme called the Semantic Web Stack (see Figure 2.2). In the next subsections, we introduce the RDF, RDFs and RDFa standards that are the most essential standards to enrich the Web with semantics.

2.2.1 RDF

The Resource Description Framework [43] (RDF) is a standard from the World Wide Web Consortium (W3C) to describe metadata about resources on the World Wide Web. The goal of RDF is to enrich web data with

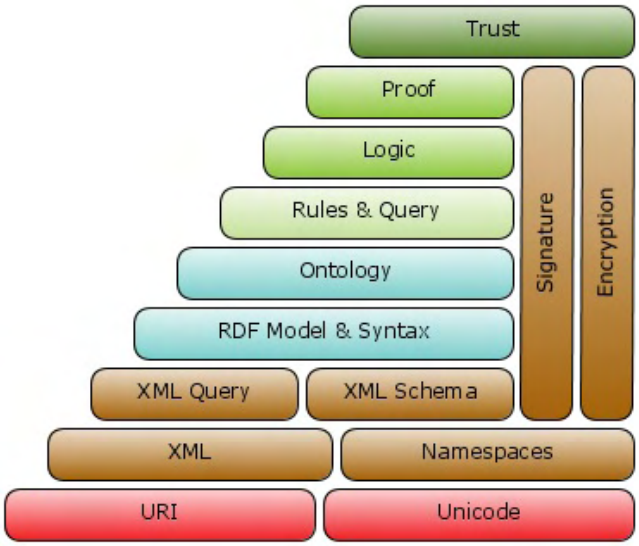

Figure 2.2: The Semantic Web Stack [45]

metadata that is machine readable. RDF can be found on the Semantic Web Stack on the RDF Model & Syntax level (see Figure 2.2). An RDF statement is a triple that has a subject, a predicate and an object. The predicate is the relation between the subject and the object. The object can be a URI, but it can also be a (typed) literal. RDF triples can be expressed in different notations (e.g. N3, Turtle, N-triples, RDF/XML). An example of an RDF triple in RDF/XML notation is shown in Listing 2.3.

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description rdf:about="This thesis">
    <dc:title>Visualisation of Web Annotations</dc:title>
    <dc:title xml:lang="en">Visualisation of Web Annotations</dc:title>
    <dc:title xml:lang="nl-BE">Weergave van Annotaties op het Web</dc:title>
  </rdf:Description>
</rdf:RDF>
```

Listing 2.3: An example of an RDF triple in RDF/XML notation

2.2.2 RDFa

RDFa [42], which stands for Resource Description Framework in Attributes, is a W3C Recommendation that adds more expressive features to HTML and XHTML (Extensible HyperText Markup Language). In an article from IBM

(International Business Machines Corporation)[3] they state, "RDFa makes your HTML pages smarter" [32]. In other words, authors of webpages can add semantics to their webpages by using the RDFa attributes. An example is depicted in Listing 2.4

```
<div xmlns:dc="http://purl.org/dc/elements/1.1/"
  about="This Thesis">
  <span property="dc:title">Visualisation of Web Annotations</span>
  <span property="dc:creator">David Swalus</span>
  <span property="dc:date">2014-01-01</span>
</div>
```

Listing 2.4: An example of the use of RDFa

2.2.3 RDFs

A Resource Description Framework Schema [44] (RDFs) is a way to describe an ontology (see the ontology layer on Figure 2.2). With RDFs, we can construct taxonomies by creating classes with properties. A well known ontology is the Dublin Core[4] which we already used in the RDFa-example (see Listing 2.4). The Dublin Core is a schema to describe semantic data for web and physical resources (e.g. video, images, web pages and books). Another example is the Friend of a Friend (FOAF) ontology which is intended to describe persons. An excerpt from the FOAF Schema definition is shown in Listing 2.5.

```
<rdfs:Class
 rdf:about="http://xmlns.com/foaf/0.1/Person"
 rdfs:label="person"
 rdfs:comment="a foaf subclass of wordnet person">

  <rdfs:subClassOf rdf:resource="http://xmlns.com/wordnet/1.6/Person" />
  <rdfs:isDefinedBy rdf:resource="http://xmlns.com/foaf/0.1/" />

 </rdfs:Class>
```

Listing 2.5: An example of an RDF Schema

2.3 Annotations

2.3.1 Overview

An annotation can be seen as metadata attached to data (e.g. text, image). When we talk about the history of annotations, we have to dig deep into

[3] IBM homepage: `http://www.ibm.com/`
[4] Dublin Core Metadata Initiative Homepage: `http://dublincore.org/`

the past. Even in old books we can find paragraphs that were annotated (see Figure 2.3). People were annotating in multiple ways for example by underlining some words, drawing some sketches or adding some text.

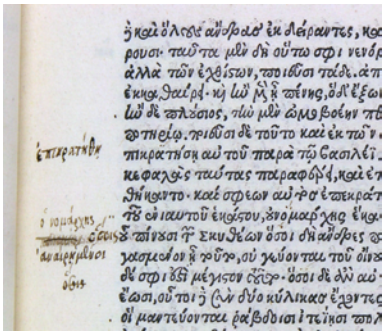

Figure 2.3: Margin of a book printed in Greek, annotated during the 1530s

Annotations are used in a wide range of domains (i.e. education, software development, law). In the following section we take a closer look at the different approaches for annotating web resources.

2.3.2 Annotations on the Web

In this section we discuss multiple ways to annotate the Web. The best known way to annotate a web resource is a hyperlink. In our solution we see hyperlinks as a subtype of annotations. This is because a hyperlink has all the properties of annotations, but they have to be complemented with at least a destination.

Marshall et al. [29] described three types of annotations. In the following, we give an overview of these types.

Types of annotations

Anchor only

Sometimes annotations are just used to emphasise a fragment in a text by adding markers to the fragments. On paper documents we often see circled or underlined sentences. On the Web underlined fragments and highlights are the most frequently used anchor markers. Anchor-only annotation does not contain a link to other pieces of the text.

Content only

Another type of annotations is the one that has a piece of text attached to the annotated text, but there is no link to a certain fragment in the text.

Composite

An annotation with content, that is linked to a certain part of the document, is called a composite link.

On the Web we find annotations in different ways, as explained in the following.

Social Media and Discussion Boards

A lot of people say that they do not use annotations on the Web. But they are not aware of the fact that when they post their comment on others' posts, by using Twitter or Facebook, they are in fact annotating. If we quote someone on discussion boards or give comments on news paper websites, we are also annotating on the Web.

Bookmarks

Most web browsers support saving URLs to enable users to store their favourites. Those favourites, which are also called bookmarks, help users to store frequently used and interesting web resources. Users can give a name to the bookmarks and order them into folders. This kind of functionality can also be seen as an annotation that appertains to the entire document [12].

Web Annotation Tools and Standards

Annotation tools are developed to add more functionality to the Web in favour of annotating the Web. Some of the tools are developed as a complete web browser (e.g. Amaya browser[5]), others are released as an add-in for browsers (e.g. Diigo[6]).

The Amaya Web Browser is an implementation of W3C's Annotea project [1, 24]. The goal of the Annotea project was to enhance the collaboration on the Web by using Semantic Web technology. The Annotea project describes ways to use RDF data to store annotations. An example of a bookmark represented in RDF is depicted in Listing 2.6.

[5]Homepage Amaya Web browser: `http://www.w3.org/Amaya/`
[6]Homepage Diigo Annotation tool: `https://www.diigo.com/`

