FAKULTÄT
FÜR !NFORMATIK

Faculty of Informatics

File Format Analysis

Monitoring the Life Cycle of File Formats in the Internet

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering & Internet Computing

eingereicht von

Stefan Schindler
Matrikelnummer 0726559

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Andreas Rauber, Univ. Doz.
Mitwirkung: Dr. Christoph Becker

Wien, 12.03.2014

(Unterschrift Verfasser)

(Unterschrift Betreuung)

FAKULTÄT
FÜR !NFORMATIK

Faculty of Informatics

File Format Analysis

Monitoring the Life Cycle of File Formats in the Internet

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering & Internet Computing

by

Stefan Schindler
Registration Number 0726559

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Andreas Rauber, Univ. Doz.
Assistance: Dr. Christoph Becker

Vienna, 12.03.2014

(Signature of Author)

(Signature of Advisor)

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.ac.at

Erklärung zur Verfassung der Arbeit

Stefan Schindler
Lobstrasse 25, 3121 Karlstetten

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

(Ort, Datum) (Unterschrift Verfasser)

Acknowledgements

I would like to thank Andreas Rauber for supervising my thesis, and for his ongoing support and motivation.
I thank Christoph Becker for giving me valuable feedback and help through the entire developing process of this thesis.
Furthermore, I'd like to thank, in lexical order, Krešimir Đuretec, Artur Kulmukhametov and Petar Petrov for their valuable hints and suggestions.
At last, I would like to thank my family and my friends for their great support throughout the years of my study. You rock!

Stefan Schindler, February 2014

Abstract

In the last years the amount of information, that is published and shared in the world wide web, increased rapidly. The web became one of the main platforms for education, communication and entertainment, but also for business processes. As published web data is very ephemeral and can be changed very easily, long-term accessibility is prevented, so it is necessary to take actions to archive these data for later use and prevent this loss of information (this process is called „web archiving“).

As the world wide web grew, many new file formats and versions have been developed, that found their way into the web usage. This leads to the risk, that data is available in older formats, that are no more readable and interpretable by newer versions of a program. This software obsolescence represents a big issue for digital objects. Another information, that is mainly important for developers of programs, that are able to create certain file formats, but also for users of these created formats, is the time it takes for a file format to disappear or to be replaced by its successor.

From these exigences, the following questions can be derived, that can be seen as typical for developers as well as users: (1) To what extend has a newly invented file format been accepted? (2) How long did it take for a specific version to replace its predecessor? (3) When will a specific version or a certain file format become obsolete?

This thesis creates a framework for the identification and analysis from files loaded from the web, and also has the ability to compute extensive statistics of the development of particular versions of certain file formats over a longer period. Therefore, different tools are used and created, that also allow future adaptation and the adding of new functionalities to the framework. As an example, an extra in-depth analysis of HyperText Markup Language files, that make a big part of www-data is performed and specific characteristics of these files are illustrated. It is possible to crawl certain (sub-)domains of the world wide web, to filter the created *warc* files, and to identify the individual files with assistance of the *File Information Tool Set*. The created identification files are aggregated and analyzed with a statistical computing program. The produced framework and the generated statistics are used to try giving an insight in the evolution of the world wide web, and to highlight patterns and trends that can be used to support digital preservation and to discover potential preservation risks.

The thesis compares the findings with already performed studies of this subject area, and gives an overview of further steps and studies to be conducted.

Kurzfassung

Die Menge an Informationen, die im World Wide Web veröffentlicht wird, ist in den letzten Jahren rapide angestiegen. Heutzutage ist das Internet eines der meistgenutzten Medien für Bildung, Kommunikation, Unterhaltung und auch Geschäftsprozesse. Da publizierte Daten im Internet sehr leicht veränderbar sind, und sich dadurch auch in sehr kurzen Zeitabständen ändern, ist es notwendig, rechtzeitig Maßnahmen zu ergreifen, um diese Daten für eine spätere Verwendung zu archivieren (dieser Prozess wird „Web Archiving“ genannt). Es werden nicht nur ständig neue Informationen im WWW publiziert, auch die verwendeten Dateiformate werden ständig erweitert, verbessert und durch neuere ersetzt. Dadurch besteht auch die Gefahr, dass Daten in älteren Formaten vorliegen, die durch neuere Versionen eines Programmes nicht mehr korrekt gelesen und angezeigt werden können. Diese Softwareobsoleszenz stellt eine große Gefahr für digitale Objekte dar. Eine weitere Information, die hauptsächlich für Entwickler von Programmen, die bestimmte Dateiformate erzeugen, aber auch für Anwender, die mit diesen erzeugten Formaten arbeiten, oder diese weiterverwenden, wichtig sein kann, ist die Information darüber, wie lange es dauert, bis ein Dateiformat verschwindet, bzw. von seinem Nachfolger abgelöst wird.

Aus diesen Anforderungen leiten sich unter anderem folgend typische Fragen ab, die sowohl für Entwickler, als auch für Anweder essentiell sind: (1) Inwieweit wurde ein neu eingeführtes Dateiformat akzeptiert? (2) Wie lange dauerte es für eine bestimmte Version, ihren Vorgänger zu ersetzen? (3) Wann wird eine bestimmte Version oder ein bestimmtes Dateiformat obsolet?

Diese Arbeit erzeugt ein Framework zur Identifikation und Analyse von aus dem Web geladenen Dateien, und ist in der Lage, umfangreiche Statistiken über die Entwicklung einzelner Versionen von bestimmten Dateiformaten über einen längeren Zeitraum zu berechnen. Dafür werden verschiedene Tools verwendet und erzeugt, die es auch ermöglichen, das Framework zukünftig noch weiter adaptieren und diesem Funktionalität hinzufügen zu können. Beispielsweise werden in der Arbeit zusätzlich auch HyperText Markup Language-Dateien, die einen großen Anteil der Dateien im WWW ausmachen, im Detail analysiert, und spezifische Eigenschaften aufgezeigt. Bestimmte (Sub-)Domains des WWW können gecrawled werden, die erzeugten *warc*-Dateien werden ausgelesen, und die einzelnen Dateien mit Hilfe des *File Information Tool Set* identifiziert. Die erzeugten Identifikationsdateien werden aggregiert und mit einem Statistikprogramm analysiert. Mit dem erzeugten Framework und den dadurch generierten Statistiken wird versucht, einen Einblick in die Evolution des WWW, und Trends und Patterns aufzuzeigen, die dabei helfen können, Digital Preservation zu unterstützen und Risiken aufzudecken.

In der Arbeit werden die gefundenen Erkenntnisse mit bereits durchgeführten Studien zu diesem Themenbereich verglichen, und eine Übersicht über zukünftige, weiterführende Schritte, und durchzuführende Forschungen gegeben.

Contents

CHAPTER 1

Introduction

In the last years the amount of information that is shared in the world wide web, increased rapidly. The web became one of the main platforms for education, communication and entertainment, but also for business processes. Digital information created and managed by different institutions is becoming more important for the long term

Due to the nature of the web, the published information is very ephemeral, which prevents long-term accessibility. Ntoulas et. al. showed, that nearly 80% of the web pages are updated or disappear within one year, losing the former information forever [44].

To prevent this loss of information, different initiatives tried to preserve parts of the web, to make them accessible for future dates. This process, called web archiving, stems from multisided drivers. These include institutional policy, legal requirements or research interests [36].

Digital objects are vulnerable to software obsolescence, because standards for encoding, representation and retrieval can be renewed and software to encode and represent data emerges by time. Digital Preservation copes with such software and hardware obsolescence related issues. It aims at maintaining digital objects authentically usable and accessible for long time periods.

As the world wide web grew, many new file formats and versions have been developed, that found their way into the web usage. Developers, as well as users, may ask the question, how these new formats and versions have been accepted. This fact can be determined by the evolution of their frequencies in the web.

Typical questions for developers and users of this formats and versions may encompass:

- To which extent has a particular newly invented format been accepted?
- How long did it take for a version to supersede its predecessor?
- When will a specific file format or version become obsolete?
- Is it necessary to support HTML tags that have been developed a long time ago?
- Which formats will be used primarily in the next years?

All these questions are related to the temporal change in the distribution of sizes and features, aiming to detect existing risks and find patterns and draw conclusions that may help to identify these preservation risks.

A possible solution to detect requested distributions is the large-scale analysis of web archives. If these archives are analysed in terms of the crawling time, the evolution over a period of time can be computed and future trends may be estimated.

This task is very challenging, as there is often not enough data available to analyze. Most snapshots cover one instant of time and can't be used for long-term analysis. Furthermore, many crawls don't provide sufficient information for an adequate analysis.

This thesis tries to give insights into the evolution of the world wide web, that can lead to increased insight into trends and patterns and thus improves digital preservation. It copes with the challenges and tries to answer the above outlined questions. It analyzes the distribution of the web and tries to give the opportunity to find preservation risks. Therefore different collections of regularly archived parts of the web will be analyzed in terms of different properties, like used formats, versions and encodings. A main focus is on HTML files, which make a large part of the web, and their internal structure. The thesis outlines the results and examines, if predictions into the future can be given.

1.1 Structure of the Work

1. Chapter 1, Introduction gives an introduction to the scope of this thesis, followed by a quick overview of the single chapters contents.
2. Chapter 2, Related work gives a short introduction on the main principles, that are used during this thesis (digital preservation, web archiving), as well as a description of web crawlers and an estimation on the current size of the world wide web. The chapter also discusses some related work that has already been published in terms of file type distribution and HTML tag analysis.
3. Chapter 3, Methodology, describes the methods that were used to fulfill the required tasks, in detail. It explains the work-flow of the file format analysis toolset, as well as all programs and outputs that were used and created within and by using this toolset.
4. Chapter 4, Results, shows some results that were obtained during the analysis of a large dataset using the file format analysis toolset. These results are displayed graphically, as well as described in the continuous text.
5. Chapter 5, Discussion and Conclusion, discusses the generated results and compares them to other studies on this topic. Furthermore, some interesting findings, that have been made, are displayed and an outlook for possible future work is given.

CHAPTER 2

Related work

This chapter gives an introduction on the related work for the core topics of this thesis. This covers insights into digital preservation and web archiving, as well as studies on the distribution of files and file types in the world wide web, and information on the life cycle of file formats and the distribution of tags in HTML files.

2.1 Digital Preservation

The goal of digital preservation is to maintain digital objects accessible in an authentic manner for a long term into the future. This fact comes with lots of different challenges, that have to be taken into account. As digital data can be highly complex, the meaning of its underlying bit stream is not understandable without further knowledge of the containing environments. Digital objects require specific environment to be accessible: files need specific programs to be displayed, programs need specific operating systems and operating systems need specific hardware components. As these environments are not stable, it happens, that files or programs cannot be opened any more, which leads to a loss of information. [17]

In 2003 the NSF-DELOS Working Group on Digital Archiving and Preservation specified 5 conditions for preservation, each of them providing benefits to society:

> „if unique information objects that are vulnerable and sensitive and therefore subject to risks can be *preserved and protected*; if preservation ensures long-term *accessibility* for researchers and the public; if preservation fosters the *accountability* of governments and organizations; if there is an economic or societal advantage in *re-using information*, or if there is a *legal requirement* to keep it.“ [28]

So there are lots of institutions that have to deal with the described digital preservation challenges, from governments and companies to libraries, schools, universities and private persons.

2.2 Web Archiving

This thesis focuses on web archiving, the preservation of content that is published online. The large scale of the content requires highly scalable analytics, its ephemerality and the constant increase of information are other factors that have to be taken into account when trying to preserve online content. There are multi sided causes to archive the internet. A very big task is to use web archiving to store and preserve information, that would be lost through the convertibility of the web, or to make them accessible to not only future generations, but also to users in 2 or 4 years. Ntoulas et al. showed, that nearly 80% of the web pages are updated or disappear within one year, losing the former information forever [44].

Additionally, institutions try to answer the question, how specific websites change in the course of time (years, months, weeks). In 2005, Hackett and Parmanto from the School of Health and Rehabilitation Sciences, University of Pittsburgh, USA, analyzed the change of usability of higher education websites [25]. For this, they used the *Wayback Machine*[1], a service of the Internet Archive[2] and Alexa Internet[3]. This tool analyzes and archives websites in unsteady intervals and provides these snapshots to the end user.

For example, the domain `http://www.tuwien.ac.at` has been crawled 602 times [4] by the Wayback machine, going back to May 25th, 1997. Indeed, there exist many domains that have been crawled much less, and not every piece of information had been preserved, but for „common“ pages, usually lots of snapshots exist. Figures 2.1 and 2.2 show the first and last crawl of the domain by the wayback machine. These figures show the immense difference in web technologies between 14 years.

A completely new usage for archived web data is the analysis of the development of file formats and versions over time, that is part of this thesis. With this method, statistical facts from the past can be derived, that lead to possible predictions for future format trends.

Web crawlers

Archiving information from the World Wide Web is a subset of digital preservation. As the size of the web exceeds the size of any existing library thousandfold, a new approach has to be achieved. Manual preservation techniques can not be considered as such an approach, since they are extremely time-consuming and would, therefore, not pay off. The web is too big and there is too much information that changes too often. Every minute old pages are modified or deleted and new pages are added.

The world wide web can be seen as a system of referencing websites. It is a directed graph with nodes representing websites and edges representing links between websites.

To archive the web, automatic tools for content gathering, called *web crawlers* are used. These programs can navigate between web pages using hyperlinks, like humans do when navigating the web.

Menczer (2007) defines a web crawler as follows:

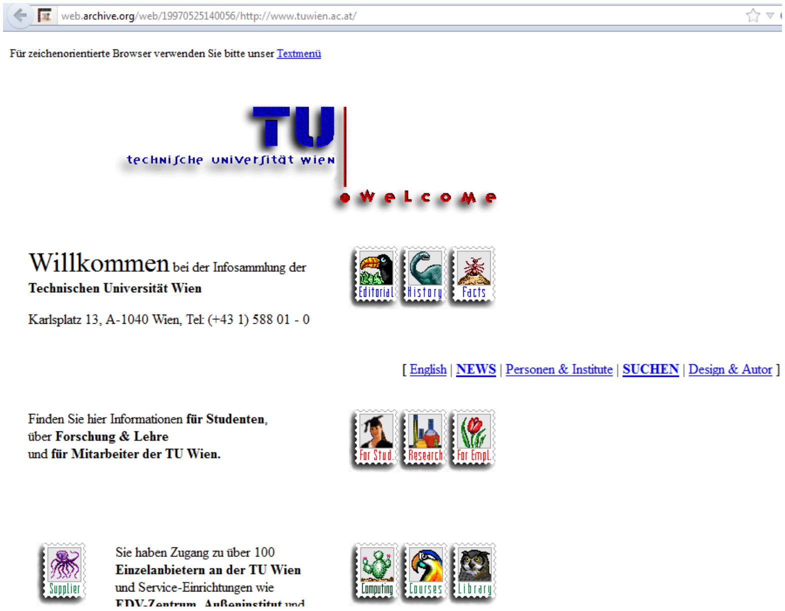
---

[1] `http://archive.org/web/web.php`

[2] `http://www.archive.org`

[3] `http://www.alexa.com`

[4] dating April 26th, 2012


web.archive.org/web/19970525140056/http://www.tuwien.ac.at/

Für zeichenorientierte Browser verwenden Sie bitte unser Textmenü

Willkommen bei der Infosammlung der
Technischen Universität Wien

Karlsplatz 13, A-1040 Wien, Tel: (+43 1) 588 01 - 0

[English | **NEWS** | Personen & Institute | **SUCHEN** | Design & Autor]

Finden Sie hier Informationen **für Studenten**,
über **Forschung & Lehre**
und **für Mitarbeiter der TU Wien.**

Sie haben Zugang zu über 100
Einzelanbietern an der TU Wien
und Service-Einrichtungen wie
EDV-Zentrum, Außeninstitut und

Figure 2.1: Homepage of `www.tuwien.ac.at` on May 25th, 1997. This was the first crawl of this domain by the wayback machine.

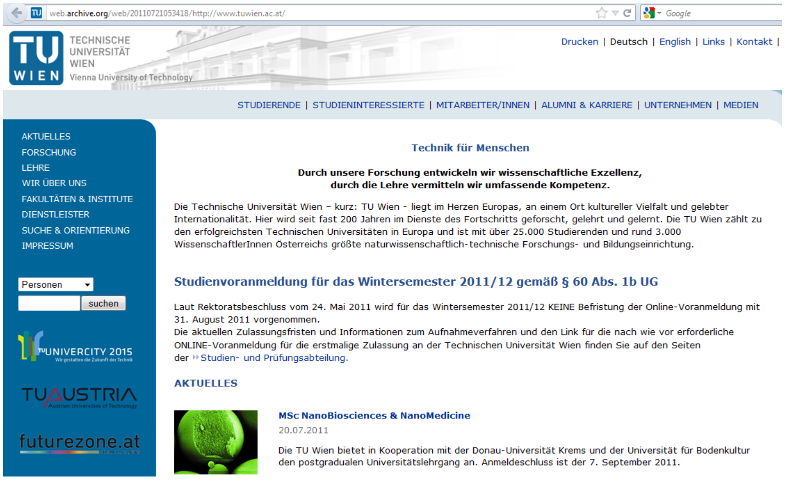
> „Web crawlers are programs that exploit the graph structure of the Web to move from page to page. From the beginning, a key motivation for designing Web crawlers has been to retrieve Web pages and add them or their representations to a local repository. Such a repository may then serve particular application needs such as those of a Web search engine. In its simplest form a crawler starts from a *seed* page and then uses the external links within it to attend to other pages. The process repeats with the new pages offering more external links to follow, until a sufficient number of pages are identified or some higher level objective is reached.“ [40]

So a web crawler is a program, that starts with a set of pages and navigates to other pages that are linked within the content. All information gathered is saved to disk or a database. But crawling the web is not as easy in terms of bounds and shaping. Traditional books and papers have a fixed, finite number of pages, that can be used to circumvent individual documents. Websites on the other hand do not follow these restrictions. They represent a loosely coupled set of documents, that are related in different ways. So flat linked files are not enough for the created information space, it also has to consist out of active information systems [17].

Many different approaches to web crawlers exist, but a basic sequential crawler, that is shown in Figure 2.3, will be explained in the next paragraphs.

The crawler starts with a defined *seed*, typically is a small set of URIs, and then follows the edges to find new nodes.

One of the most crucial tools is the *frontier*. It is responsible for selecting the next URI to be crawled. Therefore, it holds a list with the already parsed URIs, as well as the discovered URIs and has to ensure, that no URI gets crawled more than once. The frontier is always in account


web.archive.org/web/20110721053418/http://www.tuwien.ac.at/
Google
TU WIEN
TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology
Drucken | Deutsch | English | Links | Kontakt |
STUDIERENDE | STUDIENINTERESSIERTE | MITARBEITER/INNEN | ALUMNI & KARRIERE | UNTERNEHMEN | MEDIEN
AKTUELLES
FORSCHUNG
LEHRE
WIR ÜBER UNS
FAKULTÄTEN & INSTITUTE
DIENSTLEISTER
SUCHE & ORIENTIERUNG
IMPRESSUM
Personen
suchen
TU UNIVERCITY 2015
Wir gestalten die Zukunft der Technik
TU AUSTRIA
Austrian Universities of Technology
futurezone.at

Technik für Menschen

**Durch unsere Forschung entwickeln wir wissenschaftliche Exzellenz,
durch die Lehre vermitteln wir umfassende Kompetenz.**

Die Technische Universität Wien – kurz: TU Wien - liegt im Herzen Europas, an einem Ort kultureller Vielfalt und gelebter Internationalität. Hier wird seit fast 200 Jahren im Dienste des Fortschritts geforscht, gelehrt und gelernt. Die TU Wien zählt zu den erfolgreichsten Technischen Universitäten in Europa und ist mit über 25.000 Studierenden und rund 3.000 WissenschaftlerInnen Österreichs größte naturwissenschaftlich-technische Forschungs- und Bildungseinrichtung.

Studienvoranmeldung für das Wintersemester 2011/12 gemäß § 60 Abs. 1b UG

Laut Rektoratsbeschluss vom 24. Mai 2011 wird für das Wintersemester 2011/12 KEINE Befristung der Online-Voranmeldung mit 31. August 2011 vorgenommen.
Die aktuellen Zulassungsfristen und Informationen zum Aufnahmeverfahren und den Link für die nach wie vor erforderliche ONLINE-Voranmeldung für die erstmalige Zulassung an der Technischen Universität Wien finden Sie auf den Seiten der ›› Studien- und Prüfungsabteilung.

AKTUELLES

MSc NanoBiosciences & NanoMedicine
20.07.2011
Die TU Wien bietet in Kooperation mit der Donau-Universität Krems und der Universität für Bodenkultur den postgradualen Universitätslehrgang an. Anmeldeschluss ist der 7. September 2011.