```
<bm:Bookmark RDF:about="http://www.annotea.org/AnnoteaURI/neuro/
    Mon_15_May_2006_13_20_27_GMT" dc:title="Audimotor Spike Train Database (ASTD)"
    a:created="Mon, 15 May 2006 13:18:43 GMT" dc:description="Summary of recordings of spike
    trains available for analysis and guide to their context and means for their analysis"
    dc:creator="Marja" dc:date="Mon, 15 May 2006 13:37:35 GMT">
<bm:recalls RDF:resource="http://repositories.cdlib.org/mrrc/1/"/>
<foaf:maker RDF:resource="http://www.annotea.org/marja#marja"/>
<bm:hasTopic RDF:resource="http://www.annotea.org/AnnoteaURI/neuro/
    Mon_15_May_2006_13_35_14_GMT"/>
<RDFS:seeAlso RDF:resource="http://big.sfn.org/NDG/site/eavData.asp?o=29029"/>
</bm:Bookmark>
```

Listing 2.6: Annotea example to describe bookmarks

The information about the way other tools (e.g. Diigo or Annotary) store their data about annotations is not publicly available.

The most important feature that the majority of annotation tools offer is the creation of annotations inside a page. The annotations can be saved for private use or can be shared with others by making them publicly available. A screenshot of an annotation created with Diigo is shown in Figure 2.4. A screenshot of the Annotary toolbar and some bookmarks are depicted in Figure 2.5.

We compare the visualisation techniques for visualising annotations for the different tools that we tested in the next chapter (Section 3.2.1).

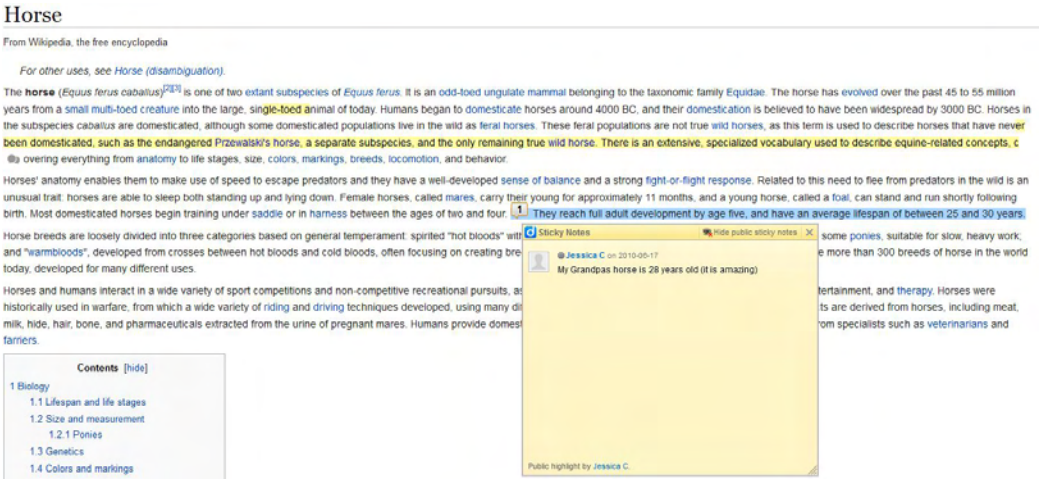

Figure 2.4: An example of an annotation created with the annotation tool Diigo

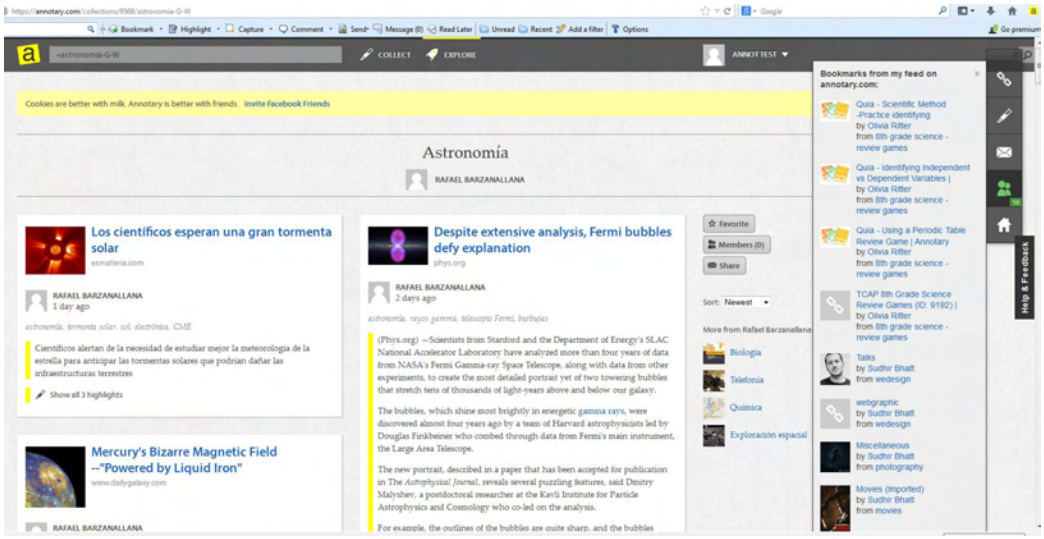

Figure 2.5: A screenshot of the Annotary toolbar and some bookmarks

3 Literature Review

In this chapter we introduce previous work that has been done to enhance the visualisation of links and annotations on the Web.

The visualisation of links has not changed since the beginning of the World Wide Web. Web browsers still use blue underlined text to visualise hyperlinks, and as we will discuss, this kind of visualisation causes readability problems. Furthermore, webbrowsers neglect the metadata authors added to their website. Users are not even aware of the existence of the metadata. The only type of annotations that are covered in the major browsers is the visualisation of links. There is no support for the user to add annotations or to visualise existing ones. Only specific third-party tools allow users to annotate web resources.

Even though there are different types of users (i.e. profiles) that use the Web, a characteristic of the current browsers is that they treat every user in the same way. There are users who want to read something, researchers, or users who want to use the Internet for leisure, etc.

3.1 Visualisation of Web Links in Browsers

In this section we give an overview of the different approaches concerning the visualisation of links in webpages.

3.1.1 The First Web Browser: WorldWideWeb

During the development of HTTP and HTML, Berners-Lee and Cailliau also developed a graphical user interface for browsing and editing HTML documents. This first browser was called WorldWideWeb and untill the end of 1990 it was the only way to browse the Web. In a version from 1993 (see Figure 3.1) we can see that hyperlinks were blue underlined pieces of text which are click-able.

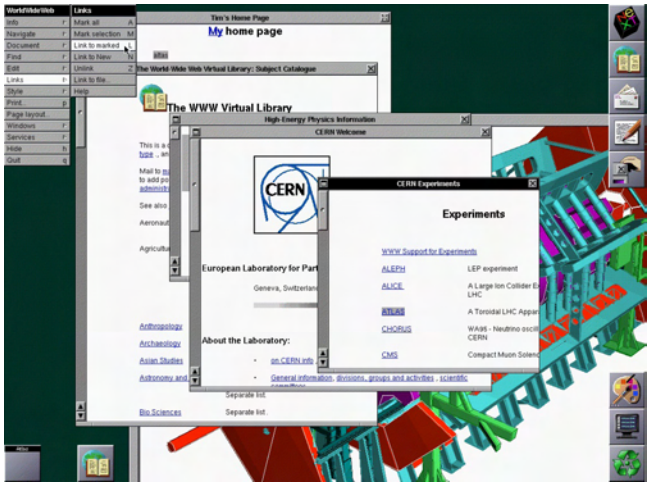

Figure 3.1: Screenshot of WorldWideWeb, the first web browser (1993)

The reason for this blue and underlined visualisation originated from the fact that at that time most of the computers only had a 16-colour monitor. Because blue was the darkest of the available colours, apart from black, the blue colour was the best choice. To be compatible with computers that have a black and white monitor, the links were also underlined [31].

3.1.2 Modern Browsers

Although now we have screens with more colours than the human eye can distinguish (i.e. 10 million colours [21]), the visualisation of links has not been changed as we can see in the standard settings for the appearance of links in modern browsers (e.g. Google Chrome, Internet Explorer, Mozilla Firefox). In most browsers there is a way to change the setting to another style, but only a minority of users change the appearance of links.

From the author's side, there are different ways to change the appearance of links. The appearance of hyperlinks can for instance, be adapted by using CSS styles.

3.1.3 Other Visualisation Techniques for Hyperlinks

In the past, other techniques for the visualisation of links have been investigated. In this section we discuss a few attempts to change the link appearance. By adapting the colours, the font, the size or by adding a marker, they tried to ameliorate the hyperlink appearance.

In his "Guide System" [7], Brown uses different text styles like bold and italic to distinguish links from normal text. This approach limits the way of decorating text because bold and italic text will be seen as links by the user.

To avoid this problem, HyperTIES [33] uses a different text colour for links than for a normal text. Another appearance of links can be found in a browser called Harmony, which supports the "Hyper-G"-publishing system [2]. Harmony uses different background colours to identify links (see Figure 3.2).

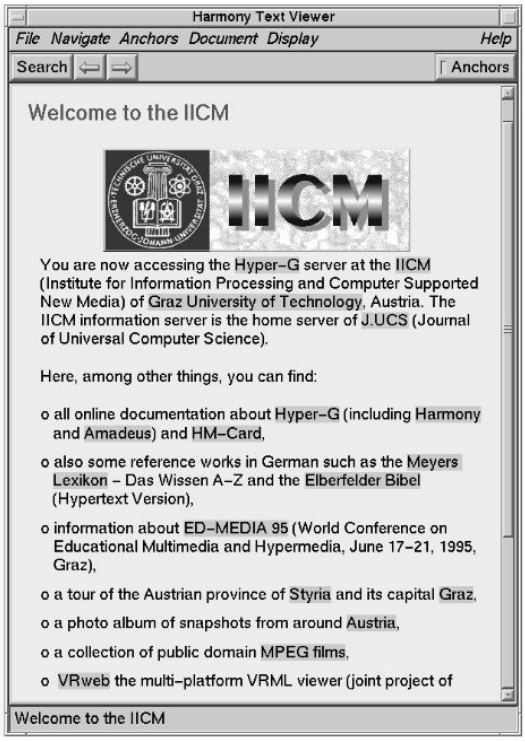

Figure 3.2: Screenshot of the Harmony browser, with highlights to visualise hyperlinks

A completely different approach can be seen in the Intermedia browser (see Figure 3.3). Little arrows were placed above the startpoint of a link, but no marker was placed to show the end. The problem with this method

is that the layout of the page will change because the marker takes up extra space on the page.

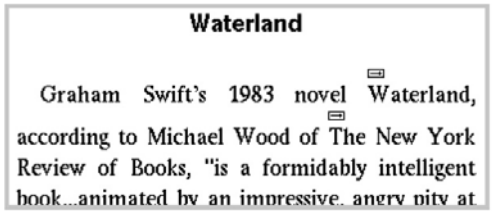

Figure 3.3: Screenshot of the Intermedia browser, with arrows as link markers

If we look at the Neptune hypertext system [11] or UdiWWW [18] (see Figure 3.4), the visualisation is done by drawing boxes around the link markers.

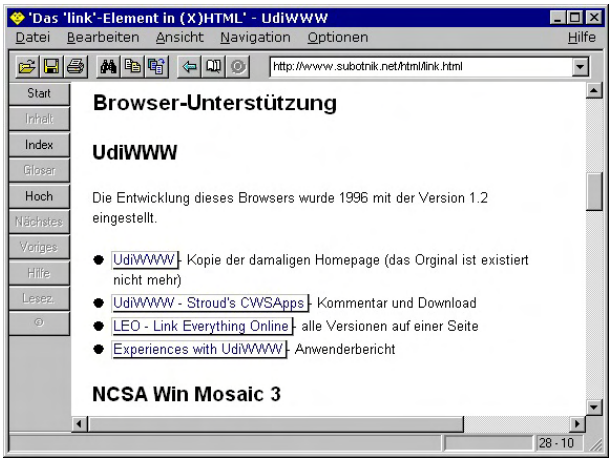

Figure 3.4: Screenshot of the UdiWWW browser, with boxes around the text for visualising links

The visualisation of links in UdiWWW causes less problems regarding the layout, but it distracts the reader of the page because the link is prominently present. In HyperCard and Storyspace, links were also surrounded by boxes, but the boxes were made invisible until the reader presses certain keys (see Figure 3.5) to make links visible [50].

In 1987, during the Hypertext '87 conference, Hypertext Designers compared all existing hypertext systems. The outcome was that the technique which was used by HyperCard and Storyspace was chosen by consensus as the best solution for showing links [6]. The advantage of hiding the links is that the text remains clear and, while people are reading, there is no distraction from link-anchors. From a design point of view, a "links on demand" trigger has to be integrated into the system to prevent distracting mode switches.

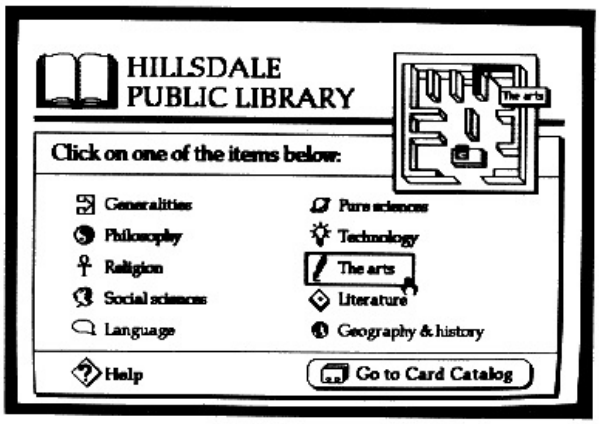

Figure 3.5: Screenshot of the HyperCard system, with boxes around the text for visualising links. Boxes are only visible when hovering over the link

In Symbolics Document Examiner [47], link markers were only highlighted when the mouse passed over them. Looking for links was like playing hide-and-seek. The Distributed Link Service [10] also uses the concept of invisible links. The user can query link servers by selecting text and the user receives results for matching links.

3.1.4 Typed Links

In 1993 Nanard and Nanard [30] indicated the importance of typed links. They mentioned MacWeb, which is a hypertext system providing typed nodes, typed links and a scripting mechanism. Typed links are links with an extra piece of metadata that represents the characteristic of the link between source and destination. In [38] and [26] they present the idea of using an ontology to describe the type of the link.

A link has a lot of characteristics (e.g. type of the destination, protocol that will be used) which may be beneficial for the user. With typed links, we can add more information to the link which can be interesting for the user. In the year 2000, Weinrich and Lamersdorf [49], stated the problem that at that time it was hard for the user to know what would happen when they click on a link in a Web browser. Even with modern browsers it is hard to predict what will happen when a link is followed. Besides the status bar which shows the url, or a part of the url of the destination, there is no other visual marker to indicate what will happen.

We believe that typed links have to be taken into account in any successful visualisation. Moreover, more information about the target of a hyperlink

should be taken into account and visualised to the user. These issues are discussed when we introduce our proposed solution in Chapter 4. In this solution we will benefit both from information about the link itself and the different properties of the destination.

3.1.5 Links on other Webpage Content

Until now we only discussed the visualisation of links tied to text anchors, not to other types of anchors (e.g. images or videos). Of course links can also be placed on other webpage content (i.e. images or videos). The problem with images (or videos) as links is that it is hard for the user to notice that the image is a link. Mostly because the author of the website hides the (standard) border for design reasons. But also because browsers like Chrome[1] hide the border by default. The problem with this is that the only way for the user to know if the image is a link is to hover over the image to see whether the mouse pointer appearance changes. The difference in visualisation on image anchors between both Internet Explorer and Google Chrome can be seen in Figure 3.6 and Figure 3.7 respectively.

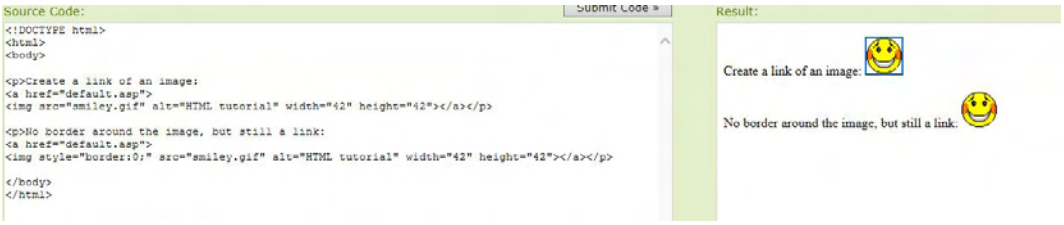

Figure 3.6: The blue border is shown around images that are link anchors in Internet Explorer 10

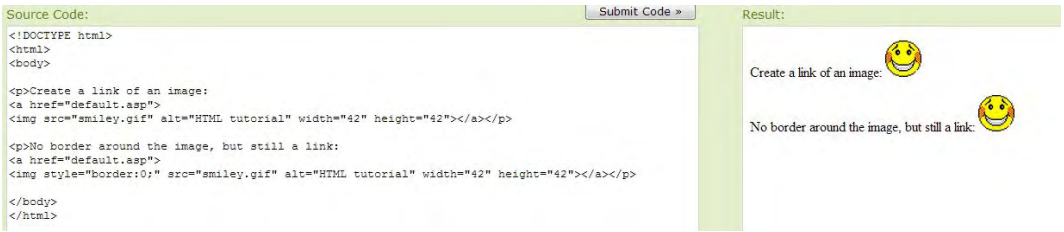

Figure 3.7: The border to indicate that the image is a hyperlink is hidden in Google Chrome

[1]Google Chrome: http://www.google.be/intl/nl/chrome/browser/

3.1.6 Advantages of the Current Visualisation of Hyperlinks

As mentioned earlier, since the first browser was developped in 1990, the blue underlined appearance was chosen. Therefore, it is a widely accepted visualisation of hyperlinks. For most people it gives a good overview of from which part of the text we can navigate to another page.

3.1.7 Disadvantages of the Current Visualisation of Hyperlinks

Visibility

As stated in [27, 16] blue is one of the colours that is hard to focus, especially elderly persons experience problems with perceiving the colour blue. Aging causes difficulty to distinguish dark blue from black. Fortunately the links are also underlined, but the problem with that styling is that it is distracting. We will go into more details in the next paragraph, where we discuss the disadvantages of blue underlined text for link visualisation.

Readability

Obendorf and Weinreich, from the University of Hamburg conducted a study [31] on the effects of the visualisation of links on the readability of the text. Different visualisation techniques such as overlay, underlined and plain text for link markers were presented to the participants. The participants were shown a webpage and after 35 seconds the website disappeared and a few questions popped up. After giving the answers, the participants had to select their certainty about the given answers. Some of the answers were in the text itself, sometimes in the text with hyperlinks. Alternately, the visualisation of the hyperlinks changed between the pages. The scores from the experiment showed that the plain text was best suited for comprehensive reading with a score of 11.58 out of 20. But it nearly matched the score 11.25 out of 20 for the overlay condition. In the underlined condition, the score was 9.08 out of 20. After the experiment, the participants were asked to rate the readability of the text for each visualisation condition. Nearly all participants preferred the text without markers. When they were asked which technique emphasised link phrases the most, 50 % of them voted for the overlays. But when participants had to say how they liked overlay links compared to underlined links, the answer was very heterogeneous.

A different kind of problem that causes readability issues is that underlining text causes the bottom parts of descenders (i.e. the letters like g, j and y) to be masked. As stated in [23], the visual rhytm and variation of ascenders and descenders aid the reading process by creating strong external shapes in words. By stroking through the bottom parts of letters the rhytm is broken.

Styling

Another disadvantage is that text that is underlined, or text in blue, can sometimes give the impression that the text is a hyperlink. And therefore, it limits the styling of text that is not a hyperlink because you can not use blue text or underlined text without causing confusion.

3.1.8 Visualisation of Overlapping Links

Overlapping links are two or more links that share (part of) the anchor, but with a distinct destination. For the visualisation of this type of links, the Harmony-browser uses a technique as depicted in Figure 3.8. This technique can help when there are only a few overlaps, but when there are more it becomes a problem. In our proposal in Chapter 4 we will use another technique to solve this problem.

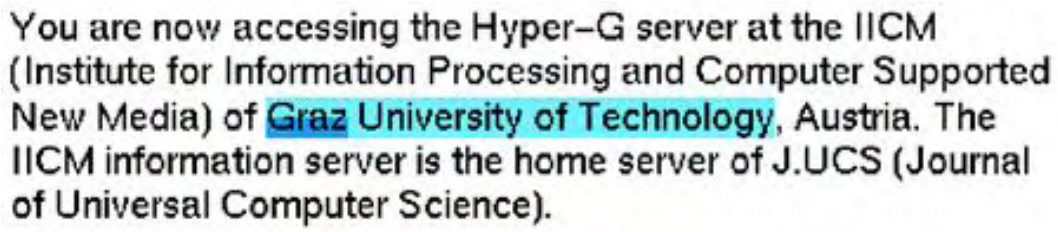

Figure 3.8: Screenshot of the Harmony browser which uses different highlight colours to distinguish links that overlap each other

3.2 Visualisation of Annotations

3.2.1 Annotation Software

During our research we tested some annotation tools. The most important characteristics of the different tools are as follows:

Annotea and Amaya Web browser

The Annotea project [24], is a project of the W3C to enhance collaboration via metadata based on web annotations and bookmarks. They use an RDF-based annotation schema for describing the annotations as metadata, and to locate the annotations in the document they use an XPointer. When users open a webpage, annotations can be loaded from one or multiple annotation servers. As mentioned in [25] by Koivunen, a member of the Annotea team, Annotea sees humans as meaning makers:

> "They create concepts, make associations, comment, clarify, annotate, review, organize, bookmark, tag and file information organised into categories and folders."

Amaya[2] is the first client implementation of Annotea. Its development started in 1996 as a showcase for the web technologies. They tried to integrate as many W3C technologies as possible, as well as RDF [22] and XPointer implementations (i.e. the Annotea project). It is an open source web browser and web editor, therefore we can browse but also create webpages and upload them to a server. The latest version dates back to January 18th 2012, and on the overview-page of the Amaya project can be read that the development was stopped [1]. Because it is a web editor, we have a lot of possibilities to annotate a webpage. The annotations can be saved for both private and public use.

Annotary

A completely other solution for annotating the Web is to use Annotary[3]. A web application together with an add-on for a web browser lets users bookmark, comment and highlight text on webpages. The data is stored in a personal storage in the Annotary system, but the format in which the data is saved is unknown. The most important functionalities are:

- Bookmark pages
- Notes (annotations for the complete page or on a part of the page)
- Highlights (with or without comment)
- Share the annotations with groups
- All annotations and bookmarks are saved in your account

[2] `http://www.w3.org/Amaya/`
[3] `https://annotary.com/`

- Add-on for a web browser (toolbar) + web application

Diigo

With a 12 person team, an annotation tool called Diigo[4] was created. Diigo stands for Digest of Internet Information, Groups, and Other Stuff. The Diigo tool is a web browser plugin. It is a tool with a lot of capabilities, the most important functionalities [14, 13] are:

- Bookmark pages
- Sticky notes (annotations for the complete page)
- Highlights (with or without comment)
- Add tags to your annotations
- Private / public annotations
- Share the annotations with groups
- All annotations, bookmarks are saved in the cloud
- Add-on for web browser (toolbar) + web application

Based on our research, it is the most complete tool on the market, although improvements can be made to make the application more user friendly. In the following section we initiate some improvements, which we will discuss in a more profound way in Chapter 4.

In Figure 3.9 we can see a comparison of the most interesting tools we tested. We compared them on how they filter the annotations, what functionality they offer related to annotations and the possibilities they have to share annotations.

3.3 Discussion

If we look at links, we can conclude that the Web offers more information that can come in handy for the user, but which is not shown. From a visualisation point of view, we learn from other research [50, 31] that how links now appear on the Web can cause some readability issues.

If we look at annotation tools, they all have in common that they only use text and highlights to visualise the annotations that are created by their

[4] `https://www.diigo.com/`

Category	Property	Amaya	Diigo	Annotary	A.nnotate
Types of annotations possible					
	Hyperlinks in page	☺	☹	☹	☹
	Bookmark a page	☺	☺	☺	☹
	Highlight	☺	☺	☺	☺
	Annotations on Marked text	☺	☺	☺	☺
	Annotations on Document Level	☺	☺	☺	☺
	Hyperlinks between fragments - same page	☹	☹	☹	☹
	Hyperlinks between fragments - different pages	☹	☹	☹	☹
	Annotation on Hyperlinks between fragments	☹	☹	☹	☹
Filter options / Visualisation options					
	Show only public	☺	☺	☺	☹
	Show only private	☹	☺	☹	☺
	Show private and Group annotations	☹	☺	☺	☹
	Show all annotations	☺	☺	☹	☹
	Show no annotations (= normal page)	☺	☺	☹	☹
	Show only annotations of group x	☹	☺	☺	☹
	Show metadata from the authors of Webpages	☹	☹	☹	☹
	Give an overview of the links on a page	☹	☹	☹	☹
	Give an overview of the metadata on a page	☹	☹	☹	☹
	Filter links on link type	☹	☹	☹	☹

Figure 3.9: A comparison of different annotation tools

own tool. We can search in the text to find the highlight you are looking for, or you can use the search engine from the tool to find the annotation in a list (not in the text itself).

Another problem that can arise is that a page can become overwhelmed when there are a lot of annotations on the page. As we can see in Figure 3.10 on the Wikipedia logo there are a lot of sticky notes. They are placed on top of each other and therefore it is hard to find the annotation you are looking for. To filter the annotations, they provide a list of all annotations in the Sidebar (i.e. a toolbar) which is also depicted in Figure 3.10. It can help filtering, but if you are really looking for annotations with a specific subject, it is still not easy to find it.

What we think that also has to be solved is that the user should have the possibility to get an overview of a page. This overview should be targeting both the content (subjects) on the page, and how the webpage relates to other pages. In Chapter 4 we come up with a solution for those and other visualisation and filtering issues.

Figure 3.10: Screenshot from the Diigo Annotation tool. The Wikipedia logo on the Wikipedia page is cluttered with sticky notes

4
Wysinwis

In this chapter we explain our approach for solving problems related to the visualisation of annotations/hyperlinks on the Web.

There are a few important things that we believe have to be solved to improve the user friendliness of web browsers. In the first few paragraphs we quickly run through the different problems we want to solve. In the following sections, we discuss our approach for resolving those problems.

First of all we think that users of the Web have to benefit more from the metadata that the authors of webpages add to their website. We will show some visualisation techniques to represent the data in a nice way. Another problem we have mentioned earlier is concerning the readability problems that can occur due to the blue underlined text as the appearance of links. By using another visualisation, regardless of how the author of the webpage intended to visualise links, we can counteract these readability problems. One more problem we will tackle is the issue with a lot of annotations on a page which result in a chaotic webpage. To solve all these issues, we will present a filtering and visualisation solution that we called the Wysinwis principle. Wysinwis is an acronym that stands for What You See Is Not What I See.

An important benefit of Wysinwis browsing is that the focus of certain fragments will be brought to a user based on the user's profile. Other annotations that are less important based on the profile, are visualised as plain text. This will enhance the readability of the text and will help the user to

find the most important fragments faster.

Another important benefit of Wysinwis browsing is that there is more information available for the user, which is now only available for search engines. The Semantic Web becomes more obvious to the users. This promotes the Semantic Web, and authors of webpages will be more inclined to add semantic data to their webpages.

If we look at small devices connected to the Internet, the Wysinwis principle will help by emphasising, and scrolling down automatically to the most important information.

4.1 The Wysinwis Concept

In this section we want to present the Wysinwis principle as a proposal for the improvement of the visualisation of annotations on a page. As we already mentioned, Wysinwis is an acronym that stands for 'What You See Is Not What I See'. Basically, the Wysinwis principle means that we render a webpage differently for different users. In other words, we change the rendering of a page based on the user's profile. The reason is that we want to limit the number of annotations on a page, because they have a negative effect on the readability. Therefore, we want to disable some annotations on a page and only enable the most interesting ones by means of the user's profile. In that way, only valuable information is visible and other annotations are neglected. This limits the amount of annotations on a webpage, which will increase readability.

Another important characteristic of our Wysinwis principle is that we do not only include annotations created by annotators, but also annotations created by the author of the webpage. In Sections 4.2.1, 4.2.2, 4.2.3 and 4.2.4 we will give an overview of the techniques authors and annotators can use to create links and annotations and we show how we can merge them and treat them in the same way.

It is a pity that in most annotation tools only the annotations created by the tool itself are taken into account. In contrary, the Wysinwis principle is also about annotations created by authors of webpages. In the current version of the Web there are some techniques for authors to add semantics to their websites. Therefore the authors of webpages are also important creators of annotations.

It is important to see that the user does not loose any webpage content on the page. Some fragments will be emphasised and others will not, according to the user's profile and preferences.

In the previous paragraphs we talked about the content on the webpage, and how the Wysinwis concept changes the visualisation of that content. We can supplement this approach with additional visualisation techniques that also make use of the annotations on the page. We propose to show some information to the user that gives an overview of the page. Based on that information, the user can decide whether to read the page or not. We can accomplish that by showing the user some keywords about the page. To give the user an idea of how the page relates to other pages, we can show a link overview graph that depicts the links with other pages.

In the following sections we will introduce how the user profile is composed and how we will define an annotation in our solution.

4.1.1 User Profile

As we mentioned earlier, the Wysinwis principle will render a webpage based on a user profile. What we see as a user profile in our solution is just a list of tags which describe the user's interests. In our prototype, described in Chapter 5, we used a simple comma-separated list of tags which is fixed, but a user profile can also be composed during browsing. A user profile for example can be based on a fixed list of keys, but also supplemented with the keywords used during a search on the Web via a search engine. Based on the tags mentioned in the user profile, the Wysinwis rendering mechanism will decide whether or not to show certain annotations and links.

The users have to benefit from the metadata, and we also give users the flexibility to add more metadata. In the following section we will describe the metadata we need in our solution.

4.1.2 Annotations

Annotations consist of an anchor, along with the metadata related to this anchor. An anchor can be a complete page but it can also be a fragment of a page. When an annotation has a whole page as an anchor, we will call it an annotation on page-level. When the anchor corresponds to a fragment of the text, we call it the in-text-level. For our Wysinwis solution, we need at least some tags as additional info for the annotation. We define more than only tags as metadata for our annotations. The reason to define more than only tags is twofold. First of all we want to be able to extend our solution and the second reason is that we have to merge annotations by annotators, and annotations by authors of webpages.

Therefore in our solution an annotation is composed of:

- an anchor
- a piece of text (e.g. a comment)
- one or more tags
- key and value pairs

In the following paragraphs, we explain the different parts of an annotation in our solution. The goal of the text field is to add a comment to the anchor. The text can be empty, which allows the creation of a simple annotation without a text linked to it. For the tags we did not choose a system with predefined tags but a collaborative creation of tags. This kind of system, called a folksonomy [39, 15] system, has already proven its usefulness on popular websites like delicious.com[1]. Apart from the popularity, we chose to use a folksonomy system because there are an infinit number of subjects that can be applied to annotations. Therefore there is no way we can create a vocabulary from the beginning. The list of key and value pairs assures that characteristics and their value can be saved for the annotation. The key/value pair-list can be null. This key and value pair will be used to easily merge the annotations of annotators with those that are created by authors. This will be explained further on in this chapter. In the following section we discuss hyperlinks because they require extra properties.

4.1.3 Hyperlinks

We already mentioned that hyperlinks are a subtype of annotations, so we inherit the properties we mentioned in the annotations-section (see 4.1.2). But because hyperlinks have to store more information than annotations we will discuss these properties in this section. In addition to inherited properties, hyperlinks have a destination and a link type. This destination can be a URL, or it can be a specific fragment (i.e. anchor) on a URL. The link type is the description of the link between both source anchor and destination anchor. The link types are predefined, as opposed to the tagging system, which is based on the folksonomy principle. The reason why we did not choose a folksonomy for link types, is because there is no infinit number of link types, in contrast to the number of tags that can exist.

[1]Home page of the social bookmarking service delicious: `delicious.com`

To summarise, in our solution a link is composed of:

- an anchor (inherited from annotation)
- a piece of text(inherited from annotation)
- one or more tags (inherited from annotation)
- key and value pairs (inherited from annotation)
- a destination (URL or a fragment)
- a link type

To come up with a list of link types, we based ourselves on an ontology created by Knowledge Media Institute[2] (KMI) [34]. Because a link exists in four forms, the link type can be used in the following ways:

a fragment of a webpage	**link type**	*a fragment of a webpage*
URL	**link type**	*a fragment of a webpage*
a fragment of a webpage	**link type**	*URL*
URL	**link type**	*URL*

Table 4.1: Table with all the possibilities for linking

The link types are classified in different categories. These categories exist only to organise the link types for the user. As an example you can find the content of two categories (i.e. "Supports" and "Challenges") with some of their link types. The link types in the example can be used for instance for research.

- Supports
  - proves
  - agreesWith
  - isConsistentWith
  - isEvidenceFor
- Challenges
  - refutes
  - disagreesWith
  - isInconsistentWith
  - isEvidenceAgainst

[2]The homepage of the Knowledge Media Institute: `http://kmi.open.ac.uk/`

4.1.4 Levels of Annotations

The Wysinwis principle makes a distinction between two levels of annotations. First of all we define page-level annotations as annotations which have a URL as anchor. In other words a page-level annotation is about the whole page. On the other hand, in-text annotations are defined on a fragment of the webpage. Therefore the anchor is a URL, but supplemented with a mechanism to select a fragment of a webpage (e.g. XPointer). In Figure 4.1 an example is given for the two levels of annotations.

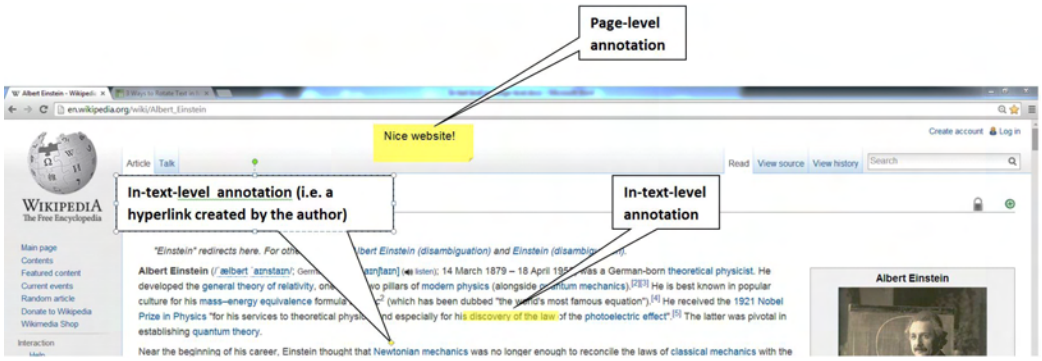

Figure 4.1: Levels for annotations

We provided the information on how we compose annotations and links, and how we defined a user profile. We also discussed the two levels of annotations. Because an important property of the Wysinwis principle is that we treat the annotations from the authors in the same way as the annotations of annotators, the following section explains how we can merge the annotations of both annotators and authors.

4.2 Annotations and Links in Wysinwis

In this section we describe the way authors annotate the web content, and how annotators can create annotations on web content. In the first two subsections we will explain how annotations can be created, in the next two subsections we describe the creation of links. We will treat page-level as well as in-text-level for annotations and links. We use some examples to show how metadata can be created for the two levels, first as an annotator, then as an author. Each subsection ends with a table that contains different possibilities to fill out the properties of respectively annotations and links. Because the tables for annotations and respectively links share the same structure, it

becomes clear that we can merge annotations from authors and annotators effortlessly.

4.2.1 Page-Level Annotations

By Annotators

Annotators can add information on page-level just by creating a comment on the URL. An example of the concept of a page-level annotation can be a sticky note for a page, like Diigo does. If we match this kind of page-level annotations to our definition, we are left with the possibilities mentioned in Table 4.2.

Reference	**Tags**	**Key & value pairs**	**Text**
URL	Tags introduced by the user of the annotation tool	null	null
URL	Tags introduced by the user of the annotation tool	null	Text introduced by the user of the annotation tool
URL	Tags introduced by the user of the annotation tool	Key & value pairs introduced by the user of the annotation tool	null
URL	Tags introduced by the user of the annotation tool	Key & value pairs introduced by the user of the annotation tool	Text introduced by the user of the annotation tool

Table 4.2: Shows how annotators can create page level annotations, and how the properties of an annotation are filled in

By Authors

Page-level annotations created by authors are defined in the `head` section of the HTML-page. The most commonly known technique is the `meta` tag in

HTML. As in the data structure dictionary, the key can be specified by the `name` attribute, and the associated value can be specified by the `content` attribute. Another way for an author of a webpage to add metadata is the `title` tag.