Figure 2.2: Homepage of `www.tuwien.ac.at` on July 21st, 2011. This was the last crawl of this domain so far.

for maintaining politeness, so that no web server is crawled to heavily. As long as uncrawled URIs are available in the frontier's list, it passes one to the *downloader*. Should there be no more forthcoming URIs, the frontier terminates and the crawler exits successfully. [26]

The *downloader* receives URIs from the frontier and tries to load them from the web server. Therefore, the downloader has to act as a HTTP client, which sends a HTTP request and receives the response. Depending on the responses HTTP status code, the downloader processes the response and submits it to the *repository* and the *parser*. Other meta data in the HTTP header can also be relevant for future treatment and is therefore also saved. [46]

An exemplary request for the site `http://www.tuwien.ac.at/index.html` could be like this:

```
GET /index.html HTTP/1.1
Host: www.tuwien.ac.at
```

If the web server finds the site, it responses with the document, including some meta data, that are necessary for saving the files into the repository, like content-length and -type, and the last date, the site was modified:

```
HTTP/1.1 200 OK
Date: Fri, 27 Apr 2012 10:26:01 GMT
Server: Apache/2.2.3 (CentOS)
Last-Modified: Thu, 10 Jun 2010 13:56:46 GMT
Content-Length: 5008
Content-Type: text/html; charset=UTF-8
...
```

Status code 200 means, the request has succeeded and the requested information will be returned. Other notable status codes, that must be taken into account, are:

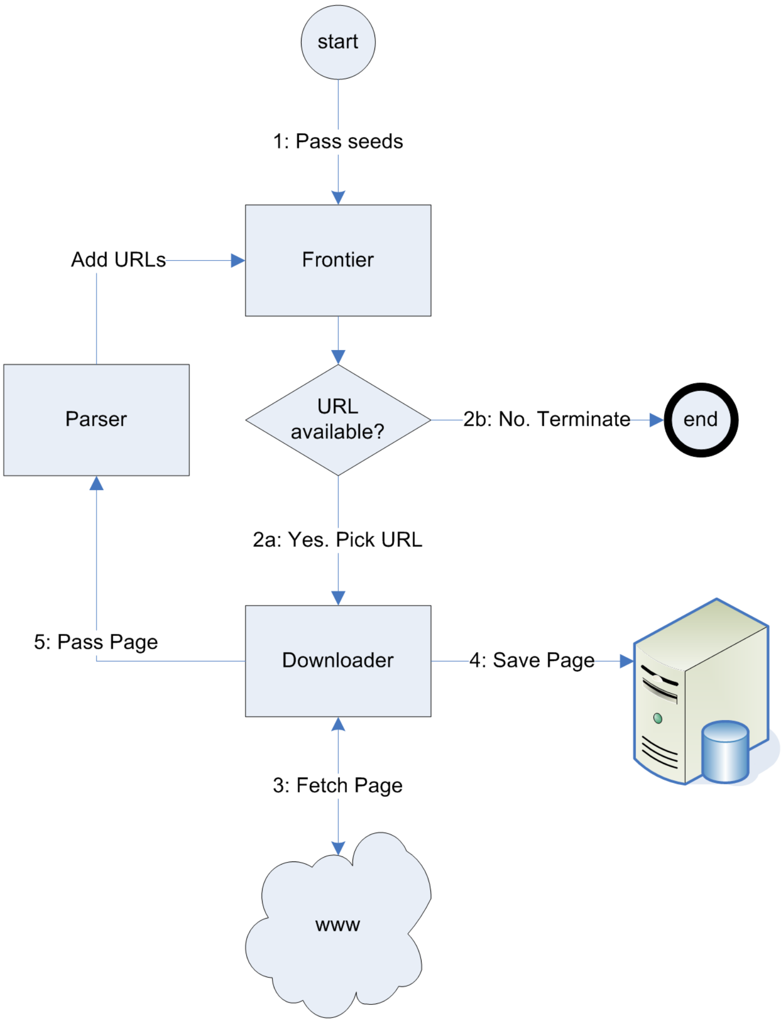


Figure 2.3: A basic sequential web crawler

- **301** Moved Permanently: The site hast moved permanently to an other location; all requests should be directed to the responded URI
- **401** Unauthorized: The request requires user authentication, that was not yet established
- **404** Not Found: The requested URI could not be found by the server. [19]

For building the repository, that receives the downloaded files, different strategies exist: Local file system served archives, web-served archives and non-web archives (for a detailed discussion of these strategies, please refer to the work of Deegan (2006), p. 85ff.).

Finally, the parser tries to fetch new URIs out of the content of the downloaded files. Parsing is a very complex discipline and requires, for instance, the usage of regular expressions or xmlparsers. New URIs, that are created by the parser, are then added to the appropriate frontiers list.

Heritrix

An example for an open source web crawler is the Internet Archive's *Heritrix 3.0*[5], that is written in Java. Heritrix can be accessed via a web interface, where crawl jobs can be configured and all kinds of reports are displayed.

The crawler is highly adaptable, nearly all parts can be extended or replaced by other implementations. It works after the following scheme:

> „
>
> 1. Choose a URI from among all those scheduled
> 2. Fetch that URI
> 3. Analyze or archive the results
> 4. Select discovered URIs of interest, and add to those scheduled
> 5. Note that the URI is done and repeat
>
> "

[41]

The Size of the Web

The web is obviously very big, but, due to the dynamic nature of the web, it is impossible to determine exactly, how much information is available in the web. But there exist some estimations that lead to an analysis, how the size of the web has changed within the last 15 years.

Basically, the web can be divided into two parts: the so-called *surface web*, that can be crawled by search engines like Google[6] and the *deep web* or *invisible web*, that is not indexable by standard search engines, because it is saved in databases and only contained in dynamically generated sites. Noor et al. are defining the deep web like this:

> „The data lies in deep web cannot be crawled and indexed by conventional web search engines. Information underlying deep web sites can only be accessed through their own query interfaces and results are produced dynamically in response to a direct request." [43]

The usage of databases, that contain necessary information and can be be queried dynamically by server sided scripts, instead of presenting the data on static HTML pages, led to a very big increase of the deep web.

There have been some studies about the size of the indexable web. In 1998, using data from December 1997, Lawrence and Giles estimated the total size of the indexable web with

---

[5] `https://webarchive.jira.com/wiki/display/Heritrix/Heritrix`

[6] `http://www.google.com`

at least 320 million pages. [37]. A newer analysis from February 1999 updated this size to 800 million documents. [38] To estimate these numbers, they analyzed the overlaps between different search engines. Overlapping, the accepted technique for measuring the size of the indexable web depending on the relative size of search engines, was introduced by Bharat and Border in 1998 [8].

Bergman used this data for his overlap analysis of the deep web in 2001, based on data collected in March 2000, where he found some important results, by comparing the deep and surface web [5]:

- Public information on the deep Web is currently 400 to 550 times larger than the commonly defined World Wide Web.
- The deep Web contains 7,500 terabytes of information compared to nineteen terabytes of information in the surface Web.
- The deep Web contains nearly 550 billion individual documents compared to the one billion of the surface Web.
- The deep Web is the largest growing category of new information on the Internet.
- More than half of the deep Web content resides in topic-specific databases.
- Original Deep Content Now Exceeds All Printed Global Content.

In 2005, Gulli and Signorini estimated the size of the public indexable web, by also using overlap analysis, with approximately 11.5 billion pages. [24]. De Kunder estimated the number of distinct web pages indexed by the 4 main search engines (Google, Yahoo!, MSN Search, Ask.com) to at least 14.3 billion in 2006 [15].

De Kunder also created a website[7], that estimates the size of the web every day since a few years, also using overlapping analysis, where the number of indexed web pages is about 37 billion at April 17th, 2012.

Figure 2.4 shows an approximation for the massive increase of pages in the indexable web based on the discussed estimations. As the web has grown nearly linear until 2000, it received an enormous growth since than, that can also be predicted for the next years.

A report by the IDC (International Data Corporation) [8] from 2011 states, that the amount of information created and replicated in the so called *digital universe* will grow by a factor of 9 in just five years [22].

2.3 File Type Distribution

In 1996, Arlitt and Williamson [2] collected the access logs of 6 web servers (Department of Computer Science in the University of Waterloo, Department of Computer Science in the University of Calgary, University of Saskatchewan, NASA's Kennedy Space Center, the web server

---

[7] `http://www.worldwidewebsize.com/`

[8] `http://www.idc.com/`

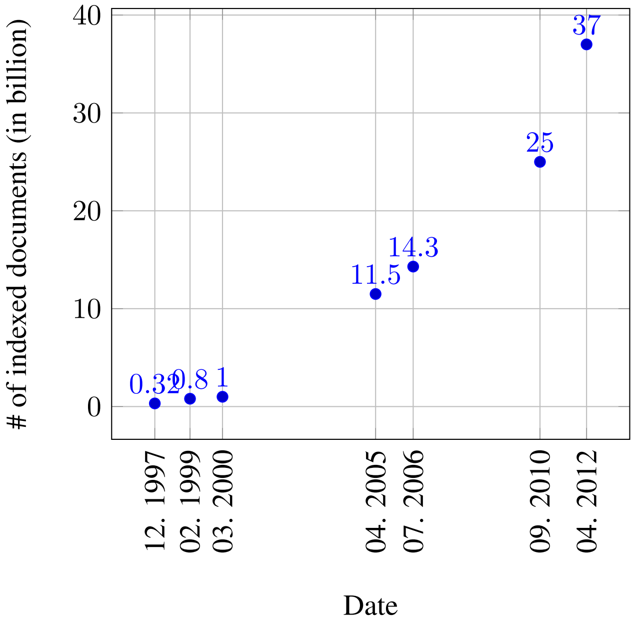


Figure 2.4: The trend of unique pages in the indexable web, derived from different studies. [37], [38], [5], [24], [15]

from ClarkNet, a commercial internet provider, and the web server at the National Center for Supercomputing Applications (NCSA) in Urbana-Champaign, Illinois).

The log duration differed from 1 week (NCSA), to 1 year (Calgary), whereas the total requests differed from more than 3 million over 2 months (NASA) to only 160.000 over 8 months (Waterloo). Arlitt and Williamson analyzed the successfully retrieved files from these web servers and split them into different document types. They used the following generic categories: HTML, Images, Sound, Video, Formatted, Dynamic and Other. Figure 2.5 shows the accumulated results over all 6 web servers.

HTML and Images sum up to nearly 95% of the whole requests to the web servers, while all other types (like audio, video and zipped content) was hardly requested at all.

Woodruff et al. collected 2.6 million unique HTML documents in 1996 and analyzed a variety of properties of these documents, including Document Size, Tag Usage, Attribute Usage and the number of In-Links. [58]

In 2004, Nanavati et al. tried to extend this study and compare the results to examine, how the properties have changed over time. They crawled an accurate 1-million-sample of the 8 billion web pages that Google claimed to index at this time, by using the Stanford WebBase project [33], that gave a list of starting points for a representative crawl. [42]

They analyzed all file types that were linked or embedded in the crawled documents; their results are displayed in figure 2.6. All linked files were grouped by their file extension, which leads to a good, but not quite accurate overview, how the occurrence of different file types has changed over the years. As file extensions aren't significant, and can be changed at will without

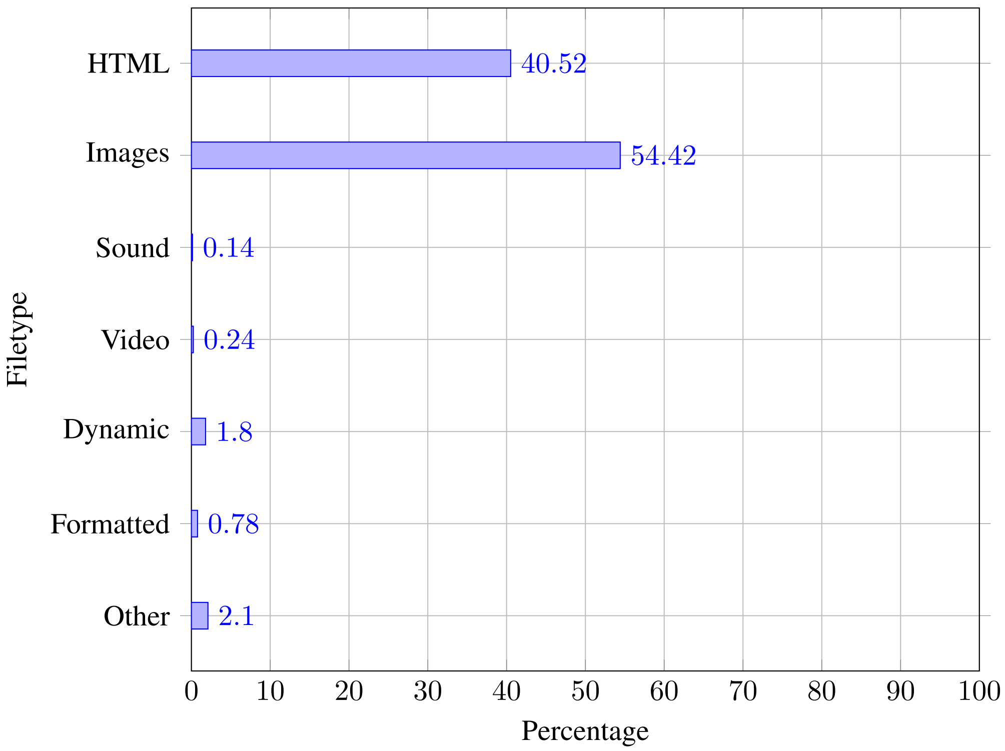


Figure 2.5: Percentage of Filetypes for all 6 web servers [2]

changing the actual file format, this analysis isn't quite accurate, but can give an approximation to the file format distribution.

The Category *Compression/Archive* consists of the files with extensions zip, z, rar, gz, gzip, taz, tgz. *Document* contains all kinds of text files, like txt, eps, doc, pdf, dvi. *Audio* files are mainly au, wav, mp3, and rm, whereas *Video* files have the extensions avi, mpeg/mpe/mpg and mov. Finally, the images are split into *GIF* and *OtherImages* (jpeg/jpg, tif/tiff, bmp).

Some interesting facts can be revealed by these results. In 2004 more than 90% of the web pages had a reference to a gif image, in 1996 it were at least over 60%. The usage of movie and audio files increased slightly, a comparison among the other contents shows a quite big increase.

(X)HTML Distribution

In 2012, Jackson analyzed the distribution of different versions in a dataset released by the UK Web Archive[9].

> „Working with JISC, the UK Web Archive has obtained a copy of all the web resources in the Internet Archive that are from the .uk domain, or embedded within .uk web pages. This contains some 2.5 billion HTTP 200 responses stretching

---

[9] `http://www.webarchive.org.uk/`

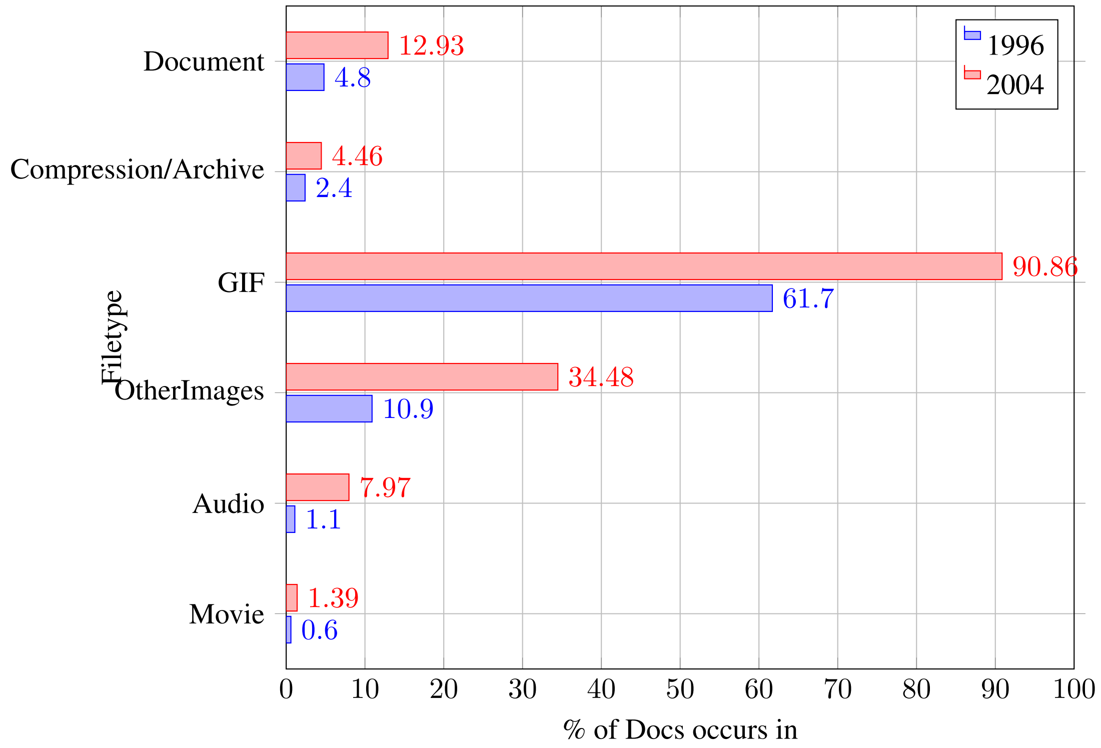


Figure 2.6: Comparison of Documents containing specific File Types, 1996 [58] and 2004 [42]

> from 1996 to 2010, neatly packed into ARC files and stored on our HDFS cluster. And seeing as it's HDFS, this means we can run Map-Reduce tasks over the whole dataset, and analyze the results.“ [35]

Jackson analyzed the mediatypes for HTML, that were output from Apache Tika and DROID and was able to create a chart that shows the usage of the different HTML and XHTML versions over 15 years. His findings are displayed in Figure 2.7.

In this dataset, HTML 2.0 was the leading format in 1996, but decreased very fast to less than 10% in 2000. In general, each newly arising format was dominant for a few years and did then decrease and gave way for newer formats. Jackson also mentioned, that in the last crawled year (2010) still all formats were found (HTML 2.0, HTML 3.2, HTML 4.0, HTML 4.01, XHTML 1.0, XHTML 1.1) [35].

2.4 HTML Tag Analysis

The results of Woodruff et al. (1996) and Nanavati et al. (2004) can also be used to analyze the usage of different tags in their crawled HTML documents. They counted the number of documents, where the different tags occurred and computed a percentage for the top 20 tags.

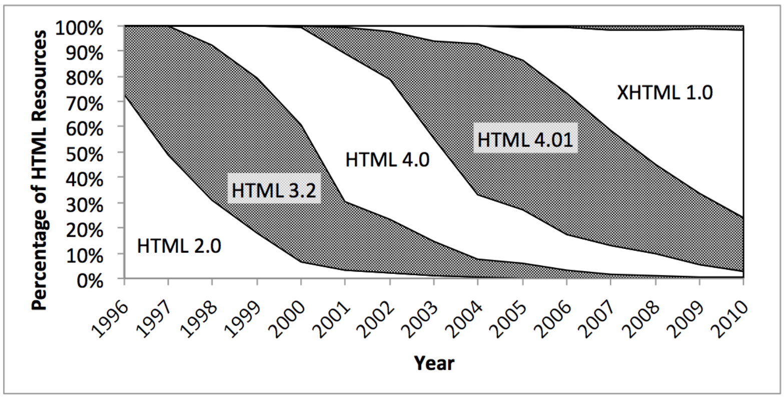


Figure 2.7: HTML Distribution of the UK Web Archive found by Jackson [35]

Figure 2.8 shows the evolution of Documents containing specific Tags. While in 1996, the `<html>`, `<head>` and `<body>` elements were used in only 60 - 70% of the documents, their usage has increased dramatically to nearly 100% in 2004. The `<div>` tag was never used in 1996, because it was introduced with HTML 3.2, that was published as a W3C (World Wide Web Consortium[10]) Recommendation in 1997. [51]

On the contrary, the usage of the header tags (`<h1>`, `<h2>`, `<h3>`, `<h4>`) dropped by nearly 50%, which could be explained by the introduction of Cascading Style Sheets (CSS) by the W3C in December 1996.

2.5 Summary

This chapter showed some related work for this master thesis. An observation that has been made, is, that crawling is possible in various occurrences. A lot of different *web crawlers* exist, that can index and save parts of the web. Further, the creation of statistics (regarding the distribution of different generic types or versions of a file format) is possible, as Arlitt and Williamson [2], Woodruff et. al. [58] or Nanavati et. al. [42] showed.

But there isn't any work available, that performs in-depth analysis over a longer time series. Also, most analysis were done based on the extensions of linked files, which is not accurate, as extensions must not say anything about the content of the files. So, the identification has not been performed by specific tools for this purpose, which could lead to possible misinterpretations. The identification by file extension is not necessarily meaningful, as will be explained in the next chapters, so it is possible, that these results do not reflect the „correct" distribution of file formats.