```
<head>
 <title>Visualisation of annotations Thesis</title>
 <meta name="description" content="My Thesis">
 <meta name="keywords" content="HTML,Annotations,Links">
 <meta name="author" content="David Swalus">
</head>
```

Listing 4.1: Page-level annotations by the author of the webpage

If we place this kind of page-level annotations within our definition, we receive the possibilities mentioned in Table 4.3. As tags we can use the list of keywords, the key and value pairs of the metadata form the key and value pairs of our annotation.

4.2.2 In-Text Annotations

In-Text annotations are annotations that exist on a part of the page. The fragments can be any type of data (e.g. text, images) that is on the page, or hybrid.

By Annotators

If we look at in-text annotations, we can not base ourselves merely on a URL, like we did for the page-level annotations. To identify a fragment, we have to use the URL together with the XPointer, or another way to describe a fragment of a webpage. The other properties that we used on page level for the annotators remain the same. Table 4.4 shows the list of possibilities.

By Authors

Authors can also add metadata to certain fragments in the text by making use of microformats[3], RDFa[4] or another mechanism that allows machine-readable data to be embedded in HTML documents. In the example (Listing 4.2) below we make use of RDFa to describe the data. We utilise two different Schema's, the Dublin Core[5] (prefix *dc*) and the Friend of a Friend schema[6] (prefix *foaf*). Of course other interesting schema's exist or can be created.

[3] http://microformats.org/
[4] http://www.w3.org/TR/xHTML-rdfa-primer/
[5] http://dublincore.org/
[6] http://www.foaf-project.org/

Reference	**Tags**	**Key & value pairs**	**Text**
URL	The list of keywords in the `meta` attribute with "keywords" as a name. (in the example in Listing 4.1 the keywords are: HTML, Annotations, Links)	null	null
URL	The list of keywords in the `meta` attribute with "keywords" as a name. (in the example in Listing 4.1 the keywords are: HTML, Annotations, Links)	null	The title (i.e. `<title>`) can be used as the text property
URL	The list of keywords in the `meta` attribute with "keywords" as a name. (in the example in Listing 4.1 the keywords are: HTML, Annotations, Links)	Key & value pairs are the name and content pairs in the `meta` attribute.	null
URL	The list of keywords in the `meta` attribute with "keywords" as a name. (in the example in Listing 4.1 the keywords are: HTML, Annotations, Links)	Key & value pairs are the name and content pairs in the `meta` attribute.	The title (i.e. `<title>`) can be used as the text property

Table 4.3: Shows how authors can create page-level annotations, and how the properties of an annotation are filled in

Reference	Tags	Key & value pairs	Text
URL + XPointer	Tags introduced by the user of the annotation tool	null	null
URL + XPointer	Tags introduced by the user of the annotation tool	null	Text introduced by the user of the annotation tool
URL + XPointer	Tags introduced by the user of the annotation tool	Key & value pairs introduced by the user of the annotation tool	null
URL + XPointer	Tags introduced by the user of the annotation tool	Key & value pairs introduced by the user of the annotation tool	Text introduced by the user of the annotation tool

Table 4.4: This table shows how annotators can create in-text-level annotations, and how the properties of an annotation are filled in

The example (Listing 4.2) is validated with the Markup Validation Service from W3C with the DocType set as "XHTML + RDFa".

```
<body about="http://example.org/david-s/#me">
 <h1>My profile</h1>
 <p>My name is
  <span property="foaf:nick">David S</span>
  and I like annotations
 </p>
 <p> My
  <span rel="foaf:interest" resource="urn:ISBN:0752820907">favorite book is the inspiring
   <span about="urn:ISBN:0752820907">
    <cite property=\textit{dc:title}>Weaving the Web</cite> by
    <span property="dc:creator">Tim Berners-Lee</span>
   </span>
  </span>.
 </p>
</body>
```

Listing 4.2: In-text-level annotations by the author of the webpage

In Table 4.5 all possibilities are depicted. For both the tags property and the text property of our annotations, we have to come to some agreement on which property we will use as the tags property, and the text property. We chose *dc:title* from the Dublin Core for our annotation's text property. For the tags property, we can take the nucleus from the *dc:title*.

Reference	Tags	Key & value pairs	Text
URL + XPointer (`<span>`)	The nucleus from the *dc:title*	null	null
URL + XPointer (`<span>`)	The nucleus from the *dc:title*	null	Text introduced by the user of the annotation tool
URL + XPointer (`<span>`)	The nucleus from the *dc:title*	Key & value pairs specified by the Microdata properties or RDFa properties and their value	null
URL + XPointer (`<span>`)	The nucleus from the *dc:title*	Key & value pairs specified by the Microdata properties or RDFa properties and their value	Text introduced by the user of the annotation tool

Table 4.5: Shows how authors can create in-text-level annotations, and how the properties of an annotation are filled in

4.2.3 Page-Level Links

By Annotators

Annotators can create page-level links just by linking two URLs to each other, or by creating links between a URL and a fragment of another document. Link types can be added to explain the link. Table 4.6 shows the possibilities. For the sake of simplicity we do not copy the properties that are inherited from *normal* annotations.

By Authors

To create page-level links, authors can use the `link` tag which is mostly used to link to a stylesheet, as in the following example (Listing 4.3). Although the W3C's intention for the `link` tag is more than just linking stylesheets, in the major browsers there is only support for stylesheets and not for other references.

Reference	Destination	Link type
URL	URL	The link type introduced by the user of the annotation tool
URL	URL + XPointer	The link type introduced by the user of the annotation tool

Table 4.6: Shows how annotators can create page-level links, and how the properties of a link are filled in

```
<head>
 <link rel="stylesheet" type="text/css" href="theme.css">
</head>
```

Listing 4.3: Page-level annotations by the author of the webpage

With the `href` attribute in the `link` tag, the linked URL can be set and by using the `rel` attribute tags can be added to give more information about the link. In Listing 4.4 an example of a valid excerpt of HTML code that shows a link definition on page-level.

```
<head>
 <link rel="agreesWith" href="http://en.wikipedia.org/wiki/Annotation">
</head>
```

Listing 4.4: Page-level annotations by the author of the webpage

In Table 4.7 the possibility to create links on the page level is mentioned, and how we can integrate it into our definition of links. The destination can be a URL, but the URL can also be a link to an anchor inside a page.

Reference	Destination	Link type
URL	URL defined in the `href` tag (The URL can be the URL to a page, or to an anchor inside a page)	The link type introduced by the `rel` tag

Table 4.7: Shows how authors can create page-level links, and how the properties of a link are filled in

4.2.4 In-Text Links

By Annotators

Annotators can, by selecting a fragment of the text, add a link to another fragment on the same page, or a fragment on another page, or to a complete page. The possibilities are shown in Table 4.8

Reference	Destination	Link type
URL + XPointer	URL	The link type introduced by the user of the annotation tool
URL + XPointer	URL + XPointer	The link type introduced by the user of the annotation tool

Table 4.8: Shows how annotators can create in-text-level links, and how the properties of a link are filled in

By Authors

With HTML5 you can create links on fragments by using the anchor tag (`<a>`) together with its `href` attribute. To express the link type in HTML5 we use the `rel` attribute, extra tags that contain information of the linked document like `hreflang` attribute, to express the language, will be stored in the key/value pairs of the annotation. An example can be found in Listing 4.5. The different properties of the links are depicted in Table 4.9

```
<a href="http://www.google.com/" rel="foaf:interest" xml:lang="de">Search engine</a>.
```

Listing 4.5: Page-level annotations by the author of the webpage

Reference	Destination	Link type
URL	URL defined in the `href` tag (The URL can be the URL to a page, or to an anchor inside a page)	The link type introduced by the `rel` tag

Table 4.9: Shows how authors can create in-text level links, and how the properties of a link are filled in

4.2.5 Merging

The merging of the annotations from authors and annotators for the different levels, is not difficult since the tables have the same structure. We can merge the data for every property (i.e. column in the table) in the same way as we can merge the tables together. This merging technique is one of the main concepts in the Wysinwis principle.

4.3 Visualisation Techniques for Annotations

In the previous section we discussed how we can merge the annotations from authors with the annotations from annotators. In this section we discuss some visualisation techniques that can help the users to find their way through the different annotations/links on a page. First we discuss the visualisations on page-level, afterwards we discuss the visualisation techniques that we use in the text (i.e. in-text-level). This section ends with an explanation of the benefits of these visualisation techniques.

4.3.1 Page-Level Link Visualisation

Sometimes it is interesting for a user to visualise the available links on a page to get an overview of how this page relates to other pages. We think of a dendrogram as one of the best ways to accomplish this task because it is commonly used for the visualisation of trees and clusters. An example of such a dendrogram is depicted in Figure 4.2.

If we use this kind of diagramme to visualise hyperlinks we can use its nodes to mark the anchors (URLs). The edges can then be used to visualise the link type. Seeing as properties exist that relate to the anchors, which can be of interest to the user, it may be wise to add these to the visualisation. This can be done without overloading the visualisation itself. The protocol that is used to get to the anchors (link resolution) can also be marked by a colour. The filetype of the anchor can be marked via an icon at the anchor. To point out that a link is (currently) unavailable, it is indicated by means of a dotted line. More information about the anchor (i.e. file size, tags, comments) can be portrayed by hovering over the anchor. A prototype of this hyperlink-visualisation is depicted in Figure 4.3.

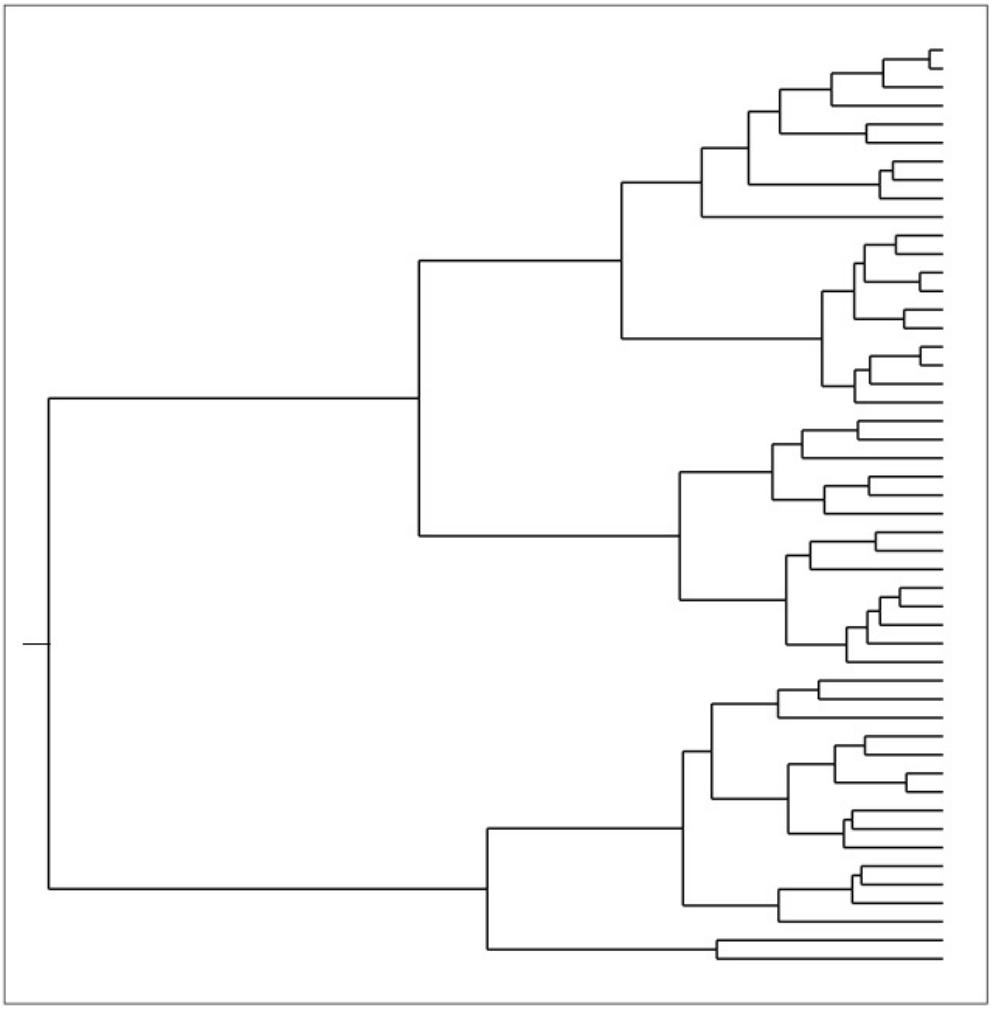

Figure 4.2: An example of a dendrogram

4.3.2 Page-Level Annotation Visualisation

Tag overview

To commence the visualisation of annotations, we would like to present a bubble chart as a way to give an overview of a page. It is a tag cloud that shows all the tags that are used in the page, either by the author or by annotators. The bigger the circle, the more important the tag is for that page. This visualisation has the benefit that you can see the importance of a keyword on a website in just a glance. Because we work with tags that are created by a folksonomy-system, problems can arise [20]. Inter-User Agreement (i.e. different tags with the same meaning) is one of those

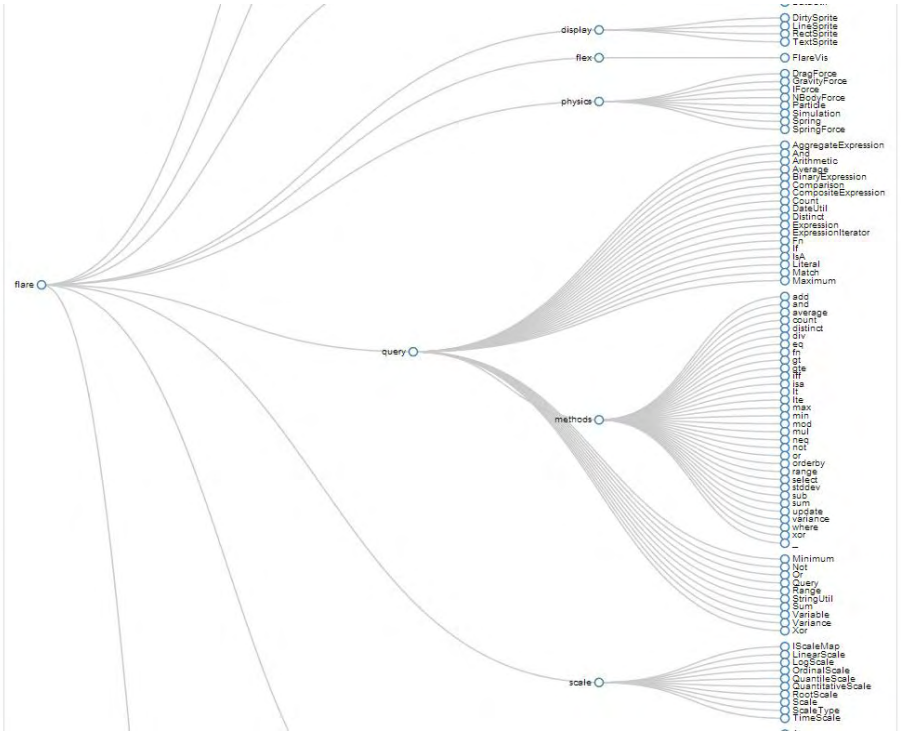

Figure 4.3: An example of a dendrogram used for link visualisation

problems. To narrow down this problem we will hand propositions to the user when he is tagging. Another problem, called tag disambiguation (i.e. same tag, but with a completely different meaning), is something that will be solved by the collaborative tagging itself because we allow multiple tagging. One tag can describe and narrow down the meaning of another tag.

4.3.3 In-Text-Level Filtering

In previous sections we discussed page-level visualisations to give the user an overview of the content of a page through the Annotation-Overview visualisation, and to give an overview of how the page is linked to other pages on the Web. The data for those visualisations came from the page-level. In the following sections we will give an overview of filtering and visualisation techniques for the in-text-level to indicate to the user where on the page there are links and annotations.

To ameliorate the readability we have to filter annotations in a way that only the annotations that are important for the user (by means of the user profile) are visible. We therefore want to present four techniques.

- **PlainText view**: To really enhance the readability, we can filter every annotation in a way that only plain text remains. According to the study of Weinreich, Obendorf and Lamersdorf [50, 31] this is the best way to read a webpage. There is no distraction for the reader, and the reader can focus on the text itself.
- **Profile view**: This visualisation filters the annotations based on the active profile of the user. Only the annotations with tags that correspond with the active profile will be shown, the other annotations are neglected. We show the annotations as yellow highlighted text, as mentioned in [50, 31] as the second best way (i.e. plain text is the best way) to visualise hyperlinks.
- **Original view**: Sometimes it can be handy to see the webpage in the same way authors intended to visualise the webpage.
- **(Original) Highlight view**: Based on the original view, the highlight view shows only the annotations from the authors but not visualised as in the original view the way the author designed it, but instead yellow highlighted text is used.

4.3.4 In-Text-Level Visualisation

To show the information about annotations, we think a tooltip is the most appropriate way to do this. With a tooltip we can give contextual information to the user. A mockup of such a tooltip is represented in Figure 4.4. This Figure shows how the anchor-text is placed on top of the page, followed by a summary of the hyperlinks and/or annotations. The anchor text is visible because it is possible that an overlap is created between different anchors. At the place where the tooltip is called upon, it is possible to have two or more anchors at two intersecting text sections related to different annotations. Of course a problem arises when there are many overlaps within the text. To resolve this problem, users can first select some text in which they are interested in, and then hover over that selection to evoke the tooltip. This tooltip would then only give the annotations regarding the selected text. Because of this feature, together with the Wysinwis filtering technique, we solve the problem with overlapping links we mentioned in Section 3.1.8.

In Figure 4.5 another visualisation technique is portrayed. This visualisation shows all the tags that are used inside the text (i.e. on the in-text-level, not on page-level). The bubbles are clickable and they can be used to change the appearance of the webpage. By clicking on a bubble, the bubble will be toggled (i.e. on -> off; off -> on), and the annotations and links that

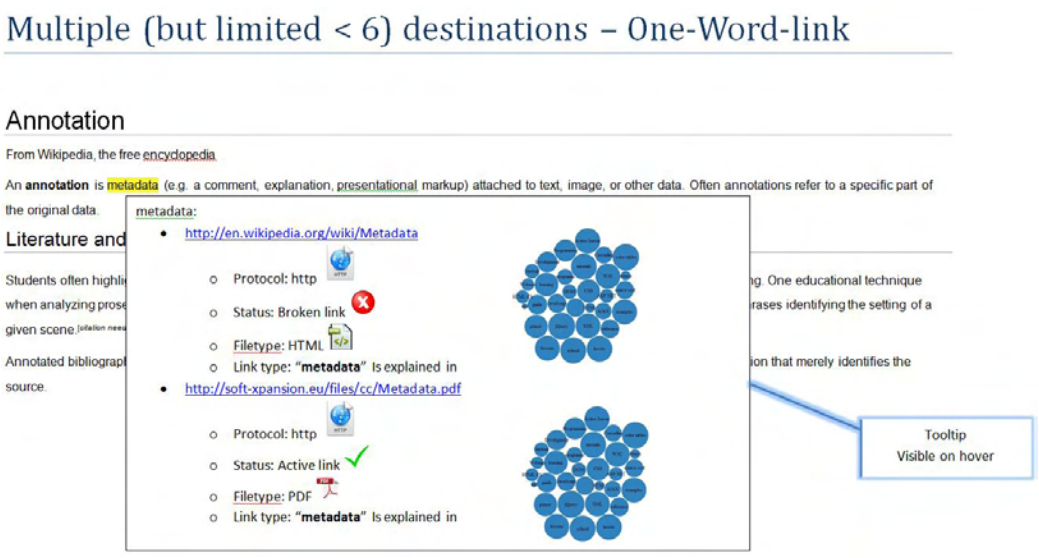

Figure 4.4: One of the mock-ups we drew to create our prototype. This figure shows the tooltip when hovering over an annotation or a link in the text

correspond with the tag will be emphasised, or not, depending on the status (i.e. on or off) of the bubble. This technique allows users to overrule their profile by selecting other tags to adapt the page rendering.

4.3.5 Benefits of the Proposed Visualisation Techniques

This section is a short summary of why these visualisation techniques are user-friendly and how they bring annotations closer to the user.

First of all there is a tag cloud which gives an overview of the webpage in the form of keywords. This gives the user an impression of whether or not the webpage contains the subject the user is looking for. The link visualisation can be called upon to give an overview of the linked documents.

If we can request both visualisations on the screen in a simple manner (e.g. by shortcuts), this can improve user-friendliness on the Web. To enhance the readability of the text we chose to mark annotations in the form of yellow highlights. Even though this choice is based on previous research [50, 31], we will still put this to the test by conducting an independent evaluation.

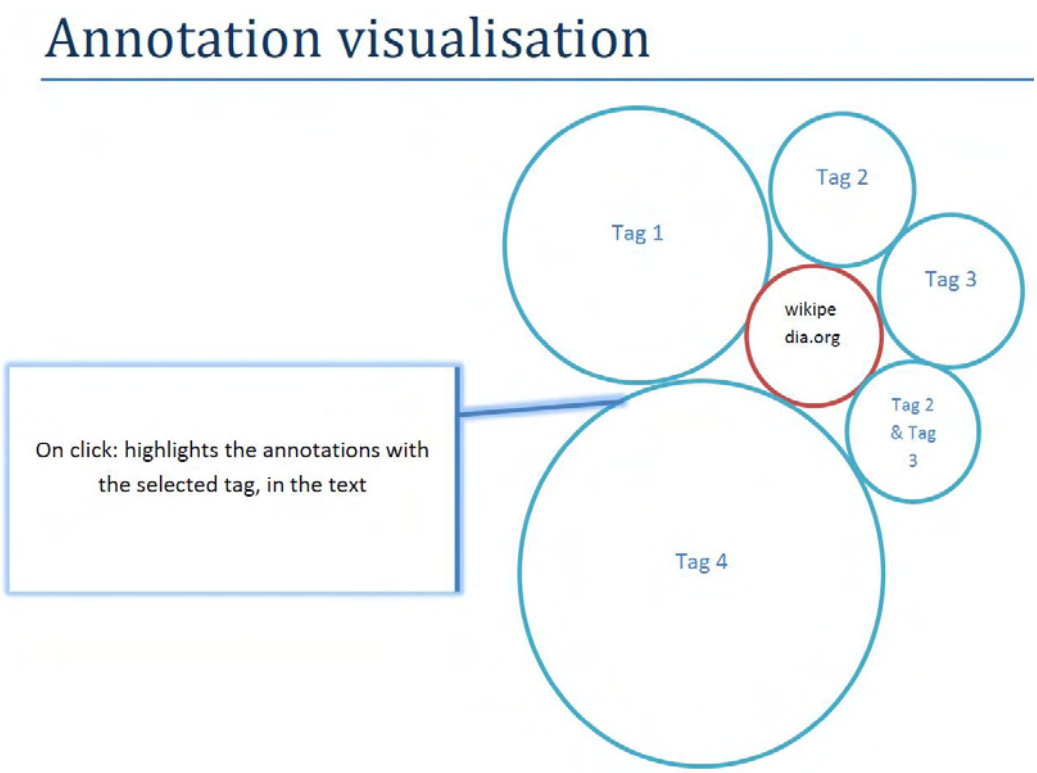

Figure 4.5: The Bubble Chart, used to show all the tags in the text

5 Prototype

In this chapter we present the Wysinwis viewer, a prototype based on the Wysinwis principle. The prototype was built with the sole purpose of allowing the evaluation of the Wysinwis principle.

5.1 Used Technology and Libraries

In the following subsections we will briefly introduce the different libraries and technologies that we used to create our prototype. Figure 5.1 shows the different libraries and technologies that were used.

5.1.1 Google Chrome Extension and Google APIs

We have built the Wysinwis viewer as an extension of a web browser because we can not force users to escape to a new browser. The reason we chose to extend Google Chrome is because we based ourselves on the web browser statistics from W3Schools[1]. According to their statistics, Google Chrome is the most used Internet browser. In Figure 5.2 we can see a screenshot of how the Wysinwis viewer is listed on the extension page of Google Chrome.

[1]The homepage of W3Schools: `http://www.w3schools.com/`

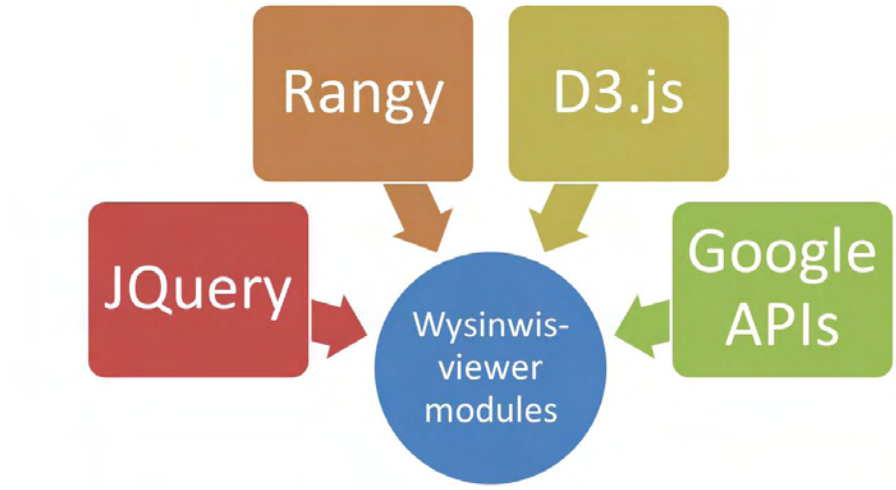

Figure 5.1: The libraries used in the Wysinwis viewer modules

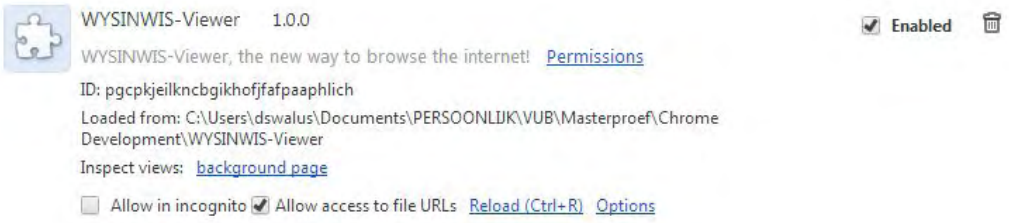

Figure 5.2: The Wysinwis viewer enabled in Google Chrome

5.1.2 jQuery

jQuery[2] is a very popular multiplatform JavaScript library. Many important companies use jQuery on their websites [46] (e.g. Google, Microsoft, IBM, Netflix). The most important feature is the easy way of working with the DOM of a webpage. You can not only select DOM objects, you can also modify them. Apart from that, jQuery facilitates the use of ajax and working with HTML events. We use jQuery in our Wysinwis system to work with the DOM of a webpage, to fire events and to use Ajax.

Here below we show a code excerpt (see Listing 5.1) from the Wysinwis viewer code for the building and visualisation of the tooltips during the on load event on the page.