The goal of this thesis is to close the discovered gaps and try to find possibilities to predict the life cycle of of newly released (and also of already existing) versions of file formats. Therefore a part of the web shall be analyzed, not focusing on a specific media type (like, for instance,

---

[10] `http://www.w3.org/`

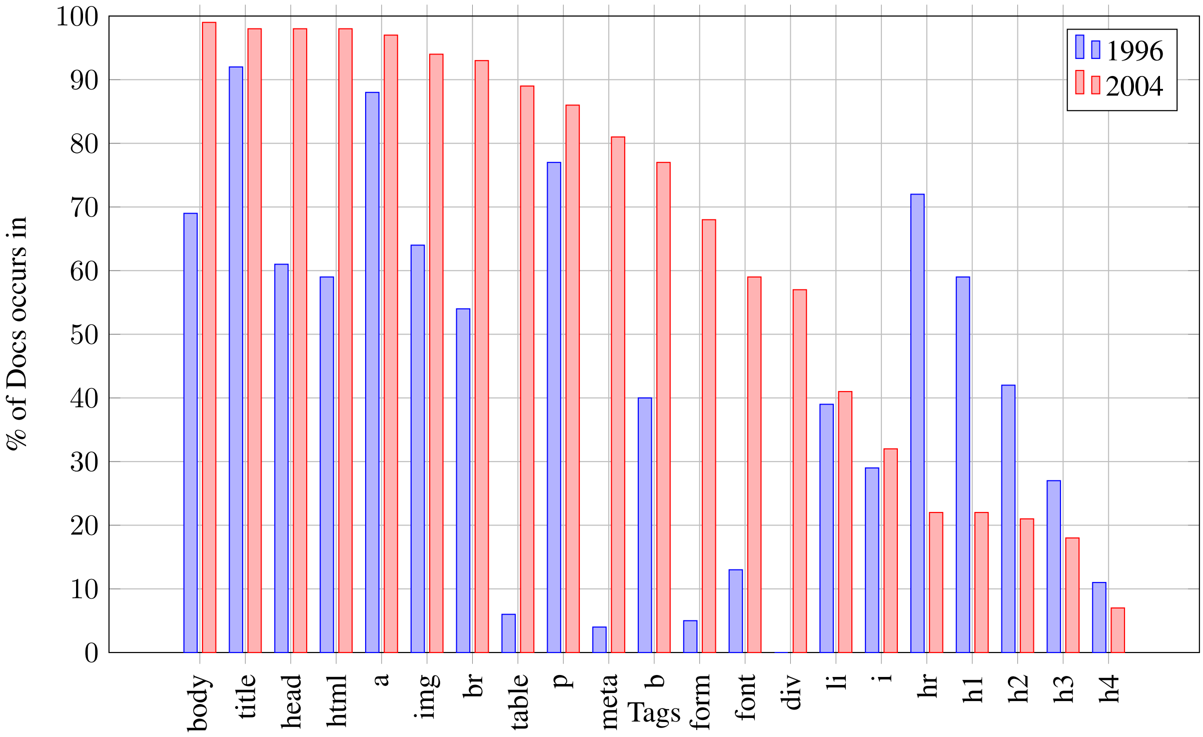


Figure 2.8: Comparison of Documents containing specific Tags, 1996 [58] and 2004 [42]

Jackson [35] did), or a generic main media type, but on all available media types and their corresponding versions.

CHAPTER 3

Methodology

As Chapter 2 showed, crawling the web and computing differences between file formats or specific file format versions over different points in time has been performed and discussed in several studies. But none of these researches cover all parts (longer time series, valid identification, sufficient granularity) to present the distribution of different versions of specific file formats over a couple of years.

The goal of this thesis is to analyze a part of the web in depth, not focusing on a specific media type (like, for instance, Jackson [35] did), or a generic main media type, but on all available media types and their corresponding versions. To create the highest possible validity, and not rely on file extensions or some other possibly misleading factors, tools for the identification of the single files shall be used, that can be found as open-source in the world wide web. Through this analysis it could be possible to predict future file format obsolescence and offer valuable clues on how long outdated versions of file formats need to be supported by newer generations of tools.

This chapter gives a detailed description of the used concepts, as well as the methods and models that were used to perform the required experiments. It outlines the basic architectures of file format analyzing, as well as the different attempts of identifying the format. Furthermore, the concrete approach, that is used within this thesis, is explained in detail.

3.1 Overview

Figure 3.1 presents an overview on the analysis framework, that was created and used during the research for this masters thesis.

At first it is necessary to crawl one or more URIs, to collect data that can be used to analyze and compare the specific parameters. Therefore, a web crawler (see Section 2.2 for more details) is used. The crawled files, including some metadata, are saved in *.warc* or *.warc.gz* format.

Afterwards, the created WARC files are processed with a WARC reader. Depending on the metadata - only *response* records (these have been sent *from* the server *to* the crawler) contain the necessary information - the reader discards the files or extracts them from the WARC file.

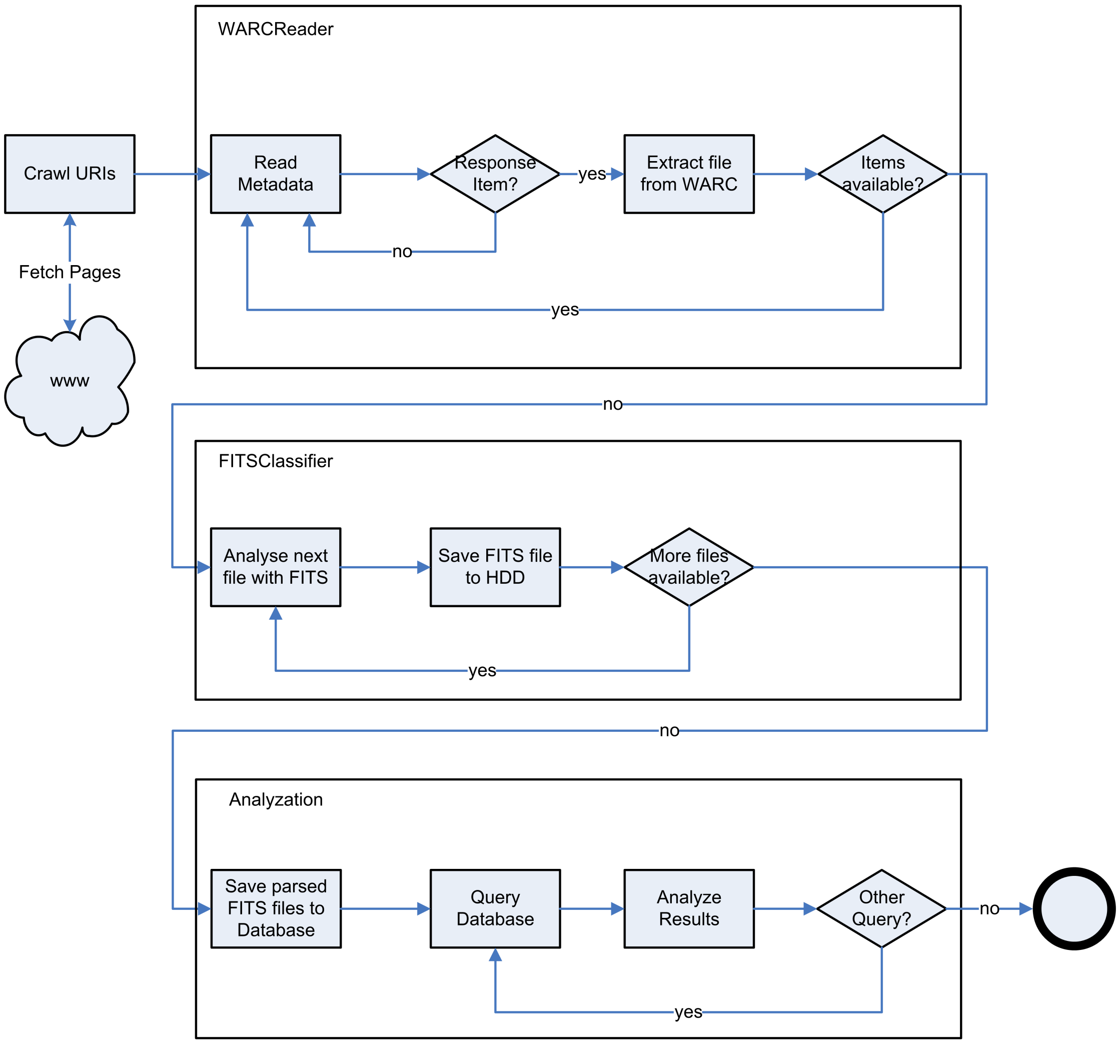


Figure 3.1: Overview of the Analysis Framework

When all files are treated, they are passed to the FITS classifier. This tool identifies and characterizes the files by various factors and saves the resulting metadata in the XML file format.

Afterwards the parameters of all characterized files are saved to a database, so they are available for multiple processes.

Finally, different analyses are performed, that can easily be extended to answer new or updated research questions.

3.2 The Crawler

To crawl the web, for this thesis, a web crawler was set up to gain test data. The Internet Archive's open-source web crawler *Heritrix 3.0*[1] was used for this case. Heritrix is a very powerful, multi threaded crawler, which allows the parallel fetching and analysis of many URIs at a time and leads to a faster discovery of the passed URIs.

The archived files were saved in gzipped WARC format, which will be explained in Section 3.3.

As a much bigger set was provided for analysis (as described in Section 4.1), the crawled test data were only used for testing the functionalities of the other framework components, and do not find their way into the results of this thesis.

3.3 The WARC Reader

Before explaining the function of the WARC reader, an explanation of the used WARC format has to be given.

The WARC Format

WARC, the Web ARChive file format is a container format for storing web crawls, that is widely used. It specifies a method for combining multiple digital resources into an archival file, together with related information. The format is an extension of the Internet Archive's ARC[2] format, that was introduced in 1996. After a few revisions, the WARC file format was approved as an international standard in May 2009 (ISO 28500:2009).

> „The WARC (Web ARChive) file format offers a convention for concatenating multiple resource records (data objects), each consisting of a set of simple text headers and an arbitrary data block into one long file.“ [34]

So, depending, how and when the files are split by the crawler, a WARC file might reach sizes up to 1GB. Figure 3.2 shows the schematic layout of a WARC file.

Each WARC record has a specific type, that identifies the following metadata and content block. Some of these types are *warcinfo*, that is usually the first record in a WARC file, *response* and *request*. For a detailed discussion of the WARC record types, please refer to [34].

WARC records are order-independent. To identify related records (such as a HTTP request and the corresponding HTTP response), each WARC record has a globally unique *WARC-Record-ID*. This unique ID also allows the management of migrated versions of records, which can be added to WARC files at later dates. [56]

Listing 3.1 shows a warcinfo record. This record is usually the first in a WARC file and identifies, amongst others, the used crawler and the WARC file format.

---

[1] `https://webarchive.jira.com/wiki/display/Heritrix/Heritrix`

[2] `http://www.digitalpreservation.gov/formats/fdd/fdd000235.shtml`

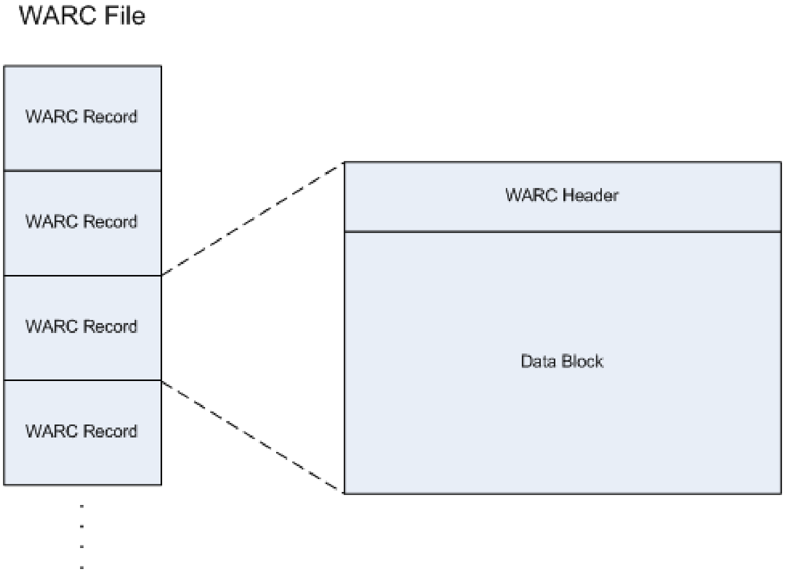


Figure 3.2: Format of a WARC File

```
WARC/1.0
WARC-Type: warcinfo
WARC-Date: 2011-11-17T11:34:23Z
WARC-Filename: WEB-20111117113421008-00000-2976~10.0.0.3~8443.warc.gz
WARC-Record-ID: <urn:uuid:16b2e8b0-7b71-4beb-8f3b-408b5f3c64af>
Content-Type: application/warc-fields
Content-Length: 394

software: Heritrix/3.0 http://crawler.archive.org
ip: 10.0.0.3
hostname: 10.0.0.3
format: WARC File Format 1.0
conformsTo: http://bibnum.bnf.fr/WARC/WARC\_ISO\_28500\_version1\_latestdraft.pdf
isPartOf: basic
description: Basic crawl starting with useful defaults
robots: obey
http-header-user-agent: Mozilla/5.0 (compatible; heritrix/3.0 +http://www.tuwien.ac.at)
```

Listing 3.1: A WARC Warcinfo Record

The next Listings, 3.2, 3.3 and 3.4 show 3 related WARC records. As said before, the order of these records in the WARC file does not matter. They a related by the *WARC-Record-ID* and *WARC-Concurrent-To*, that are highlighted in the example (the unique identifier **urn:uuid:65c2b218-a409-48df-b214-949d9f525121**). The WARC request record (Listing 3.2) shows a simple HTTP-GET request from the crawler to a web server[3]. The request record contains the Target-URI, that is to be retrieved (in that case, the index-page at `http://www.informatik.tuwien.ac.at`, the request date and time, as well as the content

---

[3] For a detailed discussion of HTTP request and response records, please refer to the HTTP 1.0 specification [6]

length of the data block, that contains the HTTP request. The field *WARC-Concurrent-To* links to the *WARC-Record-URI* of the WARC record with the corresponding server response (Listing 3.3). This WARC response record again contains some header fields, like the requested URI and the related IP address, as well as the length of the content that was returned from the server. The data block contains the HTTP response header and the content that was actually retrieved from the server, in this case a HTML document.

Additionally, a third WARC record is related with this response. It is a WARC metadata record, like displayed in Listing 3.4. This record contains important information for the crawler, like the time it has taken to retrieve the document, and the outgoing links that have been filtered from the HTML document. These links are passed to the frontier, and will later be used for crawling and discovering new files.

```
WARC/1.0
WARC-Type: request
WARC-Target-URI: http://www.informatik.tuwien.ac.at/
WARC-Date: 2011-11-17T11:34:29Z
WARC-Concurrent-To: <urn:uuid:65c2b218-a409-48df-b214-949d9f525121>
WARC-Record-ID: <urn:uuid:54cdbbac-2e9c-46f8-8130-cfd91ad62cb1>
Content-Type: application/http; msgtype=request
Content-Length: 280

GET / HTTP/1.0
User-Agent: Mozilla/5.0 (compatible; heritrix/3.0 +http://www.tuwien.ac.at)
Connection: close
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Host: www.informatik.tuwien.ac.at
Cookie: cstrack=80.123.33.69.1321529661763495
```

Listing 3.2: A WARC Request Record

Extraction of the Files

As explained in the previous sections, the archived files have to be extracted from the WARC file. For the extraction some helper files were used, that have been obtained from the Lemur Project[4]. This is a collaboration between the Center for Intelligent Information Retrieval at the University of Massachusetts Amherst[5] and the Language Technologies Institute at Carnegie Mellon University[6], that created an open source framework for building information retrieval and language modeling software.

These files are used for retrieving single WARC records from the specified WARC file. At first, the *WARC-Type* of the received WARC record is checked, because only responses contain the necessary information in the body (that are the actual files, that were retrieved from the web server and that can be used for classification). Because the WARC record still contains a header, this header is removed, as well as the HTTP response header, so only the raw data of the retrieved object is kept.

---

[4] `http://www.lemurproject.org/`

[5] `http://ciir.cs.umass.edu/`

[6] `http://www.lti.cs.cmu.edu/`

```
WARC/1.0
WARC-Type: response
WARC-Target-URI: http://www.informatik.tuwien.ac.at/
WARC-Date: 2011-11-17T11:34:29Z
WARC-Payload-Digest: sha1:TKFXIV5ZBEATQ2MTQU6VOERY3OIA2ZEG
WARC-IP-Address: 128.130.195.9
WARC-Record-ID: <urn:uuid:65c2b218-a409-48df-b214-949d9f525121>
Content-Type: application/http; msgtype=response
Content-Length: 16129

HTTP/1.1 200 OK
Date: Thu, 17 Nov 2011 11:34:24 GMT
Server: Apache
ETag: \"bfae4afe39db31761185887777f4aeca-d4f7a3e9f23081b65eaa66351a29b3c5\"
X-UA-Compatible: IE=Edge,chrome=1
Cache-Control: max-age=0, private, must-revalidate
Last-Modified: Thu, 17 Nov 2011 09:12:58 GMT
Status: 200
Content-Type: text/html; charset=utf-8
Set-Cookie: ...
Connection: close

<HTML data>...
```

Listing 3.3: A WARC Response Record

```
WARC/1.0
WARC-Type: metadata
WARC-Target-URI: http://www.informatik.tuwien.ac.at/
WARC-Date: 2011-11-17T11:34:29Z
WARC-Concurrent-To: <urn:uuid:65c2b218-a409-48df-b214-949d9f525121>
WARC-Record-ID: <urn:uuid:0133efbd-5d88-4941-a5ba-2d7a501f50be>
Content-Type: application/warc-fields
Content-Length: 3467

seed:
fetchTimeMs: 219
outlink: http://www.informatik.tuwien.ac.at/aktuelles L a/@href
outlink: http://www.informatik.tuwien.ac.at/kontakt/presse L a/@href
outlink: http://www.informatik.tuwien.ac.at/suche L form/@action
outlink: http://www.tuwien.ac.at/ L a/@href
outlink: http://www.informatik.tuwien.ac.at/www.informatik.tuwien.ac.at L input/@value
outlink: mailto:trouble@zkk.tuwien.ac.at L a/@href
outlink: http://www.informatik.tuwien.ac.at/fakultaet/zentrale-einrichtungen L a/@href
...
```

Listing 3.4: A WARC Metadata Record

Figure 3.3 shows this graphically: Only the HTTP response data block contains the valuable information that has to be restored.

The retrieved data represents an earlier archived object and is saved for further processing in some temporary directory on the hard disk. To identify the file, the retrieved name of this object is used; this name is available in the last part of the *WARC-Target-URI* in the WARC headers of the file. Because not all URIs contain the name (for instance, the WARC-Target-URI in Listing 3.3, `http://www.informatik.tuwien.ac.at` usually seeks for

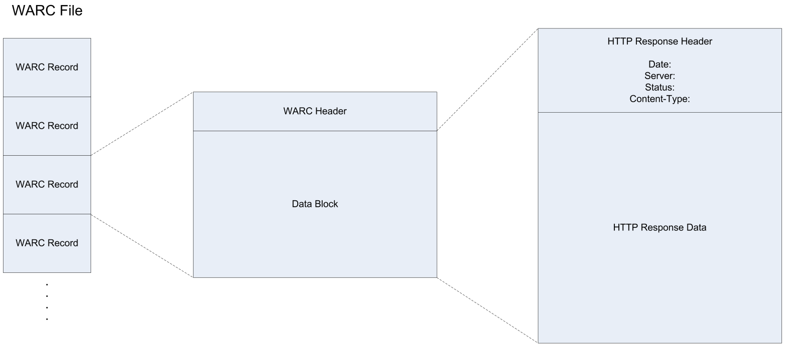


Figure 3.3: Format of a HTTP Response embedded in a WARC Record

a file located at `http://www.informatik.tuwien.ac.at/index.html` or `http://www.informatik.tuwien.ac.at/index.htm`, but does not contain the requested filename, because the identification is handled by the server automatically), the current time stamp gets prefixed to the filename, so a unique identification is possible.

3.4 The FITS Classifier

File Formats

To be able to perform the analysis of the file distributions, their correct file format has to be identified. This identification is a complex task.

All digital information consists of a sequence of bits („binary digits“), that can have the values *0* or *1* and can be stored on any physical medium (like CDs, DVDs or Harddisks). Such a sequence of bits is also called a „bit stream“. This bit stream can represent almost anything, from a single digit or a text document to an image or video. Figure 3.4 shows a random bit stream, that could represent a character, an audio file, a boolean truth table, a digit, a part of an image, or a part of a video.

Typically, digital objects are saved as *files*, that often contain logically related elements, that are linked to each other by cross-references.

After a bit stream is retrieved from a physical medium, it has to be interpreted. The specification of how to interpret it is called a *file format*.

Brown (2006) defines a file format as follows:

> „The internal structure and encoding of a digital object, which allows it to be processed, or to be rendered in human-accessible form. A digital object may be a file, or a bitstream embedded within a file.“ [10]

A file format defines whether a file is binary or ASCII and how the information is organized. For example, the Microsoft Word Format is a specification for the storage of textual data, along with special formatting information, like typography, layout and structure of the text. These

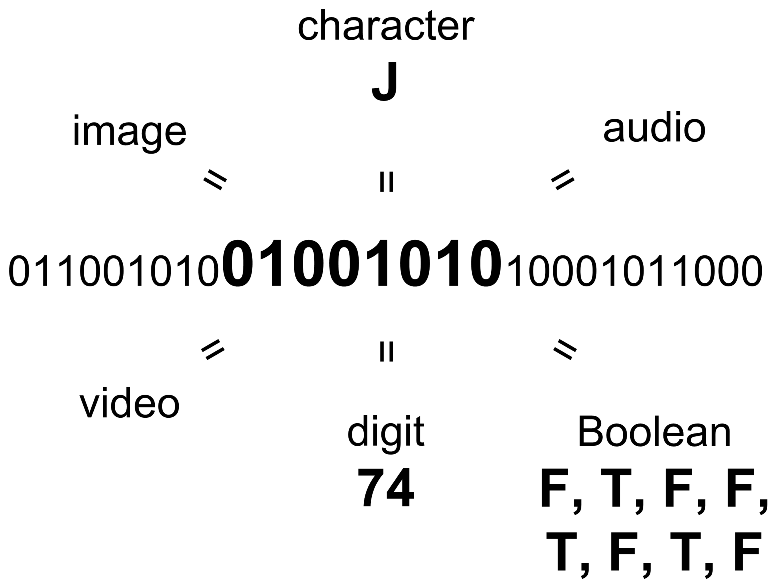


Figure 3.4: A bit stream can represent almost anything; Based on [54]

complex file formats make the binary code meaningless to a human, and they need appropriate software to be accessed. [54]

Identification of File Formats

Different approaches for identifying the format of a specific file do exist, that will be discussed here, and all of them have different accuracies.