```
$(function () {
    $(document).tooltip({
        items: "img, [data-hyperlinks], [title], a",
        preload: true,
```

[2]jQuery Homepage: `jQuery.com`

```
        content: function () {
            var element = $(this);
            var ToolTipText = "";
            if (element.is("[data-hyperlinks]")) {
    ....
```

Listing 5.1: jQuery example

5.1.3 Rangy

Rangy[3] is a cross-browser JavaScript designed to work with selections on webpages. We use this library to capture selections, and to persist them to a database.

In the following code excerpt shown in Listing 5.2, Rangy is used to get the selection from the user and to serialise it to an object, in the example the object is called "rangyrange". The rangyrange object will be used to persist the selection to the database. A pre-defined css style, in this example the css class is called 'Wysinwis-highlight', can be used to change the appearance of a rangy selection.

```
var sel = rangy.getSelection();
var rangyrange = rangy.serializeSelection(sel, true);
 ...
highlighter.highlightSelection("Wysinwis-highlight", sel);
```

Listing 5.2: Rangy example

It is a pity that not all parts of the library are fully released/tested, which causes some bugs.

5.1.4 D3.js

D3.js is a library written in JavaScript that can be used for the manipulation of documents based on data. It is built for data visualisation by using HTML, together with CSS and JavaScript. D3 stands for Data-Driven Documents and the development started in 2010. D3 allows the users to bind arbitrary data to a Document Object Model (DOM), and then apply data-driven transformations to the document. To create the visualisations, D3 uses the SVG (Scalable Vector Graphics) standard. By using D3 functions, datasets can easily be bound to SVG objects. Those different SVG objects are the building blocks for a complete visualisation.

To show a few of the unlimited possibilities of visualisations with D3.js, we show a few examples in Figures 5.3, 5.4 and 5.5.

[3]Rangy Homepage: `https://code.google.com/p/rangy/`

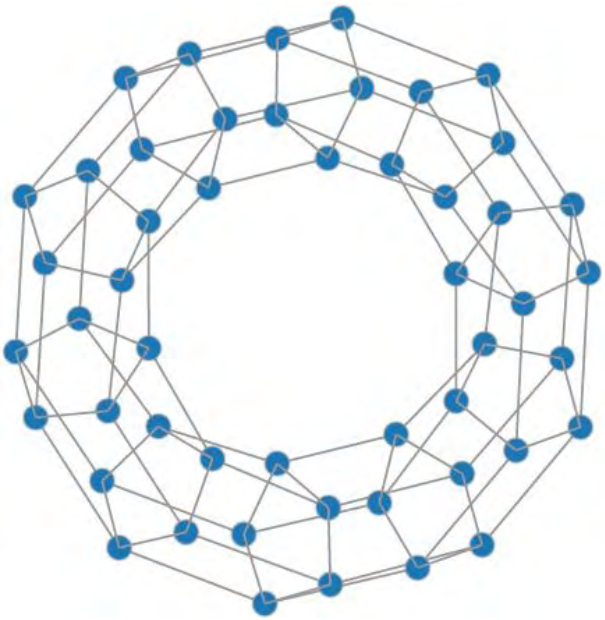

Figure 5.3: An example of the possibilities with D3.js.
Source: `http://christophermanning.org/projects/building-cubic-hamiltonian-graphs-from-lcf-notation/`

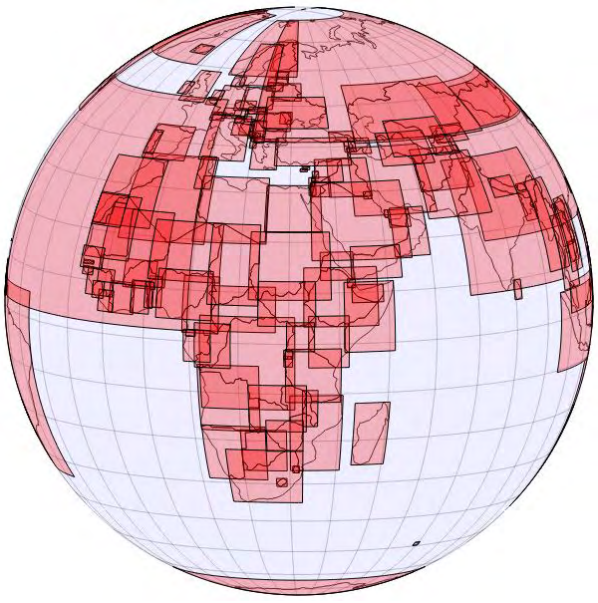

Figure 5.4: An example of the possibilities with D3.js.
Source: `http://www.jasondavies.com/maps/bounds/`

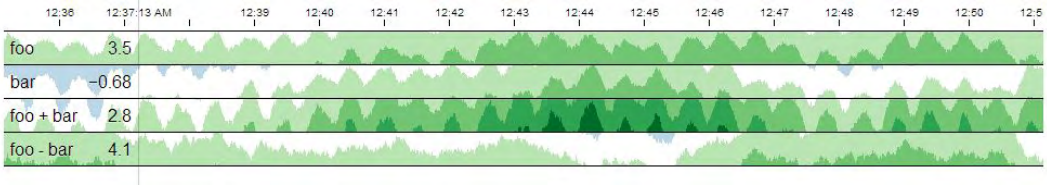

Figure 5.5: An example of the possibilities with D3.js.
Source: `http://square.github.io/cubism/`

5.2 Database

The database engine is a Web SQL Database and is based on the Resource-Selector-Link model [36, 35] (RSL model). The database consists of different tables, which we will briefly explain. The Resource table contains the web resources. The web resources are identified by the URL of the resource. The Selector table specifies the fragment of the document that will be selected as the destination or the source of a hyperlink, or the source of an annotation. The Selector table is linked to the Resource table by storing a reference to the associated resource. The selectors that we use in our Wysinwis viewer are serialised Rangy objects. We also have tables to store Comments, Tags, Link Types and Users.

Although, the goal of the Wysinwis principle is to use it in a collaborative setting, we used a local database instead of a shared one, to limit the complexity of the development of the prototype. The use of a local database suffices to show the different aspects of the Wysinwis principle for the evaluation.

5.3 Architecture

In this section we discuss the architecture of the Wysinwis viewer. An overview of the architecture is shown in Figure 5.6. Most of the scripting is done by injecting code into the webpages that the user opens. This is done by contentscripts, which have access to the DOM of the current page. The Wysinwis viewer consists of multiple contentscripts to accomplish several tasks. We have contentscripts for the different diagrammes such as the bubble chart and the link diagramme. Other contentscripts are created to support the different modes of visualisation i.e. profile view, original view or PlainText view (see Chapter 4). The background scripts allow the communication with the database, handle the context menu and read the local storage

that persists the settings. This local storage persists default behaviour regarding what mode has to be used when starting Google Chrome, and it stores the user profile. These settings are managed by the user by means of a settings script.

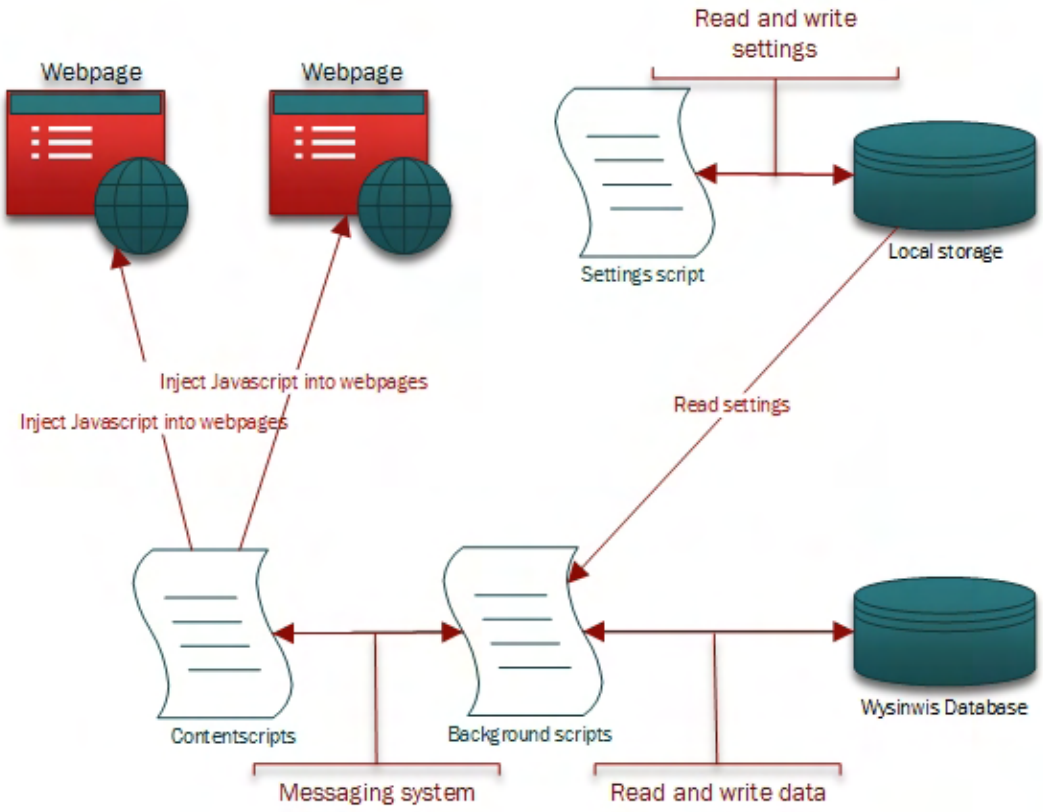

Figure 5.6: The architecture of the Wysinwis viewer

Between the contentscripts and the background scripts, a messaging system is put into place. To show how the messaging-system works, a sequence diagramme for the creation of an annotation is depicted in Figure 5.7. This diagramme is a simplified version, we only mention a successful path of the data, we do not mention any data validation.

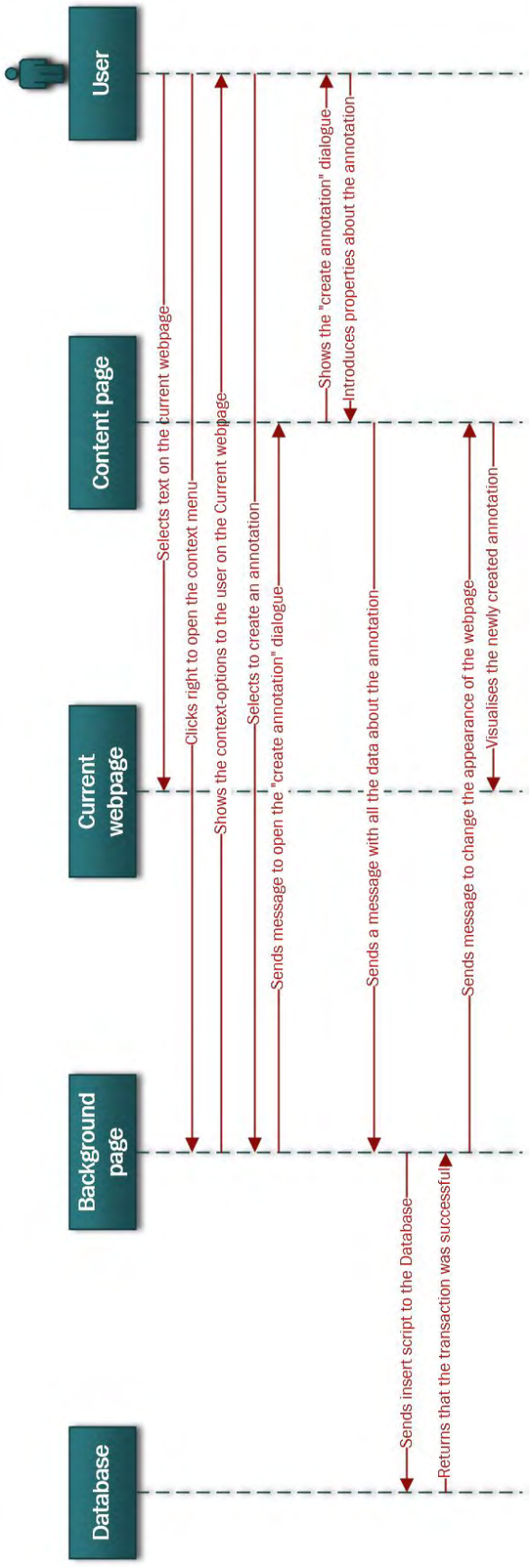

Figure 5.7: A sequence diagramme for the creation of an annotation

5.4 Use Cases

5.4.1 Text Visualisations

The user has the possibility to visualise the page in five different ways. Four of them are part of the Wysinwis principal, the other one (i.e. Show everything) exists for debugging purposes, but it is mentioned here because it was available for the users during the evaluation. In Figure 5.8 the context-menu of the Wysinwis viewer shows the different in-text visualisations. These visualisation modes will be discussed further on in this section.

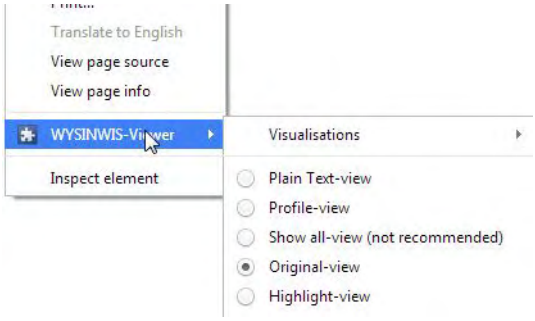

Figure 5.8: Context menu for the user to switch the visualisation modes

Original View / Author View

This visualisation ensures that the page is rendered the way the author created it. The HTML will therefore not be adjusted by the Wysinwis viewer and annotations from annotators will not be visible. In Figure 5.9 a page in the Original view is shown.

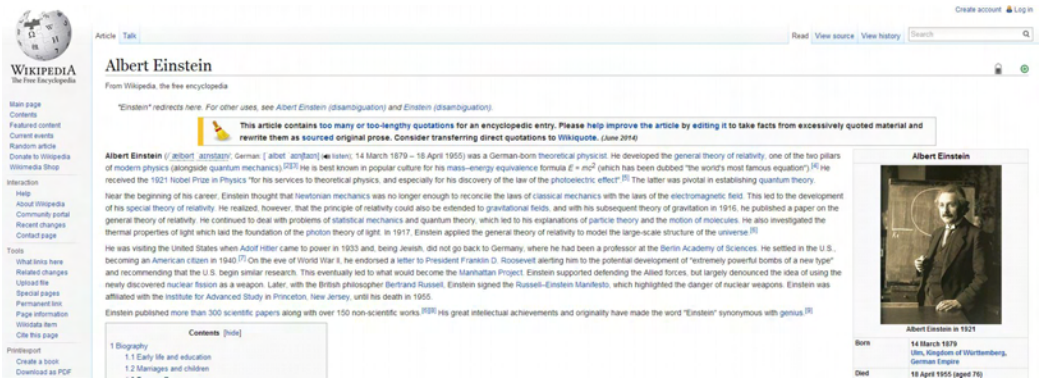

Figure 5.9: The Original view mode of the Wysinwis viewer. The page is shown as the author created it

PlainText View

In this visualisation, the Wysinwis viewer will deactivate the annotations and links from both authors and annotators. It is still possible to use the links, but visually they are portrayed as plain text. This way, a clean text is shown with black letters and a white background, which improves the readability. The example from Figure 5.9 is depicted in PlainText view mode in Figure 5.10.

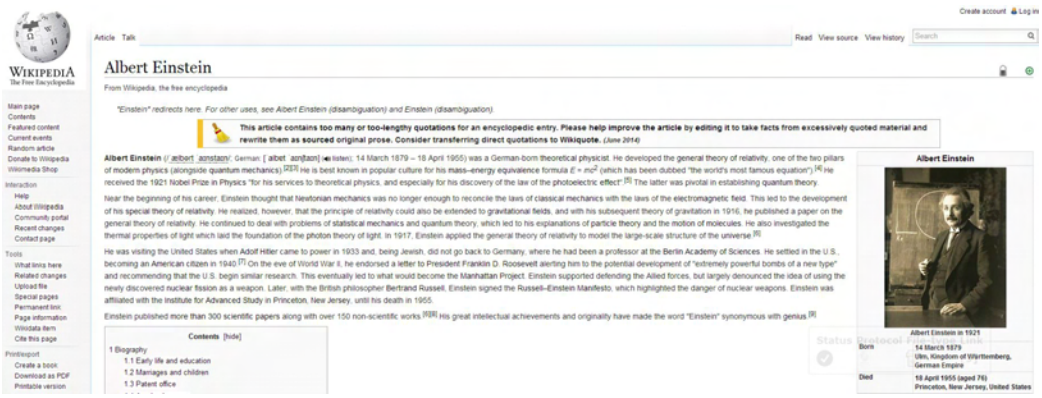

Figure 5.10: The PlainText view mode of the Wysinwis viewer shows the page without any annotations or links

Highlight View

The Highlight View shows the "Original View / Author View", but instead of the normal blue underlined texts for links, yellow highlighted text will be used. As explained earlier, the highlighting should ameliorate the readability. It also helps the user to find images that are a link. In Figure 5.11 the Highlight-style of links and annotations is shown.

Profile View

The profile visualisation will only show the annotations and links that comply with the tags, stored in the profile of the user. This has the advantage of limiting the number of annotations on the page, and only the most important information stands out. This view is the most important one because it benefits the most from the Wysinwis principle. The annotations that are shown because they comply with the profile, are portrayed with a yellow background colour and black text. An example is shown in Figure 5.12.

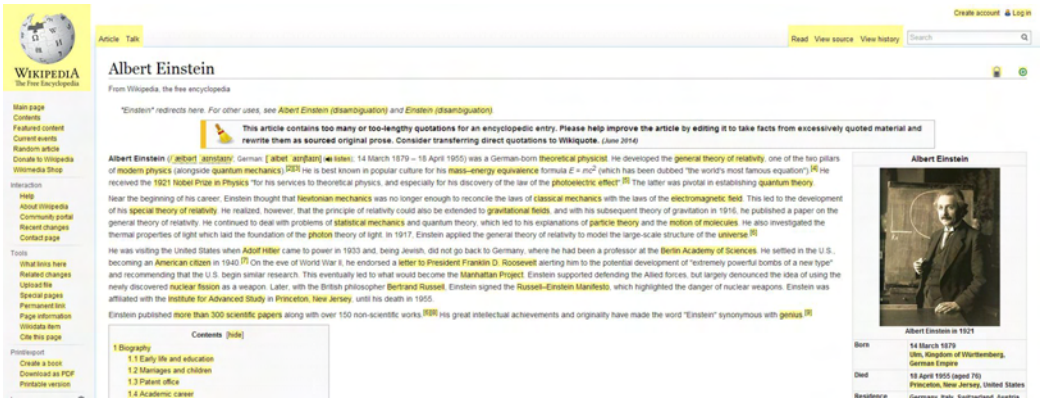

Figure 5.11: The Highlight view mode of the Wysinwis viewer shows all links and annotations in black text which is highlighted in yellow

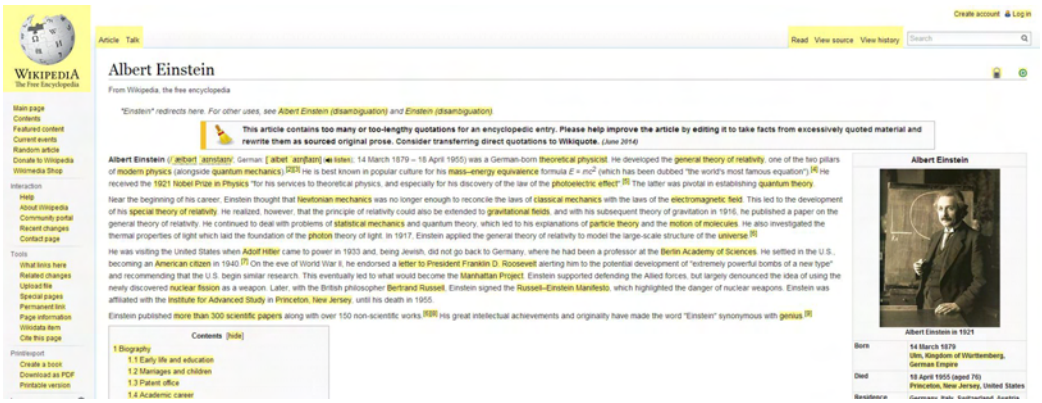

Figure 5.12: The Profile view mode of the Wysinwis viewer shows all links and annotations that correspond to the user's profile in black text which is highlighted in yellow

Show Everything

With this visualisation, all the annotations and links are shown. This visualisation can cause some readability problems when there are a lot of annotations. The page can become overwhelmed with data. Although it is not a recommended setting, it can be useful if the user is really interested in all the annotations.

5.4.2 Page Overview Visualisations

Until now we discussed only in-text visualisations. This section will show external (i.e. not in-text) visualisation techniques that can help the user to get an overview of a page and therefore get a better understanding of the topics related to the page as described in Chapter 4.

Page-Level - Bubble Diagramme

In the Bubble diagramme on page-level, the tags for all the annotations and links on the page are shown. This concerns every form of annotation, the annotations stored on page-level in the database by annotators as well as those that are added in HTML by the authors of the webpage (see 4.2.1 and 4.2.4). This functionality gives the user an overview of the tags used on the page through one visualisation. Based on this visualisation, the user can decide whether or not the page contains the information the user is looking for. The Wysinwis icon will appear in the address bar when there is metadata available. The visualisation is depicted in Figure 5.13, on the upper right side the Wysinwis icon is shown.

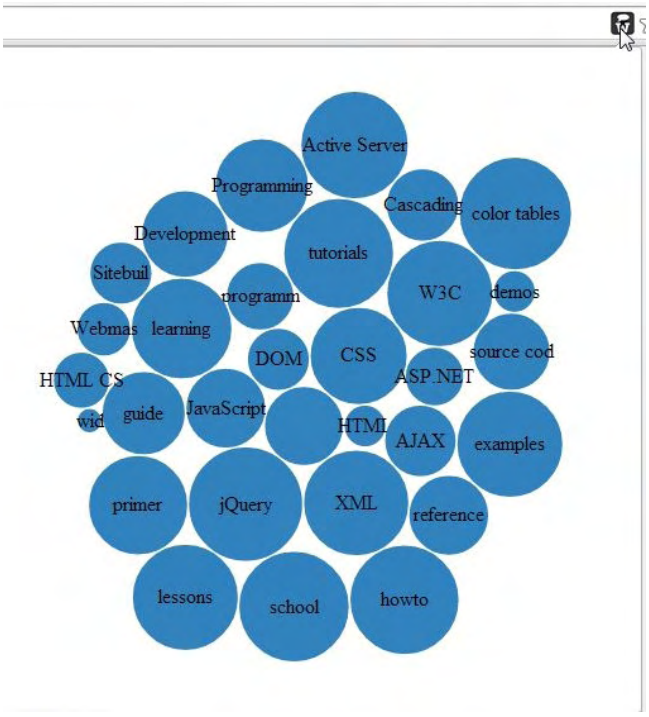

Figure 5.13: The Page-Level Bubble diagramme on from the Wysinwis viewer for `http://www.w3schools.com/`

In-Text-Level - Bubble Diagramme

The Bubble diagramme for the in-text level, shows all tags that are used on the page. This bubble diagramme is, as opposed to the bubble diagramme on page-level, an interactive diagramme. The bubbles are clickable and allow the user to change the visualisation of the page. When a bubble is clicked, all the annotations with the tag that corresponds with the bubble, are highlighted as in the Highlight view. The bubbles can be toggled on and off to start and stop highlighting respectively. When opening the Bubble diagramme, the bubbles that correspond with previously enabled annotations (i.e. when you have already used the bubble diagramme, or when the Profile-view is on) will be toggled on. An example of this visualisation can be found in Figure 5.14.

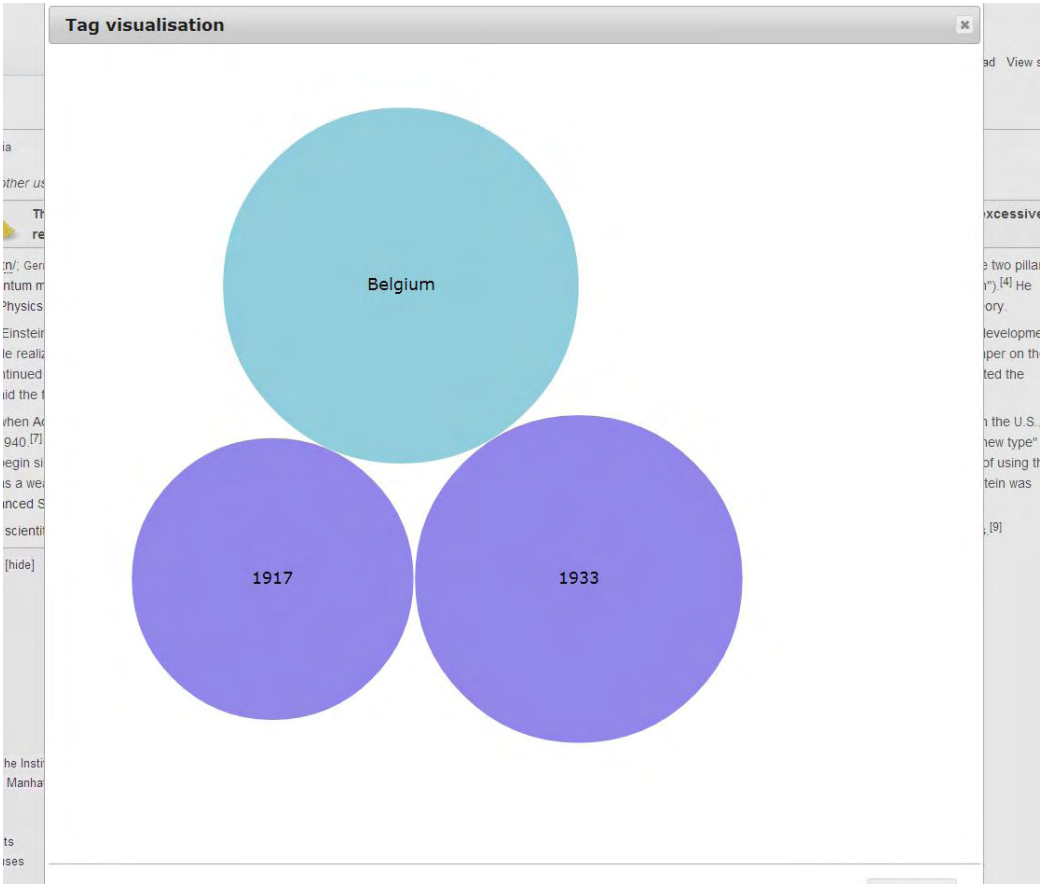

Figure 5.14: The Bubble diagramme on In-Text Level from the Wysinwis viewer. The Belgium-tag is toggled, which enables all the annotations which have the Belgium-tag linked to it

Link Diagramme

The Link diagramme shows all the links on a page, both the ones created by the annotators and those that are created by the authors. The links can be filtered based on file type, link type and protocol type. The Link diagramme dialogue is depicted in Figure 5.15. The Figure shows the diagramme together with the possibility to filter the links in the diagramme.

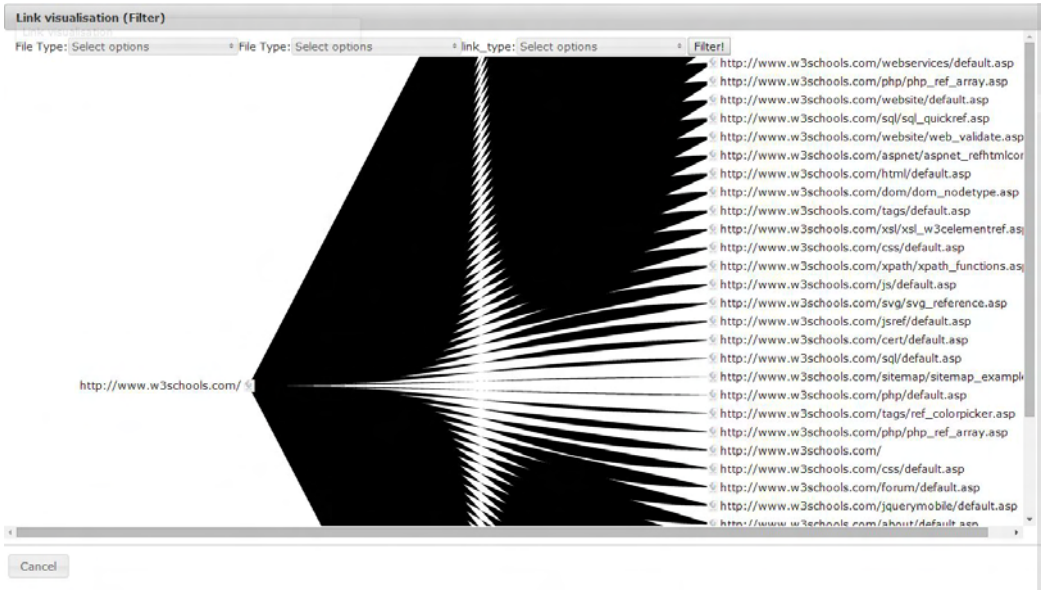

Figure 5.15: The Link diagramme dialogue which shows all the links on a page. The visualisation can be filtered on file type, link type and protocol type

5.4.3 Annotation Creation

In Figure 5.16, the dialogue for the creation of annotations is shown. The dialogue for the creation of links is depicted in Figure 5.17.

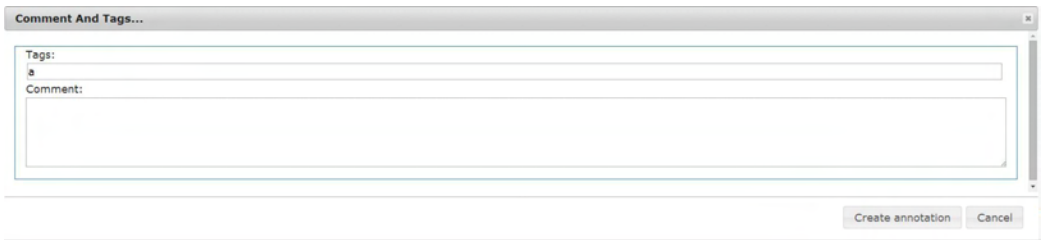

Figure 5.16: Shows the dialogue for the creation of an annotation in the Wysinwis viewer

Figure 5.17: Shows the dialogue for the creation of a link in the Wysinwis viewer

5.4.4 Change the Settings for the Wysinwis Viewer

In Figure 5.18, the settings page of the Wysinwis viewer is shown. It allows the user to define the default behaviour for the text-Visualisation (see 5.4.1) of the Wysinwis viewer. The settings page also permits the user to specify his profile. For the sake of simplicity, the profile is a comma-separated list of tags.

Figure 5.18: The Settings page of the Wysinwis viewer

6 Evaluation

In this chapter we will discuss the results of the evaluation of the Wysinwis viewer to evaluate the whole Wysinwis concept. In the first section we begin by explaining the evaluation methodology. Afterwards, we discuss the different evaluation techniques that we used in more detail. We conclude this chapter with the results and conclusions based on the evaluation.

6.1 Evaluation Methodology

We will evaluate the Wysinwis viewer by means of both a quantitative and qualitative evaluation. This approach is based on the findings of Greene, Caracelli and Graham [17] and Mandinach[28]. In their study they found that the mixed-method approach is more effective. A quantitative evaluation stands for reliable and objective results and outcomes. Whereas a qualitative evaluation is more subjective, but may shed a light on valuable information that may otherwise not have been found. Another reason to combine both methods is that we will gain a more complete understanding, and get more feedback ([9]). In the following sections we will discuss the setup, the data analysis and the evaluation of both evaluation techniques. To conclude this chapter we combine the results of these methods and we will formulate a conclusion.

6.2 Quantitative Evaluation

6.2.1 Goal

The goal of this evaluation is to get a better insight into the way people experience the use of the Wysinwis viewer. We evaluate the visualisation techniques as well as the Wysinwis principle itself. We will, among others, measure user-friendliness and added value for the features of the Wysinwis viewer.

6.2.2 Setup

Nine volunteers participated in the evaluation. The group of evaluators consisted of heterogeneous persons in terms of background, age, sex and mother tongue. The goal of the tool was explained in a demo, together with the different functionalities and the visualisation techniques. After the demo, the participants were asked to follow some task scenarios to get in touch with the system and to guide them through all the different aspects of the software. Afterwards, they were free to explore the tool in a more unrestricted way.

After several days of working with the tool, the participants were asked to give their opinion on certain aspects of the tool in the form of a questionnaire. The complete questionnaire is included in this work as an appendix (see Appendix A). The evaluators, which do not all have a technical background, are permitted to leave some questions open when they do not understand the question. The questionnaires were then completed during the interview, after an explanation of the question. This questionnaire will be used as a foundation for the interview which will be discussed in Section 6.3.

6.2.3 Questionnaire

Layout