File Extensions

The number of file formats defined by companies and organizations lies in the thousands. On March, 17, 2012, FILext[7], a popular *file extension* database, claim to have 26,024 different file extensions in their database. The file extension is a very common method to distinguish different file formats under DOS and Windows operating systems. For example, the file extension *DOC* is primarily associated with „Microsoft Word Document“, and *PNG* is a common image file extension associated with „Portable Network Graphic“. But these file extensions are neither standardized nor unique. The extension of a file can be changed very easily, but that does not change the internal meaning of the file. If a Microsoft Word Document is renamed from *file1.DOC* to *file1.PNG*, a user usually identifies this file as an image, but it is still a text document and can be opened with Microsoft Word as before.

So a big downside of file extensions is, that they do not provide sufficient granularity. As an example, the *.doc* extension does not distinguish between Word 8.0 and Word XP documents, although these are different formats. [10]

---

[7] `http://filext.com/`

Media Type (MIME Type)

The official categorization (as specified by RFC2046 in 1996 [21]) of file formats is the *IANA MIME Media type*[8], which currently has 1334 registered types. A media type consists of at least two parts: a *type*, a *subtype*, and optional parameters.

IANA defines the following 8 main types, the number in parentheses shows the number of registered subtypes in within the categories:

- application (977)
- audio (134)
- image (43)
- message (20)
- model (15)
- multipart (14)
- text (60)
- video (71)

The media types are also not unique, for example the subtype *rtf* exists in two categories, *application/rtf* and *text/rtf*. Additionally, media types do not provide sufficient granularity to tell different versions of one type apart. The PDF versions 1.0 (released in 1993) through 1.5 (released in 2001), PDF/X-1 and PDF/A all have the same media type, *application/pdf*.

File Format Registries

Some projects were developing systems which provide detailed information about internal specifications of file formats for use in digital preservation. One of these registries is PRONOM[9], that was developed by the Digital Preservation Department of the UK National Archives. This registry holds information about software products, and the file formats which each product can read and write. [45]

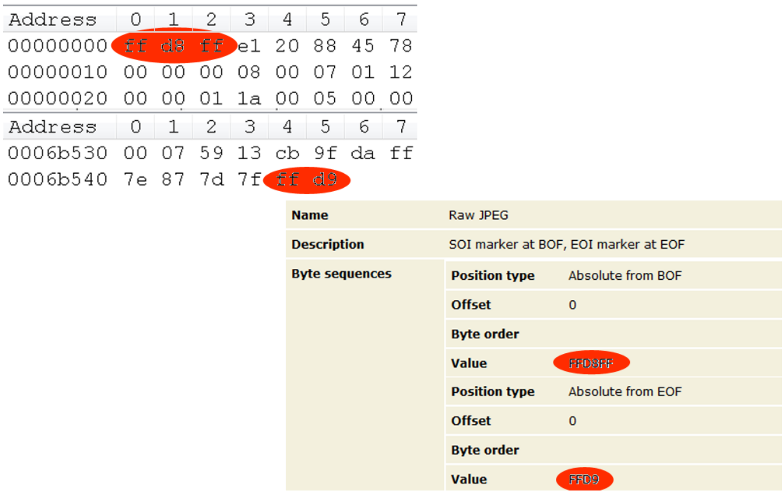
File formats in the PRONOM database are identified by a *format signature*, that is any collection of characteristics which may be used to indicate the format of a digital object. PRONOM uses two kinds of format signatures: internal and external signatures. External signatures encompass format indicators, that are external to the object bit stream, like Windows file extensions (see 3.4). As said before, these external signatures are not reliable and should only be used to provide a general indication of a file format.

Internal signatures, on the other hand, are contained within the object bit stream. File format specifications often define a specific structure of the bit stream, that is consistent between all

---

[8] `http://www.iana.org/assignments/media-types/index.html`

[9] `http://www.nationalarchives.gov.uk/PRONOM/`



```
Address   0  1  2  3  4  5  6  7
00000000 ff d8 ff e1 20 88 45 78
00000010 00 00 00 08 00 07 01 12
00000020 00 00 01 1a 00 05 00 00
Address   0  1  2  3  4  5  6  7
0006b530 00 07 59 13 cb 9f da ff
0006b540 7e 87 7d 7f ff d9
```

Name	Raw JPEG	
Description	SOI marker at BOF, EOI marker at EOF	
Byte sequences	**Position type**	Absolute from BOF
	Offset	0
	Byte order	
	Value	FFD8FF
	Position type	Absolute from EOF
	Offset	0
	Byte order	
	Value	FFD9

Figure 3.5: The hex view of a jpg file and its PRONOM definition

digital objects of this format. PRONOM tries to identify one or more byte sequences within a digital objects binary data and match these sequences with the signature file. [10]

Figure 3.5 shows parts of the hexadecimal view of the bit stream of an example JPEG file on the left and its matching PRONOM internal signature on the right[10]. The PRONOM definition for the *Raw JPEG Stream* looks for the hex values *FFD8FF* on offset 0 from BOF (beginning of file), and for the hex values *FFD9* on offset 0 from EOF (end of file). These values can indeed be found on the first and last bytes of this image file (the red circles in the figure), and therefore the file is classified as PUID fmt/41.

FITS

This thesis uses FITS (File Information Tool Set) [11] 0.6.0, an open source Java tool, that was created by the Harvard University Library Office for Information Systems [12] and identifies, validates and extracts technical metadata for various file formats.

FITS wraps different open-source tools, converts their output into a common format, and consolidates them into a single XML output file [20]. Hence, the identification of a file's format does not rely on only one tool, but is aggregated from different tools' outputs, that all have their specific strengths and weaknesses.

Currently, FITS uses the following external open source tools: Jhove [13], Exiftool [14], NLNZ

---

[10] `http://www.nationalarchives.gov.uk/PRONOM/Format/proFormatSearch.aspx?status=detailReport&id=670&strPageToDisplay=signatures`

[11] `http://code.google.com/p/fits/`

[12] `http://hul.harvard.edu/ois/`

[13] `http://hul.harvard.edu/jhove/`

[14] `http://www.sno.phy.queensu.ca/~phil/exiftool/`

Metdata Extractor [15], DROID [16] (a Java implementation, that uses the earlier discussed PRONOM registration), FFident [17] and the unix file utility command. Theses tools do not only try to determine the format and version of a file, but also other general information like file size or the last modified date, and file format specific metadata, like author and page count for PDF files.

If different tools report different results for one FITS property, a status attribute is added to the element with the value „CONFLICT“. These conflicting files are specially treated, as described in Section 3.5 on page 28.

Listing A.1 shows a sample XML-file that was created from FITS by analyzing a PDF document. This file contains a lot of information, that can be used to find, for instance, distributions of file formats and versions.

The FITS Classifier identifies the created files and saves the generated XML files to the harddisk. After the FITS process has completed (no more unidentified files), the XML files are parsed and the resulting data is saved to a database for easier analysis. This process is performed with c3po [18], a tool, that generates a profile for a given collection and produces output, that can be queried for the analysis part of this thesis [50].

HTML Parser Tool

The core FITS API does not provide sufficient information about HTML files, which are needed to fulfill one of the main tasks of this thesis. For this purpose, a new Java tool was created, that analyzes the structure, version and encoding of HTMLs. The tool is integrated as a new FITS tool and saves its findings to the consolidated FITS XML file. This tool is integrated into the analysis of Dataset DS2 (described in Section 4.1).

The tool fulfills the following tasks on the HTML files:

1. identification of the version
2. identification of the encoding
3. identification of the media type (*text/html* or *application/xhtml+xml*)
4. counting of the used tags
5. counting of embedded objects and analysis of their media types
6. counting of absolute and relative links

The HTML Parser Tool uses version 2.0 of the open source java library `org.htmlparser` [19] to iterate over the whole tags in the files which are to be analyzed.

To recognize HTML files, the parser tool follows the guidelines proposed in Section 5 of the RFC 2854 [12]. The identification of the HTML version is performed through the analysis of

---

[15] `http://meta-extractor.sourceforge.net/`

[16] `http://droid.sourceforge.net/`

[17] `http://schmidt.devlib.org/ffident/index.html`

[18] `http://github.com/peshkira/c3po`

[19] `http://htmlparser.sourceforge.net/`

the doctype (HTML document type) tag. The HTML Parser is able to identify, amongst others, the following valid HTML doctype definitions, on the basis of the W3C's list of recommended doctype definitions [13]: HTML 2.0, HTML 3.2, HTML 4.01, HTML 5, XHTML 1.0, XHTML 1.1. If the file contains an invalid definition, the parser is not able to identify the version and sets the corresponding field to *undefined*.

Listing 3.5 show 3 common doctype definitions, these are (from top to bottom) definitions for HTML 5, HTML 4.01 and XHTML 1.0.

```
<!-- HTML 5 -->
<!DOCTYPE HTML>

<!-- HTML 4.01 -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
        "http://www.w3.org/TR/html4/strict.dtd">

<!-- XHTML 1.0 -->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

Listing 3.5: Common doctype definitions

The media type of the document depends on the identified doctype definition. Based on the W3C recommendation for media type usage [20], the HTML parser tool tries to perform a strict separation between HTML and XHTML. Documents, that contain a HTML, and not a XHTML definition, are classified as *text/html*, while documents, that contain an XHTML doctype definition, are classified as *application/xhtml+xml* [4]. Documents, that don't contain any doctype definition, but valid HTML start- and end-tags (`<html>` and `</html>`, are supposed to be standard HTML files and therefore classified as *text/html*.

Next, the character encoding for the HTML file is set. The character encoding should be specified for all (X)HTML pages, to avoid the risk, that characters are incorrectly interpreted. The encoding is essential for browsers and other tools, so they can display the web sites correctly. Common encodings are, for instance, *UTF-8* or *ISO-8859-1*. For a detailed analysis of possible character sets, please see the list at IANA [21].

Another big task is the counting of used tags. The HTML parser tool is not only able to identify HTML-compliant tags, but also user defined and misspelled tags. This counting gives insights in the structure of HTML documents, as well as the usage of tags, that were invented by specific HTML versions.

The HTML parser tool is also able to identify absolute and relative hyperlinks to other websites and files. It is also possible for the HTML parser tool to analyze embedded objects, like Java Applets or Active-X-Controls. The parser tool relies on the *type* attribute of the *object* tag to identify the media type of the data used in the object.

---

[20] `http://www.w3.org/TR/xhtml-media-types/`

[21] `http://www.iana.org/assignments/character-sets`

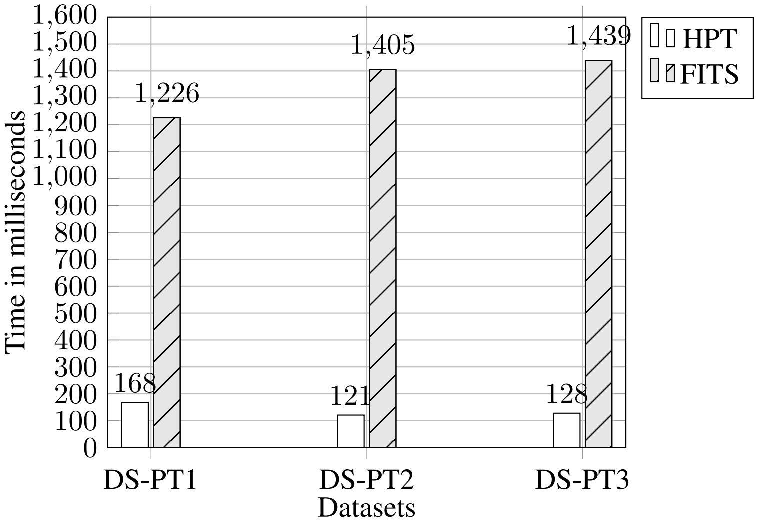


Figure 3.6: Time (in ms) to analyze one file with FITS and the HTML Parser Tool (HPT)

Performance

To avoid bottlenecks and to test the stability of the HTML Parser Tool, before analyzing the web archive files, that were used as a basis for the results of this thesis (discussed in chapter 4), some performance tests were run on smaller datasets. These tests were performed on a computer running Windows 7 with 4GB RAM and a dual core processor with 2 x 2.13 GHz.

The performance tests were run with 3 different datasets, that were collected by crawling random (sub-)domains of the web:

1. *Dataset Performance Test 1*, or *DS-PT1*, containing 2108 files

2. *Dataset Performance Test 2*, or *DS-PT2*, containing 4212 files

3. *Dataset Performance Test 3*, or *DS-PT3*, containing 10.726 files

These datasets were classified with the standard FITS tools as described in section 3.4, including the HTML Parser tool. For each dataset, the average time to classify one file with the whole FITS, as well as the classification time for the HTML Parser tool (inside FITS) was measured. Figure 3.6 shows the results of these measurings.

The performance tests have shown, that the HTML Parser Tool takes about 10 to 15 % of the whole FITS run time for its analysis process, depending on the size and type of the file. This shows, that the tool works well, without any performance errors or bottlenecks, that would lead to a significantly longer runtime for FITS.

3.5 Analysis

For easier analysis and retrieval, the generated FITS files are parsed and their contained properties and metadata are saved to a database. This process is done by c3po [22], a tool that uses FITS generated data and generates a „digital profile“ for this collection. C3po parses and saves the FITS XML data to a JSON [23]-like format, so it can easily be accessed and queried. [49].

This profile isn't stored in a classical relational database, but it uses MongoDB [24], a scalable, high-performance, open source NoSQL database. To query this database and aggregate the results, MongoDB uses a concept called *map/reduce* [16]. This concept allows parallelism and thus fast processing of very big data sets. Two user-defined functions, *map* and *reduce*, that are written in JavaScript, are executed on the server. The *map* function processes a key-value-pair and generates a set of intermediate key-value-pairs, while the *reduce* function is in charge for the aggregation of pairs with the same intermediate key. The whole in-depth handling, like scheduling the execution across multiple processors and machines and managing the occurring inter- and intra-machine communications, as well as dealing with errors, is performed by MongoDB [16]. For a detailed description of MongoDB and its concepts, please refer to [11].

An example c3po output for a single file, in this case, a HTML file, is displayed in Listing A.2. All information obtained from the FITS file are parsed, aggregated and can be accessed.

To filter the dataset for specific file formats and group the specific versions, the IANA media type, that is defined inside the FITS files' metadata block is used as the format indicator. As FITS aggregates the output of different tools into one output file, it is possible, that different tools report different versions or formats of a single file. In this case, FITS reports a *CONFLICT* status in the output file, indicating the property, that led to this conflict. An example for such a conflict is displayed in Listing 3.6, which shows conflicts in format name, as well as the identified media type (referred to as *mimetype* in the listing). The example listing reads, that Jhove identified the file as mediatype *text/xml, version 1.0*, while Exiftool and the NLNZ Metadata Extractor identified it as *text/html*, but only the NLNZ Metadata Extractor identified version 1.0, and Droid as *application/xhtml+xml, version 1.0*.

```
<identification status="CONFLICT">
   <identity format="XHTML" mimetype="text/xml" toolname="FITS" toolversion="
       0.6.0">
      <tool toolname="Jhove" toolversion="1.5" />
      <version toolname="Jhove" toolversion="1.5">1.0</version>
   </identity>
   <identity format="Hypertext Markup Language" mimetype="text/html" toolname
       ="FITS" toolversion="0.6.0">
      <tool toolname="Exiftool" toolversion="7.74" />
      <tool toolname="NLNZ Metadata Extractor" toolversion="3.4GA" />
      <version toolname="NLNZ Metadata Extractor" toolversion="3.4GA">1.0</
          version>
   </identity>
   <identity format="Extensible Hypertext Markup Language" mimetype="
       application/xhtml+xml" toolname="FITS" toolversion="0.6.0">
```

---

[22] `http://ifs.tuwien.ac.at/imp/c3po`

[23] `http://json.org/`

[24] `http://www.mongodb.org/`

```
      <tool toolname="Droid" toolversion="3.0" />
      <version toolname="Droid" toolversion="3.0">1.0</version>
      <externalIdentifier toolname="Droid" toolversion="3.0" type="puid">fmt
          /102</externalIdentifier>
   </identity>
</identification>
```

Listing 3.6: A conflict in a FITS file

These conflicts are also copied to MongoDB and need to be handled; when they occur, it is not possible to detect the „real“ or „correct“ value without manually reading the file, which would not pay off in terms of tens of thousands of files.

To deal with that problem, a method for *fuzzy classification* of the files was created. All different identified formats with their corresponding versions (identified by the same tool) are used to classify the file, this method splits the identity of a file into the different reported formats and versions and creates a probability for each *(format,version)*-pair.

Formally, the probability for a (format,version)-pair is computed conforming to Equation 3.1.

$$fv_{f,v} = \frac{t_{f,v}}{t} \tag{3.1}$$

So, the fuzzy value $fv_{f,v}$ for a (format,version)-pair of a file is computed by dividing the number of tools, that have reported this (format,version)-pair $t_{f,v}$, by the total number of tools t, that have identified this file.

The FITS file, that is shown in Listing 3.6 would lead to the following (format,version)-probabilities;

- $(text/xml, 1.0) : 0.25$
- $(text/html, 1.0) : 0.25$
- $(text/html, undefined) : 0.25$
- $(application/xhtml + xml, 1.0) : 0.25$

This kind of fuzzy classification brings a kind of certainty into the classification process: the more tools are reporting the same format and version for a file, the more definitive this is the „correct“ (format,version)-pair. A (format,version)-pair that has a probability near 1 is much more reliable, than a file with probabilities of 0.25 or 0.33.

The JavaScript-Method that was written to return triples of version, format and probability is displayed in Listing A.3.

Map/Reduce

```
function map() {
   var jsonObj = getMediaTypeProbabilityIncludingVersion(this);

   for(elem in jsonObj) // emit each object
```

```
    {
       emit(
       {
          "mediatype" : jsonObj[elem].mediatype,     // the mediatype
          "version" : jsonObj[elem].version,      // version of the corresponding media type
          "coll" : this.collection             // collection (= year)
       },
       {
          "probability" : jsonObj[elem].probability // computed probability
       });
    }
}

function reduce(key, values) {
    var res = { probability : 0.0 };
    values.forEach(function (v) {
      res.probability += v.probability;
   });
    return res;
}
```

Listing 3.7: JavaScript map and reduce functions

The *map* and *reduce* functions, that are used for counting the probabilities for all versions of all media types are displayed in Listing 3.7. These functions, that are written in JavaScript, are called automatically by MongoDB during the map-reduce process. *map* gets called on each existing FITS XML file in the used collection, it calls the *getMediaTypeProbabilityIncludingVersion*-method, that is defined beforehand and handles the return, which is a JSON [25] structure consisting of all (format, version, probability)-triples, that have been found for the file (similar to Listing 3.6). Each triple is then separately combined with the collection and emitted. In case of counting the versions of all different file formats, the emitted structures' format is displayed in Listing 3.8.

```
{
   "mediatype" : "text/html",
   "version" : "4.0",
   "coll" : "dk2010"
},
{
   "probability" : 0.66666
}

{
   "mediatype" : "text/html",
   "version" : "4.01",
   "coll" : "dk2010"
},
{
   "probability" : 0.33333
```

---

[25] JavaScript Object Notation, see `http://www.json.org/` for further information

```
}
```

Listing 3.8: Format of the emitted structures

After the mapping process has finished, the reduce phase starts, whose goal is to combine all intermediate (previously emitted) results to form the output. In case of the above example, this phase adds up all probabilities, that have the same (mediatype, version, coll)-triple, which leads to the total number of files for each specific version of each file format in one specific year. The results of the *reduce* phase look similar to the results of the mapping phase, except the probabilities are usually much higher, because many file probabilities have been summed up (Listing 3.9).

```
{
   "_id" : {
      "mediatype" : "text/html",
      "version" : "4.01",
      "coll" : "dk2005"
   },
   "value" : {
      "probability" : 2497
   }
}{
   "_id" : {
      "mediatype" : "text/html",
      "version" : "4.01",
      "coll" : "dk2006"
   },
   "value" : {
      "probability" : 7485.5
   }
}
```

Listing 3.9: Exemplary results of the *reduce* phase

HTML Tags

To retrieve and count the tags of all found (X)HTML files, new methods, including new *map/reduce*-methods were created. The process of counting the tags in a dataset is graphically illustrated in Figure 3.7. Although the actual process has concurrent elements, it is best visualized in a sequential diagram.

Until no more untreated FITS files are present in the collection, the next file is processed via the *map*-function. At first, a check for a single identification result is performed, so, it is checked, if different tools reported different formats or versions (which means, that more than one (format,version)-pair is present). If a single result has been found, all valid tags are filtered and saved as intermediate results for further processing through *reduce*. In the other case, if multiple formats or versions have been reported, all of these pairs are checked. If at least one result matches a (X)HTML file, the tags are again filtered and saved. Finally, a *finalize* function is called, that computes average tag occurrences per file. The *map*, *reduce* and *finalize* functions are displayed in Listing 3.10.

```
function map() {
```

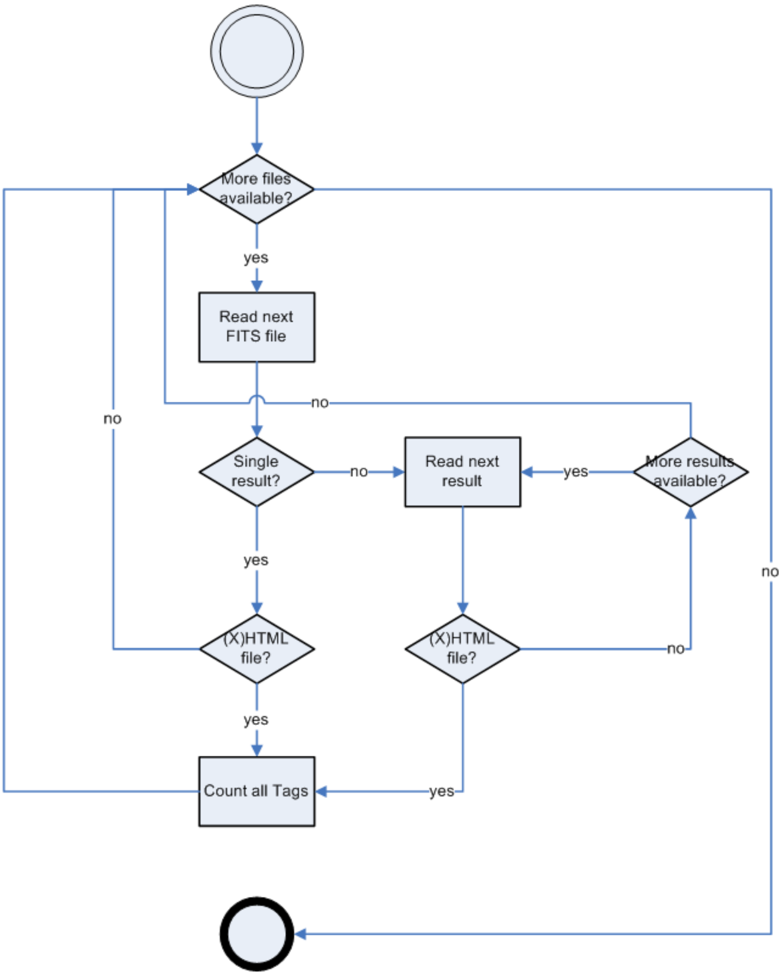


Figure 3.7: The tag counting activity

```
var relevant = 0;
// check for HTML or XHTML file
if(this.metadata.mimetype.value === "text/html" || this.metadata.mimetype.
    value === "application/xhtml+xml")
   relevant = 1;
else if(this.metadata.mimetype.values)
{
   var arrLen = this.metadata.mimetype.values.length;

   // loop over media types
   for (var i = 0; i < arrLen; i++)
   {
      if(this.metadata.mimetype.values[i] === "text/html" || this.metadata
          .mimetype.values[i] === "application/xhtml+xml")
      {
         relevant = 1;
```

```
            break;
         }
      }
   }
   if(relevant == 1)
   {
      var tags = getTagOccurrences(this);
      if (tags.length > 0) {
         for (elem in tags) {
            emit({
               "tag" : tags[elem].tag,
               "coll" : this.collection
            }, {
               "count" : tags[elem].count,
               "numFiles" : 1
            });
         }
      }
   }
}

function reduce(key, values) {
   var n = {
      count : 0,
      numFiles : 0
   };
   values.forEach(function (v) {
      n.count += v.count;
      n.numFiles += v.numFiles;
   });
   return n;
};

function finalize(who, res) {
   res.avg = res.count / res.numFiles;
   res.totalAvg = res.count / totalInputFiles;
   return res;
};
```

Listing 3.10: JavaScript *map*, *reduce* and *finalize* functions for the extraction of HTML tag occurrences

The method to filter valid HTML tags (Listing A.4) checks the FITS file for 121 different tags, that have been defined in different HTML and XHTML specifications ([51], [53], [52], [29], [48], [39]). Only these tags are filtered and counted, custom defined or misspelled tags are not taken into account.

Further processing

The results of the c3po analysis process are files containing many entries, like shown in Listing 3.9. These files can be queried and aggregated using the free statistical computing program

R [26]. With R it is easily possible to filter the data, create subsets or statistics. The R queries that were created to comput the results of this thesis are hosted on GitHub [27], where they can be downloaded and reused.

---

[26] `http://www.r-project.org/`

[27] `https://github.com/stefanschindler/ffa`

CHAPTER 4

Results

This chapter presents the results of the performed crawls and calculations, and outlines the findings, that have been made through this work. Furthermore, the dataset and the statistical background are explained.

4.1 The Datasets

3 different datasets were used for the experiments reported in the previous chapter. This section gives an overview of the different datasets, as well as their primary usage.

The Dataset for Format-Version distribution analysis (DS1)

The main dataset (referenced as DS1) used for this thesis was provided by the StatsBiblioteket (The State and University Library) in Aarhus, Denmark [1]. This institution crawled the Danish web each year, dating back to 2005. The single files were extracted from the crawled archive files and analyzed with FITS. As the resulting data volume is too big for analysis, a sample of each year has been taken, that is used for hypotheses for the whole danish web.

To prove the validity of different hypotheses, two independent subsets of the resulting *.fits.xml* files were parsed and saved to a database with c3po. The first subset, the *training set*, sizes about 1 GB of fits files per year and is used for the hypotheses. Afterwards, a second subset, the *test set*, that contains between 40% and 55% of the training set's size, tests, if the hypotheses are valid or have to be rejected.

Table 4.1 gives an overview about the sizes and count of the used files for training set and test set. The numbers in parentheses represent the test set's rounded percentages of the training set's data volume. The given file sizes and numbers of files represent the obtained FITS files.

As Jhove also reports the sizes of the characterized files, and these sizes are also aggregated to the resulting FITS files, so an overview of their sizes can also be given. Jhove could not char-

---

[1] `http://www.statsbiblioteket.dk/`

	Training Set		Test Set	
Year	FITS Size	Number of files	FITS Size	Number of files
2005	994 MB	173 001	492 MB (49.5%)	88 367 (51.1%)
2006	998 MB	165 948	447 MB (44.8%)	75 980 (45.8%)
2007	1.3 GB	205 110	522 MB (40.2%)	98 887 (48.2%)
2008	1.2 GB	248 670	661 MB (55.1%)	118 841 (47.8%)
2009	889 MB	165 935	475 MB (53.4%)	88 104 (53.1%)
2010	967 MB	153 733	532 MB (55.0%)	76 711 (49.9%)
2011	1.3 GB	208 991	703 MB (54.1%)	99 934 (47.8%)

Table 4.1: Overview of the subsets (FITS files) - DS1

acterize each file correctly, so not all file sizes are available, but it was possible to characterize at least 99.7% of each years files, which leads to sizes as shown in Table 4.2.

Taking into account only files that were characterized by Jhove, the total file size per year of Jhove's characterized files ranges from 3 GB to nearly 7.9 GB in the training set, and from 1.7 GB to 4.1 GB in the test set, and the average file sizes of the files characterized by Jhove range from 19.3 KB to 36.7 KB (training set), respectively from 20.9 KB to 40 KB (test set).

	Training Set			Test Set		
Year	Total size (GB)	% ch.	Avg. size (KB)	Total size (GB)	% ch.	Avg. size (KB)
2005	4.475	99.994%	27.12	3.097	99.96%	36.77
2006	3.053	99.999%	19.29	1.765	99.7%	24.44
2007	7.181	99.998%	36.72	3.763	99.993%	39.9
2008	7.859	99.769%	33.22	4.132	99.903%	36.5
2009	3.764	99.889%	23.81	2.664	99.918%	31.74
2010	3.916	99.826%	26.76	1.729	99.911%	23.66
2011	4.696	99.888%	23.59	1.987	99.994%	20.85

Table 4.2: Overview of the total and average file sizes per set and year (characterized files) - DS1

The Dataset for HTML Analysis (DS2)

As HTML tags, that are a part of this thesis, and are computed by the HTML Parser Tool, were not available in the obtained dataset, that was used for the other computations (DS1), a new dataset including HTML tags was obtained (DS2). This set also contains data, that were archived by the Danish StatsBiblioteket, but with HTML informations included. For the analysis of this Dataset, the proposed HTML Parser Tool (as described in Section 3.4) was included into the FITS process. To be consistent over the whole HTML analysis, DS2 was not only used for the HTML tags, but also for analyzing the distribution of the different HTML versions over time and for the used HTML encodings (for all results of this section). Unfortunatley, DS2 does not contain data from the year 2006, so for the HTML analysis only data from 2005 and 2007 to 2011 are available.

The parsing and analysis procedure was exactly as described above, with the addition, that the *.fits.xml* files also contained output of the HTML Parser Tool. This second obtained dataset (DS2) is smaller than the first one (DS1), with having 40 000 to 70 000 files per year (for the training set, as well as for the test set), only for the first analyzed years (2005) just around 8 000 files are available, as can be seen in Table 4.3.

	Training Set	**Test Set**
Year	**Number of files**	**Number of files**
2005	8 316	8 319
2007	43 456	43 454
2008	46 126	46 123
2009	72 199	72 197
2010	53 154	53 153
2011	37 724	37 723

Table 4.3: Number of files in the Dataset for HTML Analysis - DS2

The Large Scale Dataset (DS3)

Additionally, a third, much bigger dataset than the ones above, was retrieved from the Danish StatsBiblioteket, with about 1 million FITS files per year, now containing crawled years from 2005 to 2012 (compared to 2005 to 2011 for the previous collections). The main purpose of this dataset is, to compare results generated for DS1 to another, bigger, set. The number of files per year is displayed in the column *Number of files* of Table 4.4. As Jhove was also a part in the characterization of DS3, and Jhove reports the sizes of the characterized files, an overview about these can also be given. The other columns of the Table show these data; the column *% ch.* shows, how many of the crawled files could be characterized by Jhove (at leas 99.939%). The total file sizes of the Jhove metadata files range from 21 to 32 GB, while the average file sizes differs between 22 and 34 KB (which is nearly identical to the average file sizes of DS1).

Summary of the Datasets

Table 4.5 shows a summary of the total files sizes (referred to as *Size (GB)* in the table header) and number of crawled files (referred to as *# files* in the table header) for each available dataset, as well as the years where no crawl data is available for the specific sets (*Not covered*).

		Jhove metadata		
Year	**Number of files**	**Total size (GB)**	**% ch.**	**Avg. size (KB)**
2005	999 873	32.45	100%	34.02
2006	999 911	24.91	100%	26.13
2007	999 040	25.49	99.999%	26.92
2008	999 872	28.46	99.964%	29.85
2009	999 930	22.43	99.944%	23.54
2010	999 783	21.33	99.939%	22.38
2011	999 946	22.33	99.996%	23.42
2012	999 712	22.98	100%	24.1

Table 4.4: Number of files and Jhove metadata - DS3

	Dataset 1				Dataset 2				Dataset 3	
	Training Set		Test Set		Training Set		Test Set			
Year	# files	Size (GB)	# files	Size (GB)	# files	Size (GB)	# files	Size (GB)	# files	Size (GB)
2005	173 001	4.475	88 367	3.097	8 316	0.041	8 319	0.046	999 873	32.45
2006	165 948	3.053	75 980	1.765	*Not covered*		*Not covered*		999911	24.91
2007	205 110	7.181	98 887	3.763	43 456	0.906	43 454	0.967	999 040	25.49
2008	248 670	7.859	118 841	4.132	46 126	1.194	46 123	1.187	999 872	28.46
2009	165 935	3.764	88 104	2.664	72 199	1.154	72 197	1.179	999 930	22.43
2010	153 733	3.916	76 711	1.729	53 154	1.078	53 153	1.098	999 783	21.33
2011	208 991	4.696	99 934	1.987	37 724	0.867	37 723	0.837	999 646	22.33
2012	*Not covered*		*Not covered*		*Not covered*		*Not covered*		999 712	22.98

Table 4.5: Summary of the available datasets

4.2 Statistical Background

Two independent random samples (training set and test set) of the population are taken. To test files for a specific version of a specific format, they can be seen as binomial proportions, such as if a file has a specific version of a specific format, or if it doesn't have a specific version of a specific format. To be able to identify, if a proportion of a version/format combination can be seen as significant for the whole collection, the proportions of the two random samples are compared for each crawled year. This thesis uses Pearson's χ^2 test to test for equality of proportions, as described by Karl Pearson in 1900 [47].

The χ^2 test statistic is a measurement of how close the observed frequencies for a specific version of a specific file format are to the expected frequencies, and is for the used case with two samples, overall computed according to equation 4.1, with $O_{i,j}$ being the observed frequency for a specific property, and $E_{i,j}$ being the expected frequency for the same specific property.

$$\chi^2 = \sum_{j=1}^{2} \sum_{i=1}^{2} \frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}} \tag{4.1}$$

The index i in this equation is used for the 2 rows, and j for the 2 columns in a 2x2 matrix containing the values „sample 1, file is of the specific format version“, „sample 1, file is not of the specific format version“, „sample 2, file is of the specific format version“, „sample 2, file is not of the specific format version“. An example for such a matrix (containing some observed frequencies) is given in Table 4.6. These are fictitious values for the distribution of PDF version 1.6. In the first sample, 51 of 100 observed PDF files were found to be version 1.6, while 49 of the files weren't 1.6. In the second sample, 68 of 120 investigated PDF files were version 1.6, and 52 weren't.

	Sample 1	Sample 2	$\sum$
is of Version	51	68	119
is not of Version	49	52	101
$\sum$	100	120	220

Table 4.6: Exemplary observed values for 2 samples

The expected frequency for a specific cell $c_{k,l}$ is computed through equation 4.2.

$$E_{k,l} = \frac{\sum_{i=1}^{2} c_{i,l}}{\sum_{j=1}^{2} \sum_{i=1}^{2} c_{i,j}} \cdot \sum_{j=1}^{2} c_{k,j} \tag{4.2}$$

The expected values (including the computations) for the exemplary observed values (table 4.6) are displayed in table 4.7.

	Sample 1	Sample 2	$\sum$
is of Version	$\frac{51+49}{51+49+68+52} \cdot (51 + 68) \approx 54,09$	$\frac{120}{220} \cdot 119 \approx 64,91$	119
is not of Version	$\frac{100}{220} \cdot 101 \approx 45,91$	$\frac{120}{220} \cdot 101 \approx 55,09$	101
$\sum$	100	120	220

Table 4.7: Expected values (including computations) for the values observed in table 4.6

With the observed and expected values for each cell being known, equation 4.1 can be used to compute the test statistics (equation 4.3).

$$
\begin{aligned}
\chi^2 &= \sum_{j=1}^{2}\sum_{i=1}^{2} \frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}} \\
&= \frac{(51 - 54,09)^2}{54,09} + \frac{(68 - 64,91)^2}{64,91} + \frac{(49 - 45,91)^2}{45,91} + \frac{(52 - 55,09)^2}{55,09} \\
&\approx 0,177 + 0,147 + 0,208 + 0,173 \\
&= 0,705
\end{aligned} \tag{4.3}
$$

Formulas 4.1 and 4.2 can be combined and formulated. Let A be a specific version of a file format in the training set and B be the corresponding version in the test set. Then the test statistics can be written according to equation 4.4 with c_{A_x} being the found occurrences of a specific version of a file format in a specific year x and n_{A_x} the total amount of files of this file format in the year x in the training set (the same goes for the test set with c_{B_x} being the occurrences of the same file version for year x and n_{B_x} the total file count of this file format in year x).

$$
\begin{aligned}
\chi^2 = &\frac{\left(c_{A_x} - \frac{n_{A_x}}{n_{A_x}+n_{B_x}} \cdot (c_{A_x} + c_{B_x})\right)^2}{\frac{n_{A_x}}{n_{A_x}+n_{B_x}} \cdot (c_{A_x} + c_{B_x})} + \frac{\left((n_{A_x} - c_{A_x}) - \frac{n_{A_x}}{n_{A_x}+n_{B_x}} \cdot (n_{A_X} - c_{A_x} + n_{B_x} - c_{B_x})\right)^2}{\frac{n_{A_x}}{n_{A_x}+n_{B_x}} \cdot (n_{A_X} - c_{A_x} + n_{B_x} - c_{B_x})} + \\
&\frac{\left(c_{B_x} - \frac{n_{B_x}}{n_{A_x}+n_{B_x}} \cdot (c_{A_x} + c_{B_x})\right)^2}{\frac{n_{B_x}}{n_{A_x}+n_{B_x}} \cdot (c_{A_x} + c_{B_x})} + \frac{\left((n_{B_x} - c_{B_x}) - \frac{n_{B_x}}{n_{A_x}+n_{B_x}} \cdot (n_{A_X} - c_{A_x} + n_{B_x} - c_{B_x})\right)^2}{\frac{n_{B_x}}{n_{A_x}+n_{B_x}} \cdot (n_{A_X} - c_{A_x} + n_{B_x} - c_{B_x})}
\end{aligned} \tag{4.4}
$$

Let $H_0 : p_{A_x} = p_{B_x}$ be the null hypothesis, that the proportions p_{A_x} and p_{B_x} are equal under a defined significance niveau of $\alpha = 0.01$ and $H_1 : p_{A_x} \neq p_{B_x}$ be the alternative hypothesis, that the proportions p_{A_x} and p_{B_x} are *not* equal (again, considering α). As $c = p \cdot n$ and thus $p = \frac{c}{n}$, equation 4.4 can be reformed to contain p values. The test statistics X^2 is computed according to equation 4.5. This test will determine, if the proportions p_{A_x} and p_{B_x}, describing the probabilities of a specific version of a file format in the sample for a specific year x over the whole files of this file format, are significantly different.

$$\begin{aligned}\chi^2 = {} & \frac{\left((p_{A_x} \cdot n_{A_x}) - \frac{n_{A_x}}{n_{A_x}+n_{B_x}} \cdot ((p_{A_x} \cdot n_{A_x}) + (p_{B_x} \cdot n_{B_x}))\right)^2}{\frac{n_{A_x}}{n_{A_x}+n_{B_x}} \cdot ((p_{A_x} \cdot n_{A_x}) + (p_{B_x} \cdot n_{B_x}))} + \\ & \frac{\left((n_{A_x} - (p_{A_x} \cdot n_{A_x})) - \frac{n_{A_x}}{n_{A_x}+n_{B_x}} \cdot (n_{A_X} - (p_{A_x} \cdot n_{A_x}) + n_{B_x} - (p_{B_x} \cdot n_{B_x}))\right)^2}{\frac{n_{A_x}}{n_{A_x}+n_{B_x}} \cdot (n_{A_X} - (p_{A_x} \cdot n_{A_x}) + n_{B_x} - (p_{B_x} \cdot n_{B_x}))} + \\ & \frac{\left((p_{B_x} \cdot n_{B_x}) - \frac{n_{B_x}}{n_{A_x}+n_{B_x}} \cdot ((p_{A_x} \cdot n_{A_x}) + (p_{B_x} \cdot n_{B_x}))\right)^2}{\frac{n_{B_x}}{n_{A_x}+n_{B_x}} \cdot ((p_{A_x} \cdot n_{A_x}) + (p_{B_x} \cdot n_{B_x}))} + \\ & \frac{\left((n_{B_x} - (p_{B_x} \cdot n_{B_x})) - \frac{n_{B_x}}{n_{A_x}+n_{B_x}} \cdot (n_{A_X} - (p_{A_x} \cdot n_{A_x}) + n_{B_x} - (p_{B_x} \cdot n_{B_x}))\right)^2}{\frac{n_{B_x}}{n_{A_x}+n_{B_x}} \cdot (n_{A_X} - (p_{A_x} \cdot n_{A_x}) + n_{B_x} - (p_{B_x} \cdot n_{B_x}))}\end{aligned} \tag{4.5}$$

If this computed value X^2 is greater than or equal to the critical value $\chi^2_{1;\alpha}$, which is 6.64 for $\alpha = 0.01$, the null hypothesis is rejected as there is a statistically significant difference, otherwise the null hypothesis is not rejected.

Considering the previous example, whose X^2 value has been computed in equation 4.3, the computed value 0.705 is smaller than the critical value $\chi^2_{1;0.01}$, which is 6.64, so the null hypothesis, that the found values resp. the probabilites for PDF 1.6 is equal in both samples, can not be rejected.

For this thesis, these computations, as well as most figures and diagrams presented in this chapter, are performed automatically with the free statistical computing program R [2]. The χ^2 tests, for instance, are performed with the R function *prop.test*.

4.3 Conversion rules

All identification tools, that are integrated into the FITS process, try to identify the files depending on specific properties and metadata. As not all files are well-defined, some tools may possibly report wrong results. Additionally, not all tools are able to identify all types and kinds of files, so not all tools are taken into account when creating the corresponding FITS files. Through the used fuzzy classification (see 3.5), the wrong results are usually rated with a low probability, but in some cases it is possible, that (nearly) all classifying tools report this wrong result.

As not all FITS files can be checked manually, some common mistakes were identified, that would lead to highly alternating results, so they were manually (in terms of changing the R filters) corrected before computing the statistics:

1. FITS produced a lot of conflicts between media type *application/xhtml+xml 1.0* (XHTML 1.0) and HTML 1.0 (*text/html 1.0*), resp. *application/xhtml+xml 1.1* (XHTML 1.1) and HTML 1.1 (*text/html 1.1*) in its output, (HTML reported by Exiftool and the NLNZ Metadata Extractor). As the first HTML version that was officially standardized was HTML

---

[2] `http://www.r-project.org/`

2.0 in 1995 (RFC 1866[3]), and there does not exist an HTML 1.0 standard, this led to the assumption, that these files were identified wrong by some tools.

Through manually checking 50 valid[4] XHTML files, it turned out, that all of these files were identified as HTML by Exiftool and the NLNZ Metadata Extractor. Additionally, some of these files were identified as Extensible Markup Language (*text/xml*) by Droid or ffident. This potentially occurred, because an XML declaration in the first line of each XHTML document is recommended by the XHTML 1.0 and 1.1 standards ([48], [39]), and just this declaration was used to identify the file.

A manually created valid XHTML file, like displayed in Listing 4.1 produced the FITS output, that is displayed in Listing 4.2 (just the necessary part of the FITS file is shown).

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/
    TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de">
 <head>
   <title>Beispiel</title>
 </head>
 <body>
    <h1>Beispielseite</h1>
    <p>Ein Absatz</p>
    <p>Noch ein<br />
    Absatz</p>
    <ol>
      <li>Listelement</li>
      <li>Listelement</li>
    </ol>
    <p>
      <img src="bild.gif" alt="Bildmotiv" />
    </p>
 </body>
</html>
```

Listing 4.1: An exemplary valid XHTML file

Because all of the manually checked files reported the same conflicts, the decision was made for the (X)HTML results of this thesis, that all files identified as *text/html 1.0* or *text/xml 1.0* are treated as *application/xhtml+xml 1.0*. Additionally, files that were identified as *text/html 1.1* or *text/xml 1.1* are treated as *application/xhtml+xml 1.0*.