The questionnaire was created based on the article and latex-file of Hartenstein[19]. The questions of the Questionnaire were sorted based on the different sections of the Wysinwis viewer. In total, 39 questions were asked. Below is the classification along with a bit of explanation:

- **About you (3 questions)** *This is the demographic part which only asks the name, age and gender of the surveyed person.*
- **General Questions (11 questions)** *This part of the questionnaire inquires about the basic knowledge of the Semantic Web. This section*

also inquires the surveyed about how they experience the general use of the Wysinwis viewer, and what they thought of the user friendliness.

- **Questions about Visualisations (9 questions)** *This part of the questionnaire asks the users about the visualisations in the Wysinwis viewer.*
- **Wysinwis idea and Filtering (6 questions)** *The Wysinwis idea and filtering section of the questionnaire enquires the participants about their feelings about the Wysinwis concept with the filtering and in-text visualisation techniques.*
- **Only for Authors of webpages (2 questions)** *Because the Wysinwis idea demands discipline and extra work from authors of webpages, in both questions we ask them if they see the benefit for the user and if they think it is in proportion with the augmented work load.*
- **Enhancements (8 questions)** *The Wysinwis viewer implements the ideas described in Chapter 4, and shows that it is possible to build a system based on the Wysinwis principle. It can be expanded very easily with extra functionality that will enhance user friendliness as well as the "willingness to use". In this section some propositions are made to expand the Wysinwis viewer and the users are asked to what degree they find this expansion is interesting.*

The questionnaire consists of 2 types of questions:

- **Yes/No questions:** A clear choice must be made between agreeing with the statement and not agreeing with the statement.
- **Scaled questions:** The surveyed has to answer this type of question by indicating on a scale of 0 to 5, to what extent they agree with the statement.

Based on the results of the questionnaire, the questions for the qualitative evaluation were prepared (see Section 6.3).

6.2.4 Results

General Questions

In Figure 6.1, the results of some important characteristics of the Wysinsis viewer are illustrated. The Figure shows the average scores on a scale of 5 for:

- The benefit the participants see in the visualisation of Semantic data to the user. In the graph the bar is named "Benefit"
- In the bar indicated with "Personal use" the average score that the participants gave to the question whether or not they would use the Wysinwis viewer, is portrayed.
- To the question if they would recommend the Wysinwis viewer, the average score of 4 is shown in the "Recommendation"-bar
- The Wysinwis viewer-concept is based on contributions, therefore the participants were asked if they would make contributions to the system.
- The ease of use is depicted in the "Easy to use"-bar.
- The participants were also asked if the Wysinwis viewer would ameliorate the browsing-experience on the Web.

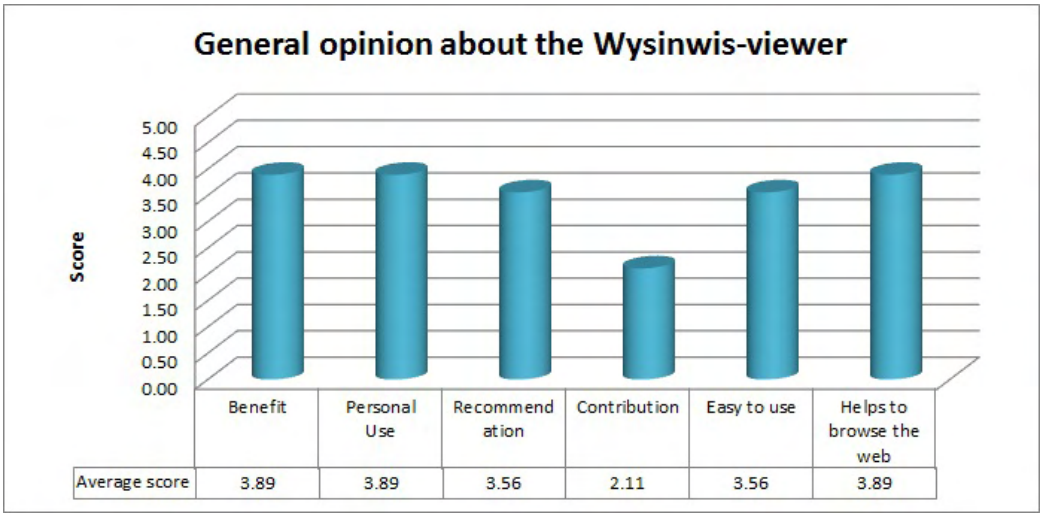

Figure 6.1: The results of the general questions about our system

As shown in Figure 6.1, the Wysinwis viewer reaches relatively high scores. Only the question about making a contribution to the system scores less than

half of the points. We had expected this score to be low. We asked in the "Enhancements" part of the questionnaire, to what extent the participants would be inclined to add annotations to the system if there was a link to Facebook [1]. This feature would for example give the user of the Wysinwis viewer the possibility to add annotations directly on the user's Facebook-page. The results are depicted in Figure 6.2. To interpret the results correctly, the questionnaire enquires the participant on his use of Facebook. The graph only includes the participants that use Facebook.

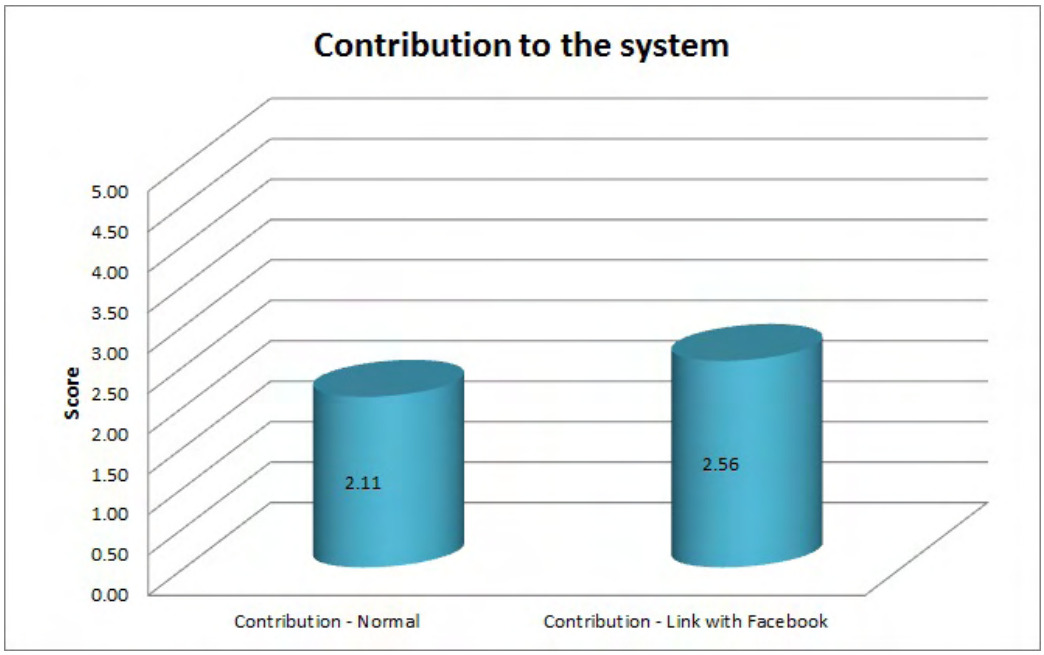

Figure 6.2: The difference in Contribution when we provide a link with Facebook and when we do not

As seen in Figure 6.2, the ability to link something to Facebook increases the desire to contribute from a 2.11 score to a much better score of 2.56. This is an augmentation of more than 20%. This shows that participation with a tool like the Wysinwis viewer will be enhanced when there is a connection with social media.

Visualisation Techniques

In the questionnaire we also evaluated the visualisation techniques we used. We asked questions on both the visualisation in the text itself, and the graphs we used to give an overview of the used tags and hyperlinks.

[1] The Homepage of the Social Media Website "Facebook": https://www.facebook.com/

Based on the findings of Weinreich and Obendorf [50, 31] we introduced the functionality to visualise links as yellow highlighted text instead of the standard blue underlined text. We asked the participants if this kind of link visualisation is less distracting. In Figure 6.3 the results for this question are shown. The same Figure (Figure 6.3) also shows the average score for the funcionality that allows links and annotations to be completely disabled. Only one person scores this less than 3 which proves, with an average score of 4.56 out of 5, that participants find plain text to be a lot easier to read than text with links and other annotations. This result supports the research of Weinreich and Obendorf [50, 31].

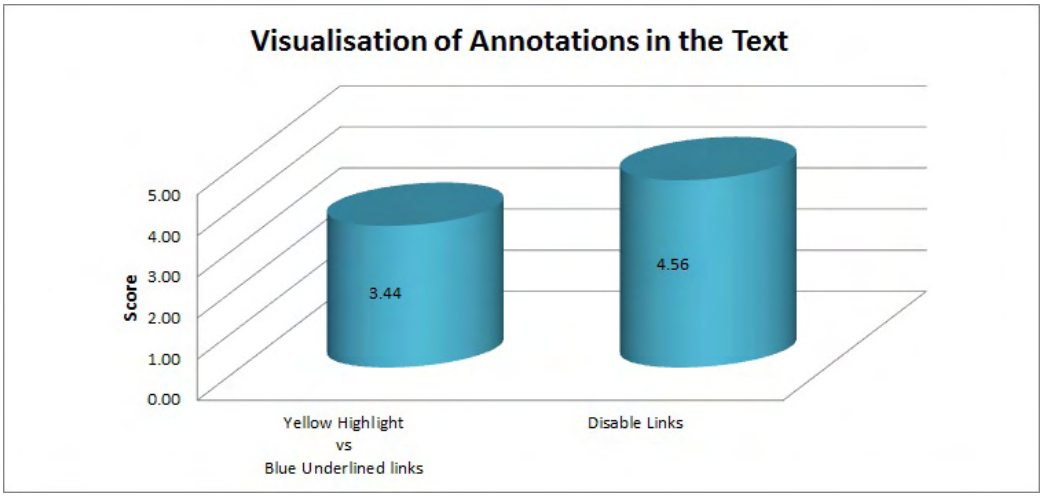

Figure 6.3: The average score for the Visualisation in the Text

In our Wysinwis viewer we introduced a Bubble Chart to give an overview of the tags used on the page. In Figure 6.4 the graph mentions the results concerning the overall impression on visualisation in the form of bubbles, the usefulness and if users would base their opinion on the visualisation to determine that the page contains the information they are looking for.

As shown in the graph (Figure 6.4), the participants would rely on the Bubble Chart to decide whether or not they would read (or even open) the page. We will discuss this further in the Qualitative Evaluation-section (Figure 6.3).

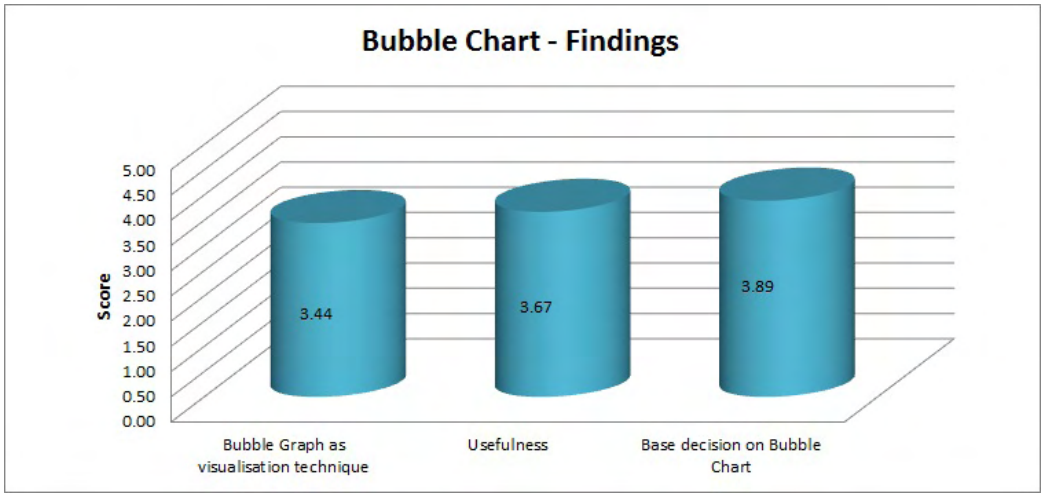

Figure 6.4: The average scores for the findings on the Bubble Chart

Wysinwis idea and Filtering-techniques

In Figure 6.5 we used the average scores for the questions about Wysinwis. The first bar mentions: "in what way the users think that the problem of the Web treating everybody the same should be resolved". A score of 3 out of 5 shows that the participants are aware of the problem and that they prefer to have it solved. The second bar (Figure 6.5) shows to what extent users believe that the Wysinwis idea increases the usability of the Web. The bar on the right indicates to what extent the participants think that the Wysinwis principle makes it easier to find information on the Web. The high scores show that the Wysinwis principle is not without potential.

In Figure 6.6 the user friendliness of the clickable bubble chart is portrayed. With this clickable bubble chart the user can filter the visualisation of annotations and links on the page. With a score of 3.78 it would seem that this type of filtering-technique, along with the Wysinwis principle can offer some assistance with the filtering of annotations.

6.3 Qualitative Evaluation

During the interview we asked open questions about the Wysinwis viewer. This interview was held with the participants individually. We asked if they had some remarks or enhancements for our Wysinwis viewer, and about the Wysinwis concept. In this section we will discuss their answers together with their general feeling about the ideas.

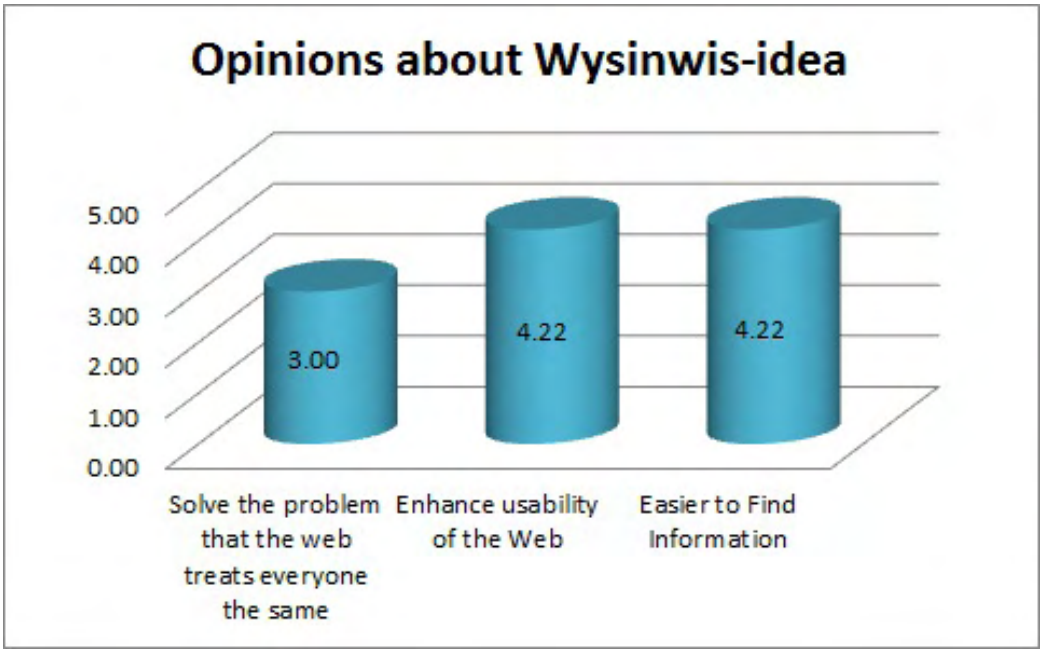

Figure 6.5: The average scores for the Wysinwis idea, and to what extent they think this approach will increase the usability of the Web

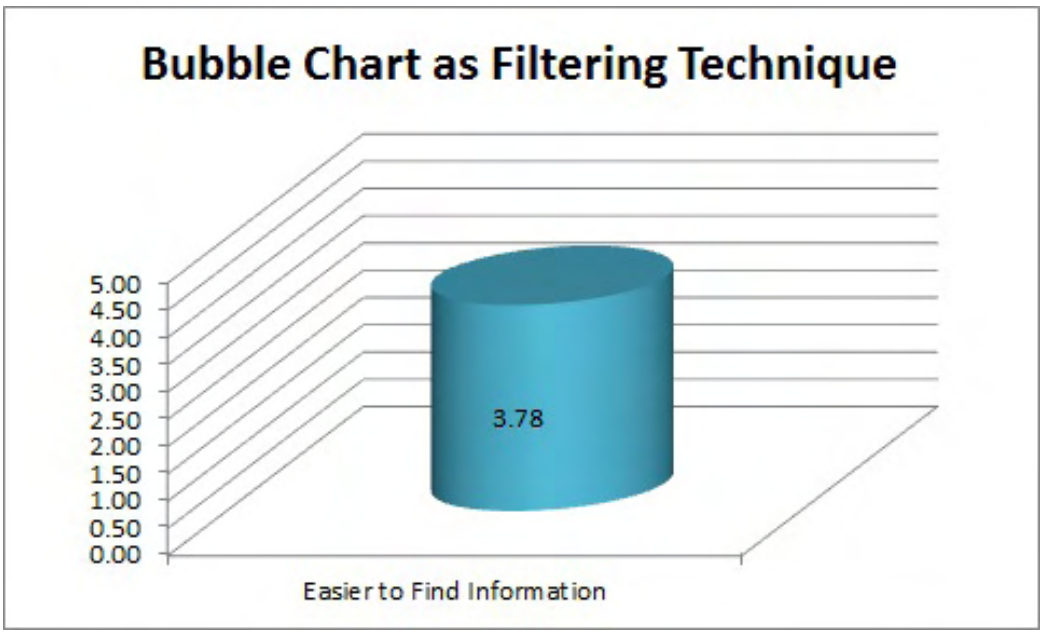

Figure 6.6: Shows to what extent filtering is user-friendly by means of the Bubble Chart

Most of the participants were pleased to get an overview of the page by means of the bubble diagramme. Although one person preferred a bar chart over a bubble diagramme because he found it more clear to see the most important tag. Another person made the comment that it would be better to omit less important tags. The problem with that is that the bubble chart does not give a complete overview anymore, but this is something we have to investigate further. Most of the people would rely on the Bubble Chart to decide whether or not to read the page.

The link diagramme was very popular for more technical people that use their computer to search for answers on a daily bases. The reason for that was the possibility to filter the links on link type and file type. Someone mentioned that he sees some problems concerning privacy when we introduce the availability-check for a link. On the other hand, all the other participants were pleased to have the availability check. To please persons who fear privacy issues, we could foresee a setting to disable the availability-check.

Three of the participants found it cumbersome to open the link visualisation and the in-text bubble chart. When we presented the Wysinwis viewer to the participants, we forgot to mention that we specified shortcuts to open the visualisations.

All the participants agreed that text would be more readable when there are less annotations and links on a page. Everyone was enthusiastic about the feature to disable all the annotations and links. It was far more clear and readable than the normal visualisation of the page. The highlights were not that popular because they were too notable. Some of the participants preferred a lighter shade of yellow. We have to consider to change the hue of the yellow. It is a pity that Weinreich [50] and Obendorf [31] didn't mention the RGB-values they used for their study.

Most of the participants found it a good idea to emphasise some parts of the text based on a profile, and to omit other annotations and links. They tested the Wysinwis viewer with a fixed profile, but they see a lot of potential when the Wysinwis viewer would be elaborated with the dynamic profile. Two persons mentioned during the interview that when a dynamic profile would be added to the Wysinwis viewer, they would change their way of using search engines. One of them gave the following example: "When I'm looking for the birth date of a celebrity, nowadays, I just use the name of that person as search criteria. Then I look for a page like wikipedia, and I search for the birth date on that page. When a dynamic profile would be in place in the Wysinwis viewer, I would add 'birth date' to my search criteria. In that way, the birth date will be emphasised when I open a page, and I can find the date more easily."

Most of the participants told me at the end of the interview spontaneously that they hope it will be investigated further and that it will be adopted by the most important webbrowsers.

6.4 Conclusion of the Evaluation

As a conclusion we can say that the participants of the evaluation were positive about the Wysinwis principle, and the visualisations we used. It would be a good idea to elaborate our Wysinwis viewer further with some extra features. The most important one is the dynamic profile, and the connection with social media.

All the participants see the benefit in a system like Wysinwis. For leisure they will not enable the functionality, but when they are looking for information or answers on the Web they see the gain in both time and user-friendliness.

7

Conclusions and Future Work

7.1 Conclusions

There are multiple ways to create hyperlinks and annotations on the Web. Web developers are able to create hyperlinks and add metadata to their pages. Furthermore, web users can create annotations and hyperlinks by using third-party annotation tools. We realised that for many reasons, there is still room for improving the visualisation of hyperlinks and annotations on the Web. These reasons are as follows:

- The web browsers treat all web users the same, regardless of their preferences or interests. Not every user is looking for the same information. Some users search for entertainment and others might be interested in scientific articles or information about their hobbies. However, there is no difference between these users when they are browisng the Web; only one visualisation is available for all.

- Apart from readability issues caused by the blue underlined visualisation of hyperlinks, it also neglects many important aspects that should be considered. For example, users are not aware of the target of the hyperlink (e.g. protocol, file type or status of the webpages).

- Web browsers neglect the metadata attached to webpages and their

hyperlinks and annotations. This metadata can definitely be exploited to deliver better visualisation of the content. Moreover, it can be used for content adaptation.

- When a lot of annotations exist on a webpage that page becomes chaotic and in some cases even unreadable. Existing filtering mechanisms are not sufficient to enable the users to maintain a clear overview of what they are looking for.

In this thesis we came up with a solution to solve the problems stated before. This solution, described in Chapter 4, is called the Wysinwis principle. Wysinwis stands for What You See Is Not What I See. By allowing users to create a profile to indicate their preferences, the visualisation of a webpage will be rendered differently in favour of the user. Moreover, hyperlinks and annotations are visualised based on user profiles. Some hyperlinks and annotations are emphasised while others are hidden. Furthermore, users can filter the annotations and can search for relevant annotations. To enhance the readability, we allow users to disable the visualisation of the annotations and hyperlinks. In that way, only plain text without any distractions remains, with no distractions in the text. An important characteristic of our solution is that we take into account the annotations created by annotators as well as those created by authors. The Wysinwis principle also describes some visualisation techniques to give users an overview of a page in terms of hyperlinks and content.

We developed a Google Chrome extension based on the Wysinwis principle. We used this tool to evaluate the Wysinwis principle. The participants were positive about the idea and they hope it will be adopted by major browsers.

7.2 Future Work

From our evaluation of the Wysinwis principle we got a lot of positive feedback. However, we have conducted a limited evaluation in a non-collaborative setting. Hence, a more elaborated study can be carried out in a collaborative setting.

The Wysinwis system itself can also be extended to increase a user's participation in enriching the Web with more metadata. A system for gratification [37] can be added to the Wysinwis system to increase the collaboration. Moreover, a grading system can enhance the Wysinwis filtering mechanism.

A
Appendix

Evaluation of the WYSINWIS-Viewer

Welcome to this questionnaire about the Wysinwis-viewer. Thank you for answering every question.
This questionnaire will be the starting point for an interview afterwards.

About you

1. **Your name:** _____
2. **How old are you?** _____
3. **Gender**
   - □ Male
   - □ Female

General Questions

4. **Had you already heard about the Semantic Web before the Wysinwis-demo?**
   □ Yes □ No
5. **Had you already used an annotation tool?**
   □ Yes □ No
6. **Do you see the benefit in using the semantics of the web by working with a tool such as the Wysinwis-viewer?**
   0 □—□—□—□—□ 5
7. **When you use a search engine and open a page from the search result: can you find the information you are looking for easily?**
   0 □—□—□—□—□ 5
8. **Do you agree that when information based on your search criteria would be emphasized on the pages of your search result, it would be easier to find the information?**
   0 □—□—□—□—□ 5
9. **Would you use the Wysinwis-viewer?**
   No □—□—□—□—□ Yes
10. **Would you recommend the use of the Wysinwis-viewer?**
    No □—□—□—□—□ Yes
11. **Do you plan on adding annotations and links to pages when you use the Wysinwis-viewer?**
    No □—□—□—□—□ Yes
12. **I found the Wysinwis-viewer easy to use...**
    0 □—□—□—□—□ 5
13. **Do you think the Wysinwis-viewer is easy to use without getting a demo?**
    0 □—□—□—□—□ 5
14. **The Wysinwis-viewer can help me browse the web...**
    0 □—□—□—□—□ 5

Questions about Visualisation

15. **Do you think the visualisation of hyperlinks is less distracting when using the Wysinwis-style of links (yellow highlight instead of blue and underlined links)?**
    0 □—□—□—□—□ 5
16. **You can disable all links and annotations on a page. Do you agree that the page is more readable?**
    0 □—□—□—□—□ 5
17. **The tag visualisation on page level and on fragment level are both done by using a bubble chart. Do you think this visualisation technique is a good way to visualise the tags?**
    0 □—□—□—□—□ 5

18. **In the Bubble Chart on page level, the tags used on the page are shown. On a scale of 0 to 5, to what extent do you think this visualisation is useful?**
    0 □—□—□—□—□ 5
19. **Based on the tags on the bubble chart on page level, do you think you can make out if the page contains the information you're looking for?**
    0 □—□—□—□—□ 5
20. **Do you think the tooltip-preview with the extra information about a link is an added value (Availability + protocoltype + link-type?**
    0 □—□—□—□—□ 5
21. **Do you think links with multiple destinations are an added value for webpages?**
    0 □—□—□—□—□ 5
22. **Do you agree that multiple destination links can help improve the structuring of the webpage?**
    0 □—□—□—□—□ 5
23. **As a user, when a link is a multi-destination link, there's an extra step (hoover and click instead of only clicking the link). Do you prefer multiple links instead of one link with multiple destinations?**
    No □—□—□—□—□ Yes

Wysinwis-idea and filtering

24. **The web treats everyone the same, do you think this has to be solved?**
    No □—□—□—□—□ Yes
25. **The web treats everyone the same, the Wysinwis-principle uses a user profile (whether or not fixed). Does the Wysinwis-priciple make the web more user friendly?**
    No □—□—□—□—□ Yes
26. **Do you think emphasized fragments based on a profile will enhance the usability?**
    No □—□—□—□—□ Yes
27. **Do you agree: It is easier to find the information I'm looking for when I use the Wysinwis-viewer.**
    No □—□—□—□—□ Yes
28. **Via the clickable Bubble Chart the page's visualisation can be adapted (filtered) to show only the selected tags. Do you think it is a user-friendly way of filtering the annotations?**
    No □—□—□—□—□ Yes
29. **The link-overview shows all the links on a page. Do you think it would be more interesting to see more levels of links, or would it be more confusing?** □ Interesting □ Confusing

Only for authors of webpages

30. **Authors of webpages have to add semantics to their links, and to some fragments of the text. Do you see the benefit for the user?**
    No □—□—□—□—□ Yes
31. **Authors of webpages have to add semantics to their links, and some fragments of the text. Are the benefits for the user in proportion with the augmented work for the authors?**
    No □—□—□—□—□ Yes

Enhancements

32. **Do you use Social Media?**
    □ Yes □ No
33. **Would you be encouraged to use the tool when we provide a link with Facebook or other social media? An annotation can be placed on your Facebook.**
    No □—□—□—□—□ Yes

34. **The goal of the Wysinwis-viewer is also to have the possibility of a dynamic profile, based on the previous search criteria. Will it enhance the usability?**
No □—□—□—□—□ Yes
35. **On pages, such as Facebook, you don't want to see Annotations. Would it be interesting to have a list of pages where the Wysinwis-viewer will be deactivated all the time? (black list)**
No □—□—□—□—□ Yes
36. **On the web, we can divide links into two categories. Structural links (i.e. menu structure) which have no semantic link with the document, and on the other hand, there are Semantic Links that do have a semantic meaning. We now allow filtering the links on linktype, document type and protocol. Would it be better to allow filtering on semantic and non-semantic links?**
No □—□—□—□—□ Yes
37. **Do you think it would be better to make a visual distinction (e.g. other background-color) between both types of links?**
No □—□—□—□—□ Yes
38. **Would it be a good idea to grade users, to sort annotations?**
No □—□—□—□—□ Yes
39. **Would it be a good idea to grade the annotation itself, to sort the annotations?**
No □—□—□—□—□ Yes

Bibliography

[1] W. Amaya-team. Welcome to Amaya - Amaya: Overview, 2012. `http://www.w3.org/Amaya/Overview.html`.

[2] K. Andrews, F. Kappe, and H. Maurer. The Hyper-G Network Information System. *Journal of Universal Computer Science (J. UCS)*, 1(4):206–220, 1995.

[3] H. Berghel. Who Won the Mosaic War? *Communications of the ACM*, 41(10):13–16, 1998.

[4] T. Berners-Lee. Information Management: A Proposal, march 1989. `http://www.w3.org/History/1989/proposal.html`.

[5] T. Berners-Lee. HTML Tags, march 1990-1992. `http://www.w3.org/History/19921103-hypertext/hypertext/WWW/MarkUp/Tags.html`.

[6] M. Bernstein. Structural Patterns and Hypertext Rhetoric. *ACM Computing Surveys*, 31(4es):19, 1999.

[7] P. J. Brown. Turning Ideas into Products: The Guide System. In J. B. Smith and F. G. Halasz, editors, *Proceedings of the ACM Conference on Hypertext*, HYPERTEXT '87, pages 33–40, New York, NY, USA, 1987. ACM.

[8] V. Bush. As We May Think. *Atlantic Monthly*, 176(1):641–649, 1945.

[9] V. J. Caracelli and J. C. Greene. Crafting Mixed-Method Evaluation Designs. *New directions for evaluation*, 1997(74):19–32, 1997.

[10] L. A. Carr, D. C. DeRoure, W. Hall, and G. J. Hill. The Distributed Link Service: A Tool for Publishers, Authors and Readers. *World Wide Web Journal*, 1(1):647–656, 1995.

[11] N. Delisle and M. Schwartz. Neptune: A Hypertext System for CAD Applications. In C. Zaniolo, editor, *ACM SIGMOD Record*, volume 15, pages 132–143. ACM Press, New York, 1986.

[12] L. Denoue and L. Vignollet. An Annotation Tool for Web Browsers and its Applications to Information Retrieval. In J.-J. Mariani and D. Harman, editors, *In Proceedings of RIAO 2000, Sixth Conference on Content Based Multimedia Information Access*, pages 180–195. CID, 2000.

[13] Diigo Inc. Diigo, 2014. `https://edshelf.com/tool/diigo`.

[14] Diigoteam. About Diigo, 2014. `https://www.diigo.com/about`.

[15] G. W. Furnas, C. Fake, L. von Ahn, J. Schachter, S. Golder, K. Fox, M. Davis, C. Marlow, and M. Naaman. Why Do Tagging Systems Work? In R. J. Gary Olson, editor, *CHI '06 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '06, pages 36–39, New York, NY, USA, 2006. ACM.

[16] K. Graves. A Comparison of Link Readability Techniques. *20th Computer Science Seminar*, 20:SD3–T2–1, 2004.

[17] J. C. Greene, V. J. Caracelli, and W. F. Graham. Toward a Conceptual Framework for Mixed-Method Evaluation Designs. *Educational evaluation and policy analysis*, 11(3):255–274, 1989.

[18] M. Gutfeldt. Browsers with LINK Support, 2000-2001. `http://gutfeldt.ch/matthias/translation/LINK/ENaddendum.html`.

[19] S. Hartenstein. Creating Questionnaires with LaTeX, 2010. `http://www.svenhartenstein.de/creating-questionnaires-with-latex/`.

[20] P. Heymann, D. Ramage, and H. Garcia-Molina. Social Tag Prediction. In S.-H. Myaeng, D. W. Oard, F. Sebastiani, T.-S. Chua, and M.-K. Leong, editors, *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 531–538, New York, NY, USA, 2008. ACM.

[21] D. B. Judd and G. Wyszecki. *Color in Business, Science, and Industry.* Pure and Applied Optics Series. Wiley, 1975.

[22] J. Kahan and M.-R. Koivunen. Annotea: An Open RDF Infrastructure for Shared Web Annotations. *Computer Networks*, 39(5):589, 2002.

[23] P. Kahn and K. Lenk. Design: Principles of Typography for User Interface Design. *Interactions*, 5(6):15–, 1998.

[24] M. R. Koivunen. Annotea and Semantic Web Supported Collaboration. In *Invited talk at Workshop on User Aspects of the Semantic Web (UserSWeb) at European Semantic Web Conference*, pages 5–16, 2005.

[25] M.-R. Koivunen. Semantic Authoring by Tagging with Annotea Social Bookmarks and Topics. In I. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, editors, *In The 5th International Semantic Web Conference (ISWC2006) - 1st Semantic Authoring and Annotation Workshop (SAAW2006)*, SAAW2006, 2006.

[26] G. Li, V. Uren, E. Motta, S. B. Shum, and J. Domingue. ClaiMaker: Weaving a Semantic Web of Research Papers. *Lecture Notes in Computer Science*, 1:436–441, 2002.

[27] J. N. Lythgoe. *The Ecology of Vision.* Clarendon Press, 1979.

[28] E. Mandinach. The Development of Effective Evaluation Methods for e-Learning: A Concept Paper and Action Plan. *The Teachers College Record*, 107(8):1814–1836, 2005.

[29] C. C. Marshall and A. J. B. Brush. Exploring the Relationship Between Personal and Public Annotations. In H. Chen, H. D. Wactlar, C. chih Chen, E.-P. Lim, and M. G. Christel, editors, *Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '04, pages 349–357, New York, NY, USA, 2004. ACM.

[30] J. Nanard and M. Nanard. Should Anchors Be Typed Too?: An Experiment with MacWeb. In S. E. Poltrock, editor, *Proceedings of the Fifth ACM Conference on Hypertext*, HYPERTEXT '93, pages 51–62, New York, NY, USA, 1993. ACM.

[31] H. Obendorf and H. Weinreich. Comparing Link Marker Visualization Techniques: Changes in Reading Behavior. In N. E. Mastorakis, V. Mladenov, Z. Bojkovic, D. Simian, S. Kartalopoulos, A. Varonides, C. Udriste, E. Kindler, S. Narayanan, J. L. Mauri, H. Parsiani, and K. L. Man, editors, *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 736–745, New York, NY, USA, 2003. ACM.

[32] U. Ogbuji. Make your HTML Pages Smarter with RDFa 1.1 Lite, may 2012. `http://www.ibm.com/developerworks/xml/library/wa-rdfalite/index.html?ca=dat`.

[33] B. Shneiderman, C. Plaisant, R. Botafogo, D. Hopkins, and W. Weiland. Designing to facilitate browsing: A look back at the Hyperties workstation browser. *Hypermedia*, 3:101–117, 1998.

[34] S. B. Shum, E. Motta, and J. Domingue. ScholOnto: An Ontology-Based Digital Library Server for Research Documents and Discourse. *International Journal on Digital Libraries*, 3(3):237–248, 2000.

[35] B. Signer. What is Wrong with Digital Documents? A Conceptual Model for Structural Cross-Media Content Composition and Reuse. In J. Parsons, M. Saeki, P. Shoval, C. C. Woo, and Y. Wand, editors, *Proceedings of the 29th International Conference on Conceptual Modeling*, LNCS, pages 391–404. Springer, 2010.

[36] B. Signer and M. C. Norrie. As We May Link: A General Metamodel for Hypermedia Systems. In C. Parent, K.-D. Schewe, V. C. Storey, and B. Thalheim, editors, *Conceptual Modeling-ER 2007: 26th International Conference on Conceptual Modeling, Auckland, New Zealand, November 5-9, 2007, Proceedings*, volume 4801 of *Lecture Notes in Computer Science*, pages 359–374. Springer, 2007.

[37] H. Takeda and I. Ohmukai. Building Semantic Web Applications as Information and Knowledge Sharing Systems. In *Proceedings of End User Aspects of the Semantic Web*, 2005.

[38] V. Uren, S. B. Shum, G. Li, J. Domingue, and E. Motta. Scholarly Publishing and Argument in Hyperspace. 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*, WWW '03, pages 244–250, New York, NY, USA, 2003. ACM.

[39] T. Vander Wal. Folksonomy, 2007. `http://www.vanderwal.net/essays/051130/folksonomy.pdf`.

[40] W3C. HTML5 Reference, august 2010. `http://dev.w3.org/html5/html-author/`.

[41] W3C. Differences from HTML4, may 2013. `http://www.w3.org/TR/html5-diff/\#new-apis`.

[42] W3C. RDFa 1.1 Primer - Second Edition - Rich Structured Data Markup for Web Documents. `http://www.w3.org/TR/xhtml-rdfa-primer/`, 2013. `http://www.w3.org/TR/xhtml-rdfa-primer/`.

[43] W3C. RDF 1.1 Primer. `http://www.w3.org/TR/rdf11-primer/`, 2014. `http://www.w3.org/TR/rdf11-primer/`.

[44] W3C. RDF Schema 1.1. `http://www.w3.org/TR/rdf-schema/`, 2014. `http://www.w3.org/TR/rdf-schema/`.

[45] W3C. Representing Knowledge in the Semantic Web - slide "The Semantic Web stack", 2014. `http://www.w3c.it/talks/2005/openCulture/slide7-0.html`.

[46] W3Schools. jQuery Introduction, 2014. `http://www.w3schools.com/jquery/jquery_intro.asp`.

[47] J. H. Walker. Document Examiner: Delivery Interface for Hypertext Documents. In J. B. Smith and F. G. Halasz, editors, *Proceedings of the ACM Conference on Hypertext*, HYPERTEXT '87, pages 307–323, New York, NY, USA, 1987. ACM.

[48] T. Wang, L. Wu, and Z. Lin. The Revival of Mozilla in the Browser War Against Internet Explorer. In Q. Li and T.-P. Liang, editors, *Proceedings of the 7th international conference on Electronic commerce*, ACM International Conference Proceeding Series, pages 159–166. ACM, 2005.

[49] H. Weinreich and W. Lamersdorf. Concepts for Improved Visualization of Web Link Attributes. *Computer Networks*, 33(1-6):403–416, 2000.

[50] H. Weinreich, H. Obendorf, and W. Lamersdorf. The Look of the Link - Concepts for the User Interface of Extended Hyperlinks. In K. Grønbæk, H. C. Davis, and Y. Douglas, editors, *Proceedings of the 12th ACM Conference on Hypertext and Hypermedia*, HYPERTEXT '01, pages 19–28, New York, NY, USA, 2001. ACM.