2. The NLNZ Metadata Extractor identifies many JFIF files (media type *image/jpeg*) as version 1.1 and 1.2, which are not standardized. So these identified files are manually changed to the standardized JFIF 1.01 and 1.02 definitions [27].

---

[3] `http://tools.ietf.org/html/rfc1866`

[4] as reported by the W3C Markup Validation Service (`http://validator.w3.org/`

```
<identification status="CONFLICT">
   <identity format="Extensible Hypertext Markup Language" mimetype="
       application/xhtml+xml" toolname="FITS" toolversion="0.6.0">
      <tool toolname="HtmlInfo" toolversion="0.7" />
      <version toolname="HtmlInfo" toolversion="0.7">1.0</version>
   </identity>
   <identity format="Extensible Markup Language" mimetype="text/xml" toolname
       ="FITS" toolversion="0.6.0">
      <tool toolname="file utility" toolversion="5.03" />
      <tool toolname="Droid" toolversion="3.0" />
      <tool toolname="ffident" toolversion="0.2" />
      <version toolname="Droid" toolversion="3.0">1.0</version>
      <externalIdentifier toolname="Droid" toolversion="3.0" type="puid">fmt
          /101</externalIdentifier>
   </identity>
   <identity format="Hypertext Markup Language" mimetype="text/html" toolname
       ="FITS" toolversion="0.6.0">
      <tool toolname="Exiftool" toolversion="7.74" />
      <version toolname="Exiftool" toolversion="7.74">1.0</version>
   </identity>
</identification>
```

Listing 4.2: The FITS identification part of the XHTML file shown in Listing 4.1

3. The file utility tool identifies a lot of JPEG image files as version *4360*. As no related specifications could be found through an exhaustive search, these results aren't taken into account when computing the results of the undergoing analysis.

4. The file utility tool identifies all icon files as media type *image/x-ico*. The correct and registered media type would be *image/vnd.microsoft.icon* [5], so all these files are manually changed to this media type.

4.4 Version usage over time

The analysis of version usage over time explains, how intensively a specific version of a format has been used compared to the whole usage of this file format. If multiple versions of multiple different file formats have similar trends, it is possible to predict the development of current and future released versions and the need of supporting tools and preserving these tools and versions. The dataset used for the computations in this section is DS1.

Only versions, where the probability difference between training set and test set are not statistically significant according to the described background on page 40 are taken into account, as

---

[5] This is defined on the IANA website:

> „This media type is currently being labeled with a potpourri of names, including “text/ico”, “image/ico”, “image/icon”, “application/ico”, and so on. While “image/x-icon” is also used, this registration intends to finally clarify the media type for this file format.“ [55]

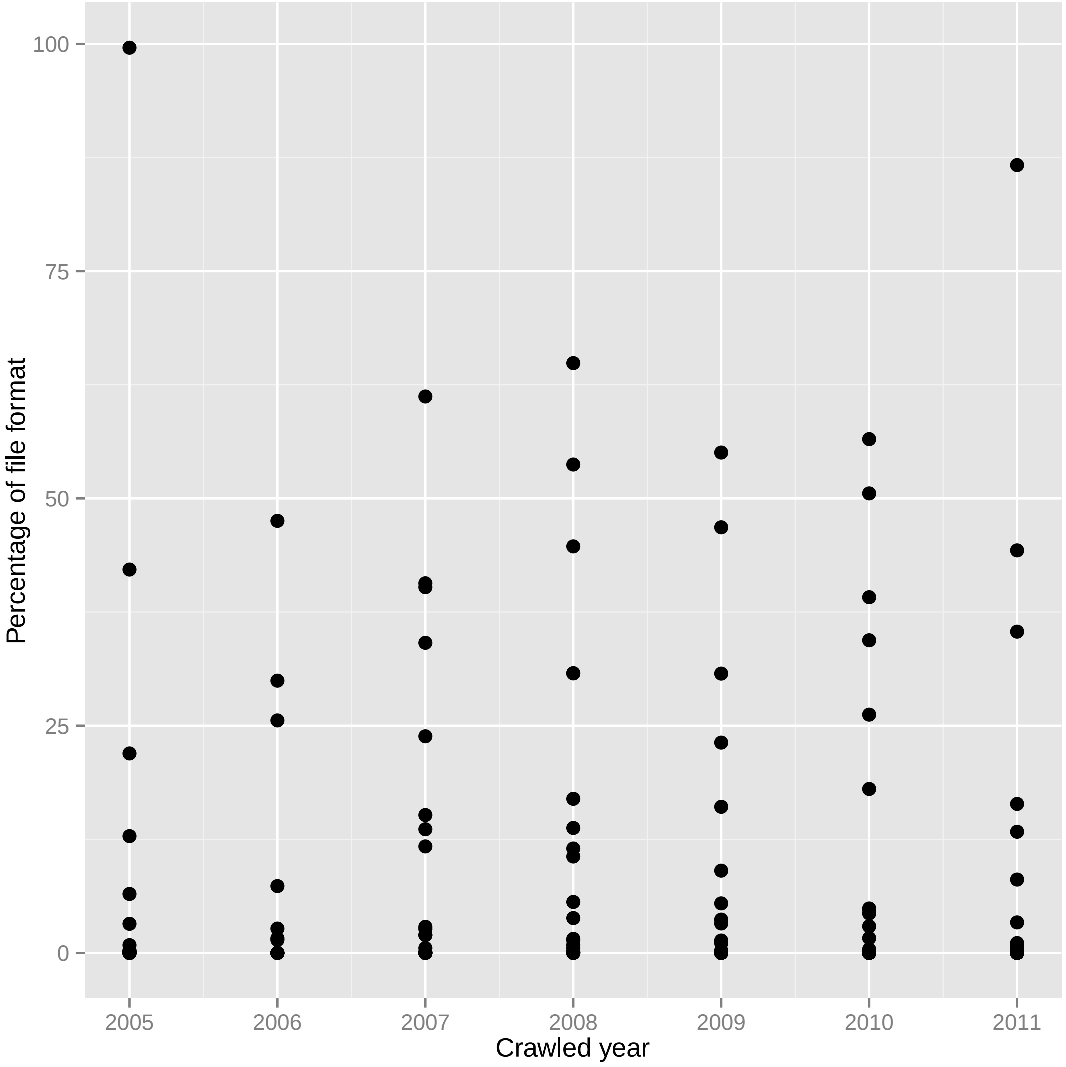


Figure 4.1: Scatterplot of version usage over all crawled years

others would falsify the results. An other constraint is, that only file formats, where at least 5 different version occurrences have been retrieved through the crawl, are used for the computations in this section.

A first basic approach is the creation of a scatterplot containing all the version percentages over the crawled years 2005 to 2011. Figure 4.1 shows such a scatterplot containing all versions of the main file formats that have been found through the crawl (PDF, Image files, HTML files, Flash files). The horizontal (x-) axis shows the crawled years from 2005 to 2011, while the vertical (y-) axis shows the usage rate of a specific version of a specific file format compared to all used versions of the according file format. The single dots in the diagram represent the usage of one version of a file format compared to all found versions of this file format in one year.

This figure already gives some indications on the usage of individual file format versions. In the range of 0 to 20% of usage, many data points exist, whereas the spreading is further increasing upwards. This allows the conclusion, that a few versions are always used very frequently, while a lot of others just make a small part of the whole usage.

The point that lies at nearly 100% in the first year (2005) is *text/html 4.01* (HTML version 4.01). More than 99% of all crawled HTML files had that version in 2005. This value could be misleading, because of the small sample size for HTML in the first year. The top ranked versions in 2011 are, again, *text/html 4.01* (about 44% of all crawled HTML files in this year) and *application/x-shockwave-flash 5* with a usage of more than 86% of all Flash files.

As this projection only takes into account the crawled years, there are indeed some hints to the development of single versions over the years, it is necessary, to also include the years since the official release of the versions, because it can be assumed, that the usage quite after release can not be put on a level with the usage 10 years after the release.

Figure 4.2 displays the development of selected versions over the crawled years, namely HTML 3.2 (marked by triangles), Flash 6 (marked by crosses) and PDF 1.6 (marked by filled dots). This figure shows, that the usage of Flash 6 decrease from 15% in 2007 to nearly 0% 4 years later, while HTML 3.2 and also PDF 1.6 were hardly used throughout the years, staying around 5% and slightly dropping in the last time. The year 2006 lacks data for HTML 3.2 and PDF 1.6, because these data had statistically significant derivations between training and test set and were therefore not used for the computations, otherwise the results would have been inconclusive. Even though different trends can be estimated through this figures, it gives no clue, if it is possible to compute a generally valid usage trend, because only a few years are known, and these don't suffice.

To be able to compute the development cycle over time, it is necessary, to increase the scale of the x-axis. It is changed from containing the crawled years 2005 to 2011 to containing the years since official release of the individual versions. As an example, the PDF version 1.7 was released in the year 2006 ([31]), so all crawled years are changed as follows:

- it is not possible, that such an item was crawled in 2005, except it has been misidentified
- an item, that has been crawled in 2006, is changed to year 0 (0 years after the release of this version)
- an item, that has been crawled in 2007, is changed to year 1 (1 years after the release of this version)
- items, that have been crawled in 2008 to 2011, are changed to year 2 to 5, respectively (2 to 5 years after the release of this version)

An overview of the release years for some common file format versions is shown in Table B.1. By aggregating the percentual usage for the years since release for the different versions of a specific file format, more data points are available. A fictitious example to clarify this process is given:

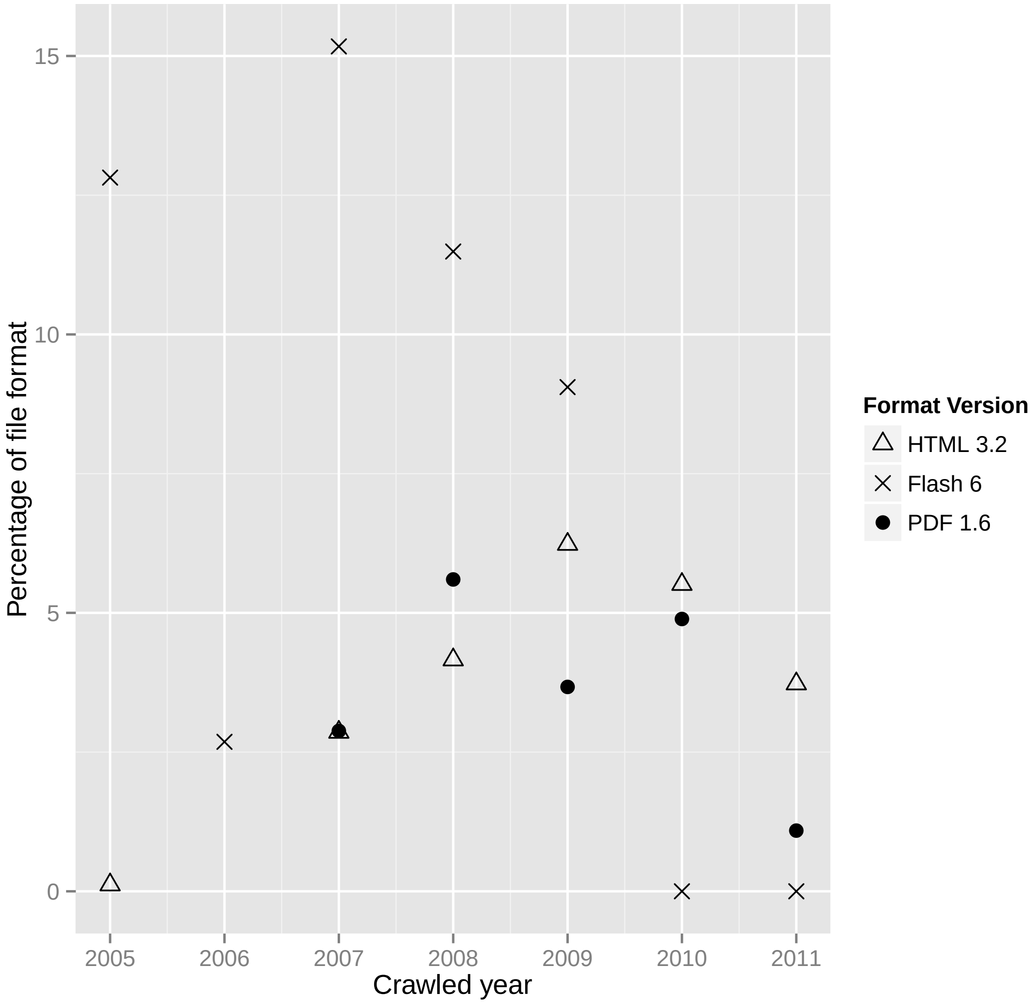


Figure 4.2: Scatterplot of selected file format/version combinations over all crawled years

- PDF 1.3 was officially released in 2000, so the crawled years 2005 to 2011 cover the years 5 to 11 since its release.

- PDF 1.5 was officially released in 2003, so the crawled years 2005 to 2011 cover the years 2 to 8 since its release.

- Let 15% of all crawled *application/pdf* files of 2005 be of version 1.3 (5 years after 1.3 was released)

- Let 20% of all crawled *application/pdf* files of 2008 be of version 1.5 (5 years after 1.5 was released)

- the aggregation of both versions means, that 5 years after a versions release 17,5% $((15\% + 20\%)/2)$ of all files of this format have this specific version.

- Let 45% of all crawled *application/pdf* files of 2008 be of version 1.3 (8 years after 1.3 was released)

- Let 40% of all crawled *application/pdf* files of 2011 be of version 1.5 (8 years after 1.5 was released)

- the aggregation of both versions means, that 8 years after a versions release 42,5% $((45\% + 40\%)/2)$ of all files of this format have this specific version.

Through the associated enhancement of x-axis points from 7 to about 20, considerably more indicators for a curve computation are available, which increases the accuracy and significance of this curve.

As 20 different x-axis data points are now available, the whole lifetime of a specific version can be taken into account, instead of only the 7 data points that were arising from the crawls of the 7 years (2005 to 2011). Now it is possible to determine via regression analysis, if a correlation between the years, that a version already exists, and the percentual usage of this version compared to all used versions of this file format, exists, to specify a formula that is able to forecast the development of newly released versions. Because the relationship between the x- and the y-axis is not linear, polynomial regression is used, in which the relationship is modeled as an *n*th order polynomial. The general formula for an *n*th order polynomial is given in Equation 4.6

$$f(x) = b_0 + b_1 \cdot x + b_2 \cdot x^2 + ... + b_n \cdot x^n \tag{4.6}$$

The computation of this polynomial regression is performed with the statistical computing program R. For each file format, all existing versions are collected and converted to the existence-year-view. All percentage values for all versions of these formats are plotted to a graph. If multiple y-values exist for one x-value (which will and shall be the case quite often), a mean of these values is computed. The resulting graph has at most one data point for each x-value. Then, the best fitting polynomial for the file format is found and the regression line is computed and plotted to the graph.

Figure 4.3 shows the regression lines of the three selected main file formats PDF, HTML and Flash. The x-axis in this graph represents the years since release, and the y-axis the percentual usage compared to all versions that were used of the corresponding file format. The dots represent all versions average percentual values for one year of existence. The line, that is drawn in each graph is the regression line that has been computed for these average data points.

The regression line is just an estimation of the real values. It is obvious, that the percentage values can never be less than 0% (negative), but as the regression line is a mathematically computed curve of *n* polynomials, best fitting it, leads to these values.

The three plots look very similar, especially the first and the third one, all versions beginning by 0% in year 0 after release and again dropping to 0% after about 13 to 16 years of usage (and

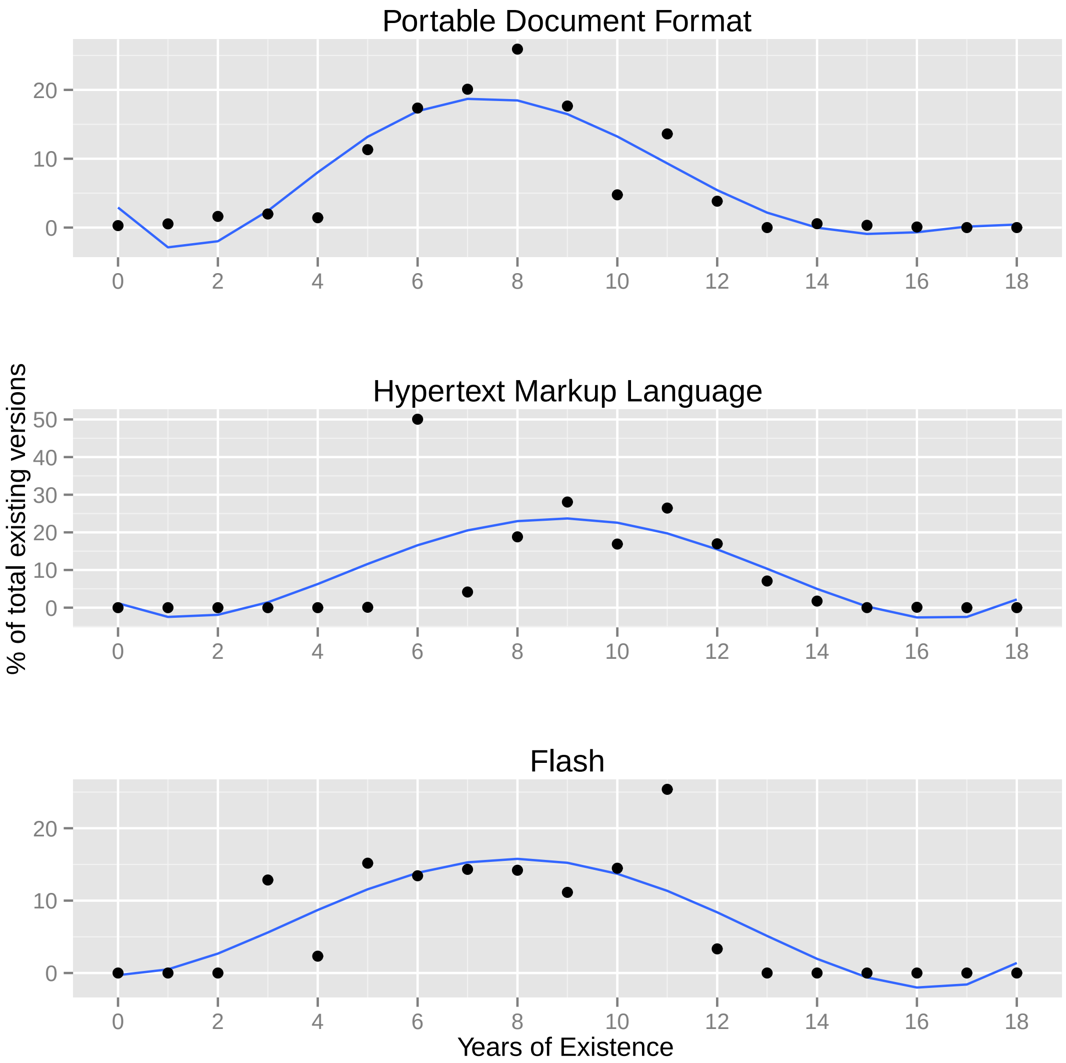


Figure 4.3: Regression analysis for selected file formats

thereby being replaced by newer versions). The versions are hardly used the first 2 to 4 years after release. After this introduction phase it increases, reaching the maximum of at least 25% somewhere around the 6th to 10th year and then slightly decreasing. The HTML plot has one outlier in year 6, that prevents the curve to accord with the other two, in other respects the trends are very alike. As a possible reason for this outlier, the small sample size for HTML in year 2005 could again be mentioned, as already pointed out previously in this chapter. Only 8 000 files are available in total for this year, and more than 99% of this year's found HTML files were version 4.01. 2005 was the sixth year of this versions release, which leads to this outlier.

The plot for Flash also has an outlier, in this case in year 11 after release. A possible reason

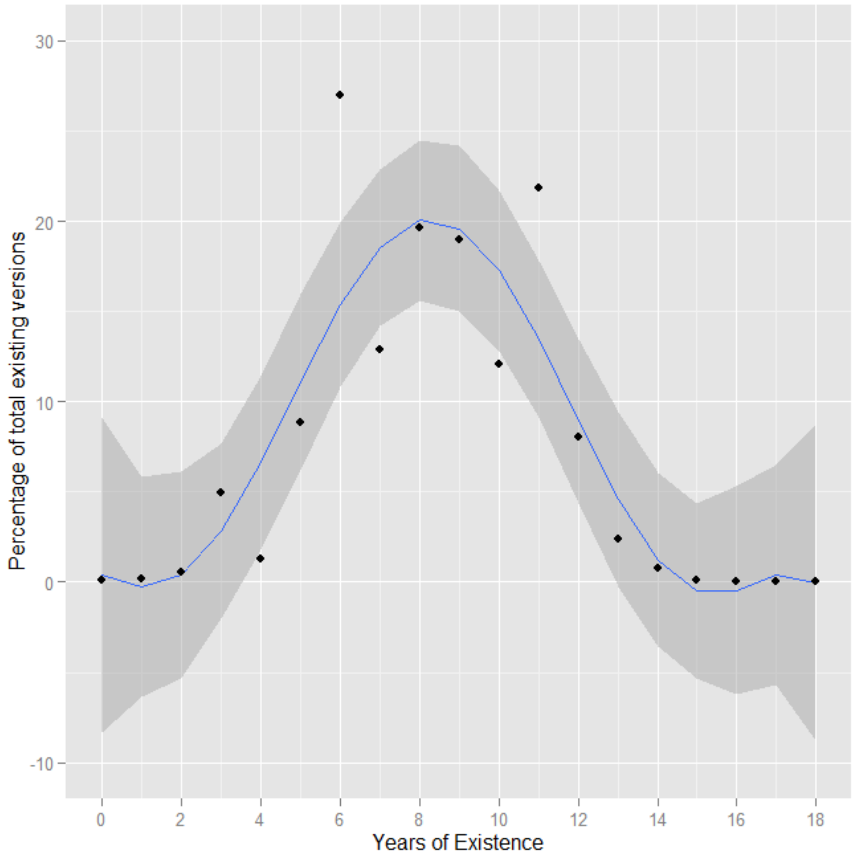


Figure 4.4: Regression analysis for the dataset

for this outlier is the very small absolute count of Flash files found in the crawl of 2011 (only 10 files could be classified with a specific version). More than 85% of 2011's Flash files had version 5, which existed the 11th year at this point, but as that little Flash files were existing in this year's crawl, this percentage could potentially lead to falsified assumptions.

The fact, that these plots look similar overall, motivates the try to aggregate all formats to one graph and thereby perform a polynomial regression for the whole dataset, without significantly changing the results of the computation. This aggregation is only possible, because the plots seem to have a similar acceptance rate. The aggregation is realized by computing the averages of all file format versions (again, only file formats, where at least 5 different versions, whose distribution differences between training set and test set aren't statistically significant, have been crawled over the years) for each existence year and performing a polynomial regression over these computed average values. These average values are shown in figure 4.4, including the 6th grade regression line, which is the most verisimilar trend for version development. The shaded area represents a 90% confidence interval for this regression line. Although the regression line drops beyond 0% after about 15 years, it is obvious, that the usage rate can't be less than 0%.

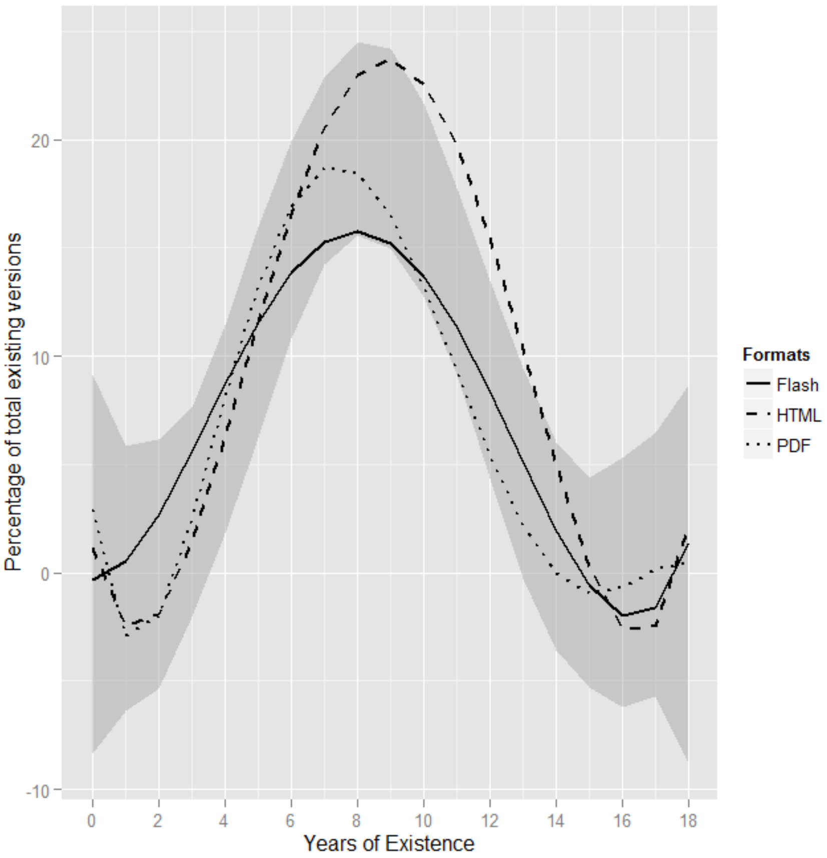


Figure 4.5: Confidence interval of the aggregated regression with the regression lines of selected formats

The version usage slightly increases to around 5% until year 4 since release and then receives a high increase, until it reaches the top usage of about 20% in the years 8 to 10. Then it drops rapidly to under 2% in year 14 and then totally disappears in the next few years.

To test the graphical fitting of the computed aggregated regression line to the single file formats, figure 4.5 shows the confidence interval of the aggregated regression line, along with the best fitting regression lines of some main file formats. The solid line is the regression line for the Flash format (media type: *application/x-shockwave-flash*), while the dashed line represents the regression line for HTML (media types *application/xhtml+xml* and *text/html*) and the dotted line for PDF (media type *application/pdf*).

As can be seen from this figure, nearly the whole file format's regression lines lie within the aggregated format's confidence interval, which leads to a high correlation between the aggregated and the file format's regression.

Summary of Version Usage

This section gave a generally valid development curve of file format versions. To do so the crawled data's grouping was altered from the year of crawl (2005 to 2011) to the number of years since the version was released (about 0 to 20 years). This increased the number of data points for the x-axis from 7 to around 20 and led to a better computation of the whole usage trend. To make the trend line more robust, only file formats with at least 5 different reported versions (by the crawler) were taken into account. Most heavily used crawled formats had more than four reported versions, except *image/png* and *image/gif*, which both had 4 different crawled versions, or *text/xml*, which had 3 different crawled versions.

Versions whose usage rate in the test set significantly differed from that in the training set were ignored, so the resulting curve can be used as a deduction to file format development in the world wide web.

At the basis of this data a polynomial regression was performed, the resulting regression line is first slightly, then rapidly increasing, until it reaches the top of about 20 to 25% after 6 to 10 years, and dropping rapidly to less than 5% in year 14.

This trend says, that it takes some years, before new versions are accepted. They are then used for a few years, before they get replaced by newer versions of the file format.

4.5 Textual formats

This section compares the web usage of the three main formats used to publicize and share textual information, along with formatting, graphics and tables, the *Portable Document Format* (called PDF in this section), the *Open Document Format* (called ODF in this section) and the *Microsoft Word Document* (called DOC). It also uses data set DS1 as the data source.

The *Portable Document Format*, a very common format, that's regularly used in the world wide web, is further described in section 4.6.

The Open Document Format is an XML-based file format, that is, amongst others, used for word processing documents. It was developed by a technical committee within the Organization for the Advancement of Structured Information Standards (OASIS) and provides an open alternative to proprietary document formats. ODF 1.0 was published in 2005, the current specification, ODF 1.1, was released in February 2007 and removed some compatibility issues. [57]

The Microsoft Word Document is a file format, that is mainly created by Microsoft Word, a proprietary word processor, that has first been released by Microsoft in 1983 [1]. First it was a binary format under the media type *application/msword*, that was replaced as the default format by an XML format with Microsoft Word 2007 [14].

For identification of the single formats, the following media types were used, that are registered at the IANA:

- *application/pdf* for PDF
- *application/vnd.oasis.opendocument.text* for ODF
- *application/msword* and *application/vnd.openxmlformats-officedocument.wordprocessingml.document* for DOC

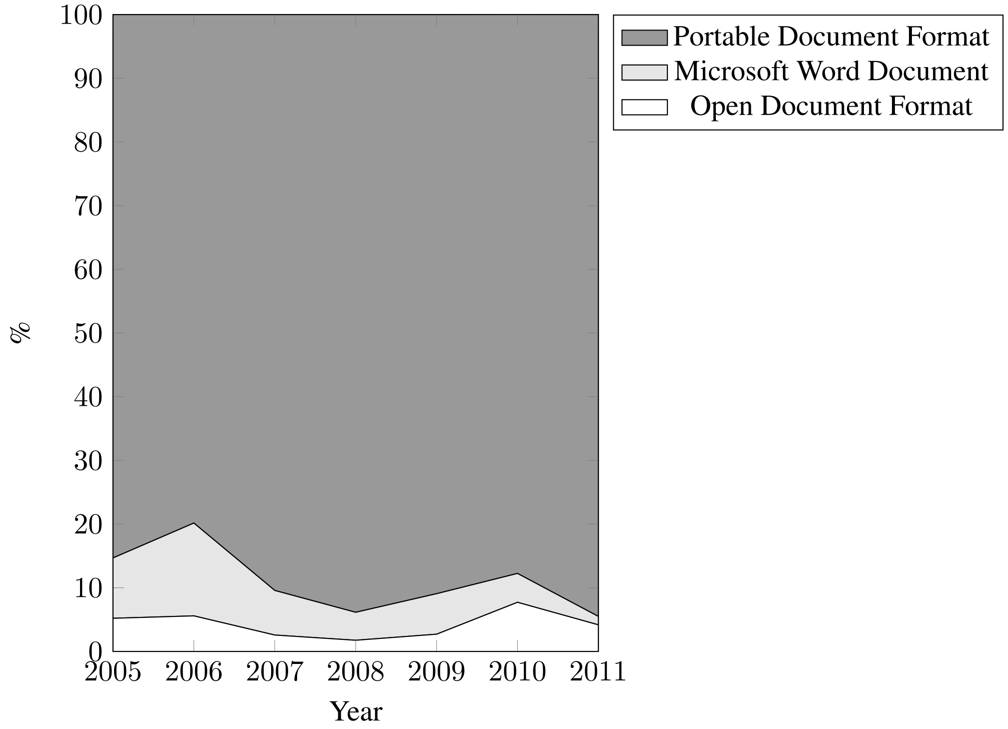


Figure 4.6: Development of textual formats found in the Training Set

The analysis of the FITS files in the training set led to an assumption on the development of theses formats as shown in Figure 4.6.

Figure 4.6 leads to the assumption, that PDF files were by far always used most since the year 2005, even increasing the usage rate from about 85% in 2005 to 95% in 2011. This hypothesis can be proven, except the break to 80% in 2006, which produces a slightly raising PDF usage each year and can lead to an even higher usage rate in the future.

The null hypothesis for the other two formats says, that DOC and ODF were both used much less than PDF in every year, with DOC being used more often than ODF until 2010, where their order changed. DOC started at about 9.5% in 2005, whereas ODF had about 4% less, being around 5%. This hypothesis can be proven in most terms. The developments for DOC and ODF are not that accurate, because the X^2 values for DOC 2006 and ODF 2008 are too high, so the values for these two years have statistically significant deviations. Despite this fact, the overall trend has been proven, as well as the facts, that both formats lost usage over the years, and that DOC has been overtaken by ODF in the year 2010. While the usage of ODF nearly stayed on one level, and just suffered a marginal decrease over the years, DOC dropped away relatively strong, which led to a usage of about 1.3% in the year 2011.

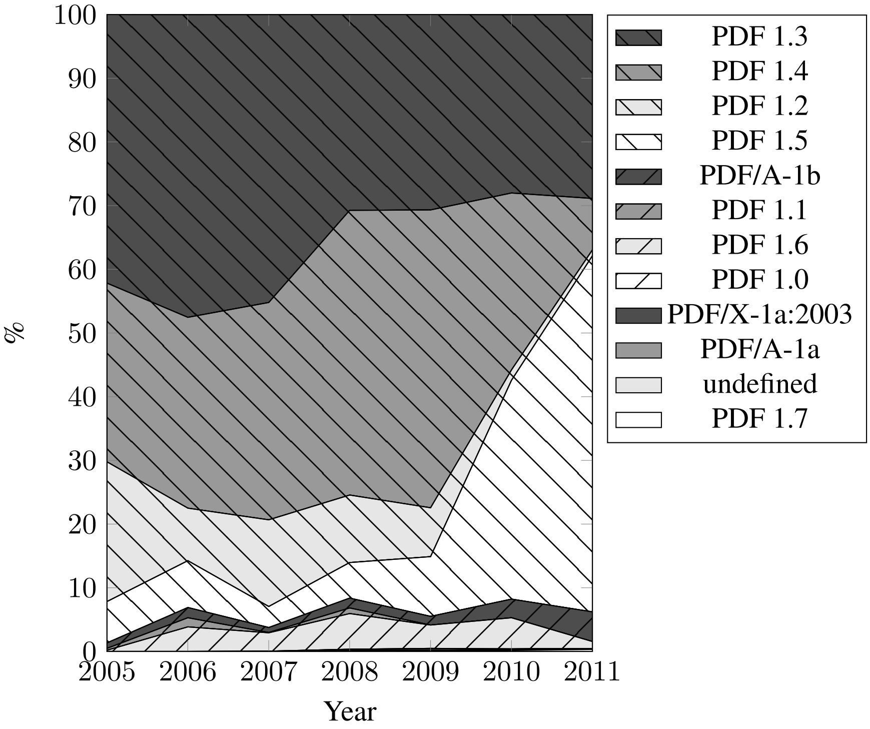


Figure 4.7: PDF Versions found in the Training Set

4.6 Portable Document Format

A very common format, that is regularly used in the world wide web, is *PDF*, the *Portable Document Format*, which is „a file format for representing documents in a manner independent of the application software, hardware, and operating system used to create them and of the output device on which they are to be displayed or printed.“ [31]

PDF was developed and first released by Adobe Systems[6] in 1993 as PDF 1.0. Subsequent releases have added new functionalities to the specification and PDF became the de facto standard for printable documents on the web. PDF is able to preserve and reproduce the original document appearance, including text, drawings, photos or charts. [9]

The IANA media type for PDF files is *application/pdf*, so this media type is used for identifying PDF files in the collection. The PDF versions, that were found in the training set over the years, are displayed in figure 4.7.

The collection consists of 11 different PDF versions, along with a small set of files, whose version could not be identified by FITS, because of missing meta information or corrupted blocks in the PDF file.

The data found in the training set leads to the following assumptions:

---

[6] `http://www.adobe.com/`

1. PDF 1.0, which was published in 1993, isn't used any more. It was still available until 2009 with about 0.1% of the whole PDF files, but fell to 0% since then.

2. PDF 1.1, which was published in 1994, did also totally disappear. The usage decreased from 1.4% in 2006 to 0% in 2009.

3. PDF 1.2 (published in 1996) was heavily used in 2005 with nearly 22%, it rapidly decreased to 8% the next year, increased again to 13.6% in 2007 and then continuous decreased to less than 1% in 2011.

4. The most used PDF version in the first crawled year (2005) was version 1.3 with over 42%, despite its much earlier publication in the year 2000. It even increased to 47.5% in 2006, and then fell to about 30%, where it is nearly stable since 2008. This version has a much higher usage as derived in Figure 4.4, but the process is still similar.

5. The usage of PDF 1.4 (2001) constantly increased from 28% in 2005 to 46.7% in the year 2009. It then sustained a downfall of 40% to only 8% usage in 2011.

6. PDF 1.5 is the most used PDF version in the last two years. It was published in 2003, and was crawled for about 6 to 9% between 2005 and 2009. It then heavily increased to 34% usage in 2010 and nearly 56% in 2011.

7. PDF 1.6 was not accepted until now. Its publication in 2004 followed a little usage of 0.19% after one year, then fluctuating between 2.8% and a maximum of 5.6% (in 2008), and falling to a new minimum of about 1% in 2011. This contradicts the trend derive in Figure 4.4.

8. PDF 1.7 is nearly not used nowadays. First published in 2006, it was never used until 2010, then the usage doubled from 0.16% to 0.31% in the year 2011.

9. PDF/A-1a: PDF/A was published in 2005 and is based on PDF 1.4. It is geared towards long term preservation and aims to create PDF files, whose visual appearance is made easily preservable over the years. Some restrictions for ensuring this are, for instance, the interdiction of references to external resources, who couldn't be accessible any more in the future, or the prohibition to embed audio or video files into the document. PDF/A defines two degrees of conformance: PDF/A-1a, and PDF/A-1b, with PDF/A-1a (full visual reproducibility, embedding of unicode text, structure regarding the content) being more restrictive than PDF/A-1b (only visual reproducibility is necessary). [18]

   PDF/A-1a was nearly not used in all the crawled years, despite a very little usage in 2008 of 0.06%.

10. PDF/A-1b, in contrast to PDF/A-1a, is regularly used through the years. It began with 0.86% in 2005 and increased, with small ups and downs, to 4.66% in 2011.

11. PDF/X-1a:2003 is an implementation of PDF/X, which defines another subset of the PDF standard, and contains information that are mostly relevant for printing purposes. It was

first found in 2008, representing 0.17% of all PDF files. This version stayed at this level until 2011.

12. There are hardly any undefined PDF files (these are files, that were identified as mediatype *application/pdf*, but whose specific version could not be defined by some tools).

To validate the null hypotheses, the training set's values are now compared to the test set's, as described in Section 4.2 on page 40.

1. The first hypothesis holds. PDF 1.0 is no more used in the web.
2. This hypothesis also holds. There is one outlier in 2007 with a X^2-value of 10.24 (with 0.05% in the training set and 0.81% in the test set), but that does not affect the fact, that PDF 1.1 isn't used since 2009.
3. The PDF 1.2-hypothesis holds. Its usage dramatically decreased from 22% in 2005 to less than 1% in the years 2010 and 2011. The year 2009 does not hold, with a X^2 value of 13.6121, but as the other years' values aren't in the critical area, approximate development has been proven.
4. This hypothesis partly holds. The development of PDF 1.3 can not be proven, because 3 of the 7 X^2 values have statistically significant derivations (2007: 45.21% to 34.89%, 2010: 28.01% to 20.93% and 2011: 28.88% to 15.37%). However, it can be said, that PDF 1.3 still occurs in at least 15% of 2011's PDF files, despite it's publication 12 years ago.
5. PDF 1.4's development could be mostly verified. The increase to over 45% by the year 2009 holds, but 2010 received a slightly higher X^2 value of 6.6813; the downfall to 8% in 2011 could again be verified.
6. The hypothesis, that PDF 1.5 is the most used version nowadays, also holds, it's increase from 2009 to 2010 could also be proved. Only 2011 receives a very high X^2 value of 45.9542 (55.90% to 74.61%), so the usage in 2011 could not be proved.
7. PDF 1.6: This hypothesis holds.
8. PDF 1.7: This hypothesis holds.
9. PDF/A-1a: This hypothesis holds.
10. The hypothesis concerning the development of PDF/A-1b mostly holds. The small ups and downs could not be proven totally, years 2007 and 2011 received a statistically significant derivation, with an X^2 value of 21.1302 in 2007 (0.78% to 2.99%) and an X^2 value of 20.5972 in 2011 (4.66% to 0.35%).
11. PDF/X-1a:2003: This hypothesis holds.
12. The hypothesis, that there are hardly any undefined PDF files also holds. As these files could also be refereed to as *unidentifiable*, this fact proves, that mostly all programs and tools for creating PDF files are able to set the necessary meta data.

Comparison to regression analysis

Comparing this chapter's X^2 - holding findings to the regression analysis for PDF and for the whole collection (see Figures 4.4 and 4.5) leads to some interesting observations: Some of the versions fit to the predicted trend (PDF 1.0, 1.1, 1.2 or 1.7) while other versions usage is higher (version 1.3, 1.5) or lower (version 1.6) than predicted. As the regression is computed by minimizing the sum of squared residuals (which is the difference between the true and the predicted value), this led to the computed regression lines, which shall be seen as a prediction of the development.

The fact that some versions do not match the predicted curve can have multiple reasons. A major cause is, that only 7 crawled years are available for the analysis. The availability of a longer time period would lead to a higher accuracy in the computed regression line. In this case the residuals would probably shrink, and the prediction for the single years could probably be more accurate. There exist some unimposed factors, that could have an affect on the usage trend of single versions. PDF 1.3, for example, had a much higher usage over all years than predicted. As the previous version (namely PDF 1.2) was released 4 years before 1.3, which is a very long period, and it did also invent some major new features, like the support of asian typesets, or the support for digital signatures, this could be a reason for a very frequent usage right from the beginning [30]. Other unimposed factors for PDF could be the closing of security holes by a new version or the release of new PDF processing software, that saves PDF files in specific versions.

Summary of Portable Document Format Analysis

This section tried to give an insight into the development of single PDF versions in the danish dataset by assuming distribution hypotheses through the training set and trying to prove their validity with data from the test set, which was mostly possible, and the results could therefore be used as a deduction to the PDF development in the whole world wide web.

PDF 1.0 and 1.1 aren't used anymore, PDF 1.2 decreased to less than 1% in 2011, so it can forecast, that it's usage will stop soon, and these files will no longer need to be supported by PDF viewing and editing tools. PDF 1.3, despite already publicized in 2000, is still used in at least 15%, and it actually overtook version 1.4, which fell to 8% in 2011. PDF 1.5 is the format, that is mostly used nowadays despite more detailed percentages can not be given in this case, much more than newer formats like PDF 1.6 and 1.7, that are nearly not used, with less than 1% usage in 2011.

The other found versions are not very common, PDF/A-1a and PDF/X-1a:2003 are hardly used, just PDF/A-1b was used in about 3% of the PDF files in 2010. A value for 2011 could not be proven, but a look on the other years give occasions to assume a relatively stable usage of 2% to 3% in the next years.

No other PDF versions than the discussed have been found in the training set, this could also be proved using the test set.

PDF 1.5 was introduced 2003, but it took 6 to 7 years, before it was heavily used, and is now the most used version by far. PDF 1.6 is still not in use, even though it was introduced in 2004, the same fact goes for PDF 1.7, which was introduced in 2006. The analyzed PDF data, as well as the general version trend line, that is computed in section 4.4 on page 44, indicate, that the newer PDF version's usage will increase in the next years, and they will displace the older versions (< 1.6) in some time.

4.7 (X)HTML

As HTML tags, that are a part of this thesis, were not available in the dataset that was used for the other computations (see DS1 on page 35), a new dataset including HTML tags was obtained (DS2, page 36), that was analyzed with FITS including the HTML Parser Tool (see Section 3.4). This set also contains data, that were archived by the Danish StatsBiblioteket. To be consistent over the whole HTML analysis, this dataset was not only used for the HTML tags, but also for analyzing the distribution of the different HTML versions over time.

This section outlines the different versions of all HTML and XHTML files that were found in this dataset. Unfortunately, for the year 2006 no (X)HTML data is available, so the results are only filled with data from 2005 and 2007 to 2011. HTML, the HyperText Markup Language is the main language for the display of online content, web pages and other information in a web browser. HTML is written in different tags (encapsulated in chevrons), which are interpreted by the browser, who displays the specified information.

HTML was first published as the RFC 1866 standard in 1995, as specification „HTML 2.0“ [7]. The IANA media type registered with HTML is *text/html*.

To make HTML more extensible and to highly increase the interoperability with other data formats, XHTML (Extendable HyperText Markup Language) was invented. It was first released as W3C recommendation „XHTML 1.0“ in 2000 [48].

> „XHTML 1.0 (this specification) is the first document type in the XHTML family. It is a reformulation of the three HTML 4 document types as applications of XML 1.0. It is intended to be used as a language for content that is both XML-conforming and, if some simple guidelines are followed, operates in HTML 4 conforming user agents.“ [48]

So, using XHTML, XML code is written, but it is restricted to a predefined set of elements (the elements, that are known from HTML). XHTML files are registered under IANA media type *application/xhtml+xml* [4].

Just as a quick recall of chapter 4.3, a lot of misidentified HTML files existed, which have been adjusted manually as follows:

- Files that have been misidentified as *text/html 1.0* (HTML 1.0) or *text/xml 1.0* (XML 1.0) are manually changed to the „correct“ XHTML 1.0 definition *application/xhtml+xml 1.0*.

- Files that have been misidentified as *text/html 1.1* (HTML 1.1) or *text/xml 1.1* (XML 1.1) are manually changed to the „correct“ XHTML 1.1 definition *application/xhtml+xml 1.1*.

After the described manual corrections, a first development diagram can be shown in Figure 4.8. As seen in the figure, each year except the first (2005) a lot of undefined HTML and undefined XHTML files have been found. This undefined files sum up to more than 60% of all

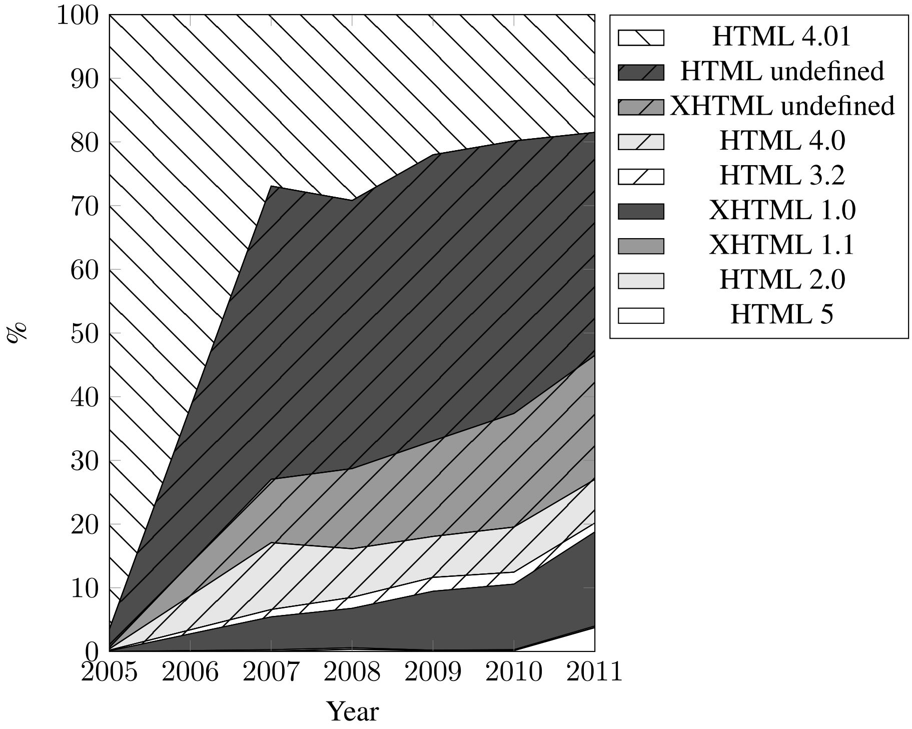


Figure 4.8: HTML Versions found in the dataset

found (X)HTML versions in some years. In this case, „undefined“ means, that the version of the file could not be parsed by FITS because of, most likely, missing or invalid doctype definitions inside the (X)HTML document.

This massive amount of (X)HTML files lacking a valid document type definition can arise from several reasons: many web sites are built with content management systems or other modular patterns, where scripts on the web server are called, that load parts dynamically, depending on user and browser requests. Such dynamically created sites usually consist of a header, a footer and the actual content, which are stored in different fragmentary (X)HTML files on the web server. Although these files are not valid (because of missing doctype definitions), the created file, that is sent to the browser mostly has a valid document type definition. Because the crawler did not only crawl „full“ (valid) (X)HTML files, but also these fragmentary files, a lot of undefined files exist.

So, figure 4.9 shows all found (X)HTML versions with the undefined files removed from the collection. In 2005, HTML 4.01 was the most used format by far. Although its usage decreased to about 41% in 2011, it was still the most used format. The older HTML 4.0 and HTML 3.2 versions are also still in use and stayed at around 20% (HTML 4.0) and 4 to 5% (HTML 3.2).

XHTML 1.0 and the later released XHTML 1.1 were never used in 2005. While XHTML 1.0 constantly increased to over 30% in 2011, which is the second most frequent format in this year, the newer XHTML version 1.1 stayed under 1% over all years, although it was intended to

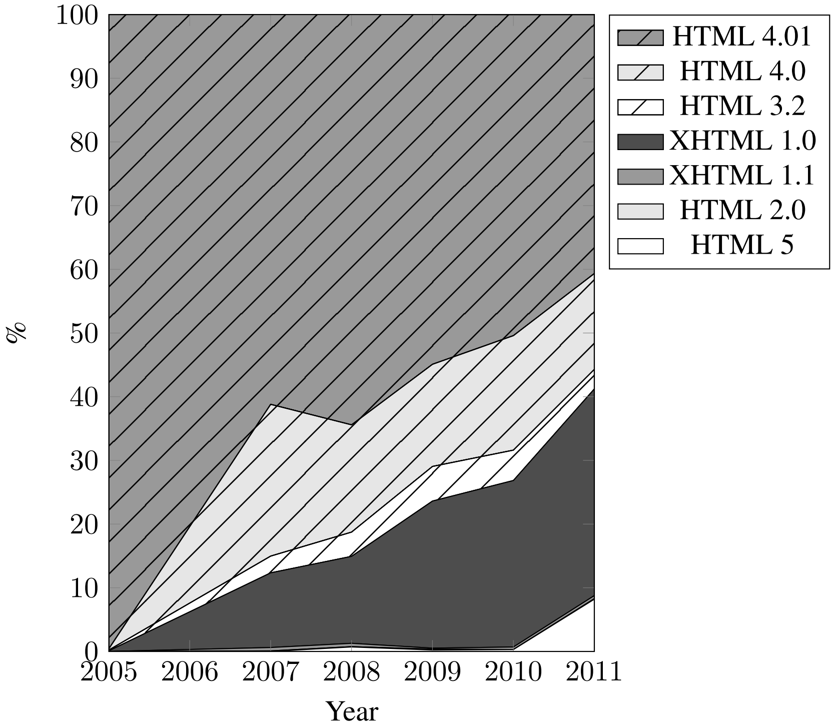


Figure 4.9: HTML Versions found in the dataset, undefined files excluded

replace XHTML 1.0. This may be explained by the very small syntax changes, that have been made between XHTML 1.0 and XHTML 1.1 [39].

A remarkable fact is, that HTML 5 was already used in 10% of all crawled (X)HTML files in 2011, although it has not been released as a standard until 2012, but only as updated working drafts. The first working draft was released in January 2007, and it seems as if developers try to be ready to use HTML 5 as soon, as it is standardized and therefore already use this new document type in one-tenth of all (X)HTML files [29].

Comparison to regression analysis

Similar to the PDF analysis (discussed in Chapter 4.6) , a deviation to the predicted regression lines exists for the found (X)HTML versions. The possible reason for this deviation are also similar to the PDF's reasons. Unimposed factors might exist, that led to altering trends for the single versions. Aas explained in the previous section, HTML 5 was already in use, although it has not been released as a standard until 2012. An unimposed factor for developers could be their intention to be „ready“ for this versions, and the availability of different browsers that already supported HTML 5.

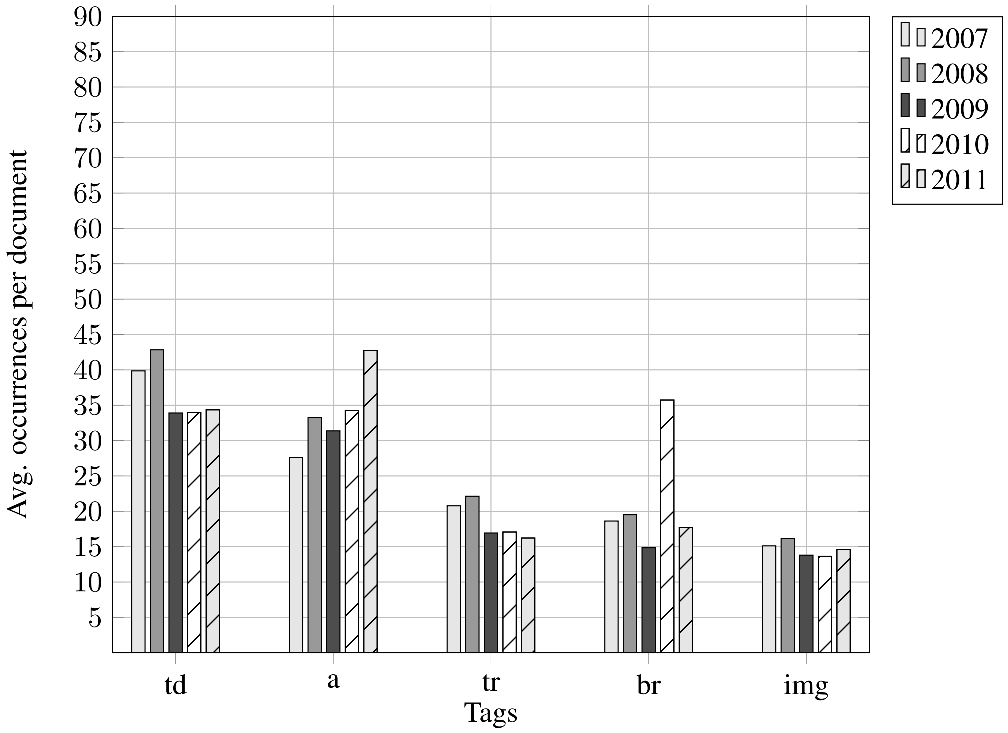


Figure 4.10: Evolution of the most used HTML Tags per file (average) for 2007

HTML Tags

This section explains different characteristics of the HTML tags, that have been found in the crawled collection. All findings have been verified through checking the test set's values.

The average top used HTML tags per file are displayed in Figures 4.10 and 4.11. The first diagram shows the tags, that were most used in 2007 and their development over the years, while in the second diagrams the five top used tags in 2011 are evaluated. One of the top tags over all years was <*a*>, which is usually used for hyperlinking to other web pages. In 2005, nearly 80 <*a*>-tags were used on an average crawled page. Although it decreased by nearly 50% to about 40 in 2011, it was still the most used tag. This immensive decrease and the small number of total (X)HTML files for the year 2005 tends towards an outlier. As the classified files for 2005 don't seem to be representative, only 2007 to 2011 are used for the analysis of this section.

The usage of the top tags was very similar in the years 2007 to 2011. The only tag that increased from year to year was <*div*>, whose average usage increased neary threefold from an average usage of about 11 in 2007 to more than 31 in the year 2011. This leads to the assumption, that nowadays much more data is aggregated and formatted by using <*div*> tags. Without going to much into detail, a possible reason for this cause could be the release of different frameworks like Bootstrap[7], that come with a lot of design templates to help web developers style their web

---

[7] `http://getbootstrap.com/`

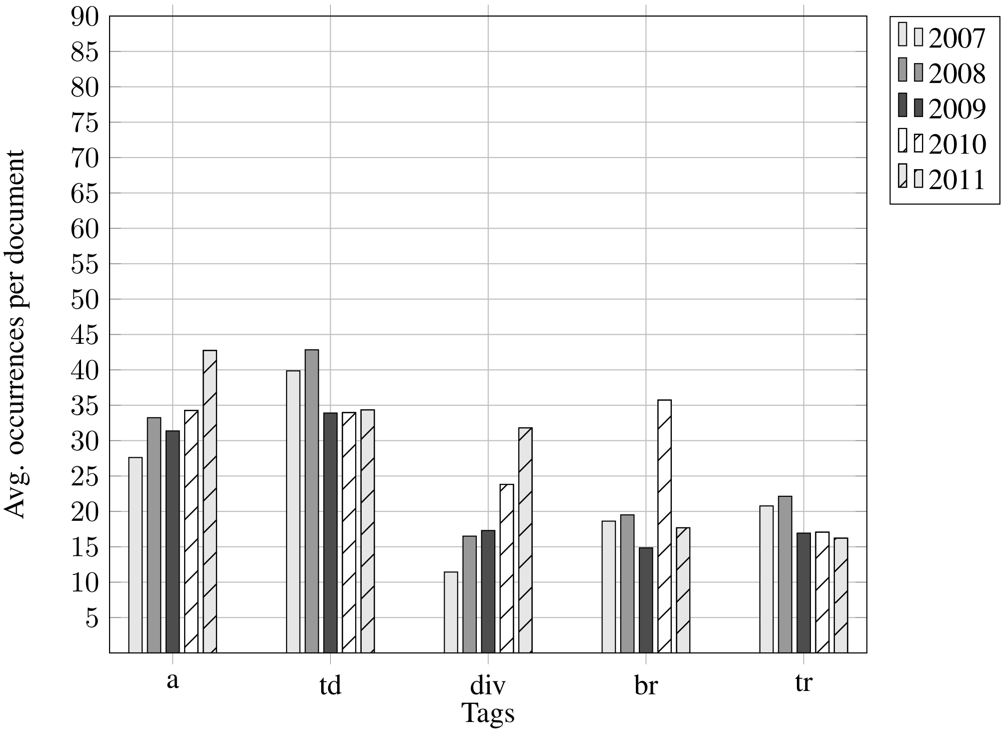


Figure 4.11: Evolution of the most used HTML Tags per file (average) for 2011

projects, and require the massive usage of <*div*> tags.

Average Tag Usage over all Years

Figure 4.12 shows some selected average tag uses per document, averaged over the years 2007 to 2011 (according to Equation 4.7).

$$avgUsage = \frac{\sum_{i}^{years} avgUsage_i}{\#years} \tag{4.7}$$

The top used tag is <*td*> with slightly less than 37 usages per document. The <*a*> tag follows second with nearly 34 usages per document, which shows the relevance and importance of hyperlinking in the web environment.

The <*script*> tag was used about 4.4 times per file. Its main usages are the loading of a script file or the identification of a script embedded in the page. The script runs on the client's machine, usually when the document is loading. Typically, the scripting language is JavaScript, but HTML does not rely on this and other languages, like VBScript or Tcl can also be used.

Although not displayed in the figure, the <*noscript*> tag should also be mentioned. This tag contains content, that is rendered, if the user agent (typically the browser) does not support

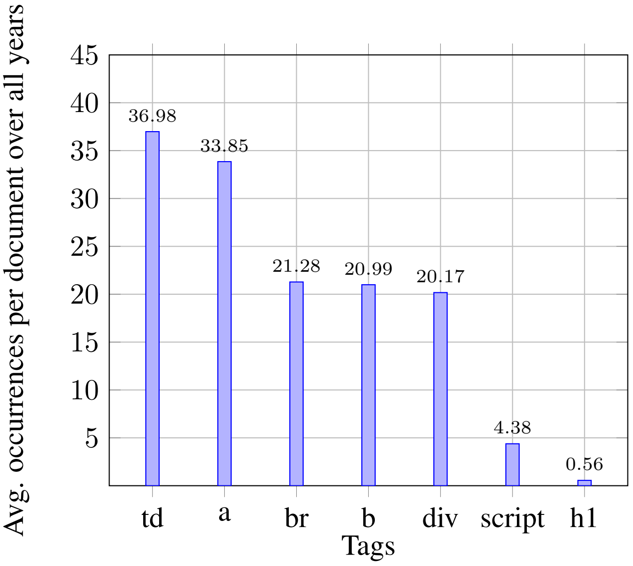


Figure 4.12: Selection of average tag usages over the years 2007 to 2011

scripting, or if scripts are deactivated (e.g. for security reasons) [52]. This tag has been used 0.41 times in average per file, which is much less than the <*script*> tag was used. It seems, that most web developers do not think about the necessity of using this tag, to make web pages navigable and usable although scripting is not enabled.

<*h1*>, the heading with the highest level, mainly used to structure web sites, was used 0.55 times per file, while the following, heading levels were used more (<*h2*>: 0.66 times, <*h3*>: 0.61) ,respectively less often (<*h4*>: 0.23 times).

HTML Encodings

This section gives a short overview of the found HTML character encodings in all crawled (X)HTML files. All hypotheses that have been derived from the training set could be confirmed by checking the corresponding values from the test set, so in this section only the confirmed values will be mentioned. In this context, character encodings are used to map the bit sequences transported over the world wide web to browser-displayable characters. One sequence of bits may map to different characters when using different character encodings.

The information about the character encoding were mostly generated by the Htmlinfo tool and Exiftool and could be read from the FITS XML files. In total, 27 different character encodings were found in the dataset, but most of them being used in less than 0.1% of all (X)HTML files. The three main encodings are ISO-8859-1 (an 8-bit character set for Western European languages), UTF-8 (a variable-length character encoding for Unicode) and windows-1252 (an 8-bit character encoding for Western European alphabets). The usage of these encodings, along with all other found encodings, summed up as „other encodings“ can be found in Figure 4.13.

The content encoding depends very much on the part of the world, the spoken language and

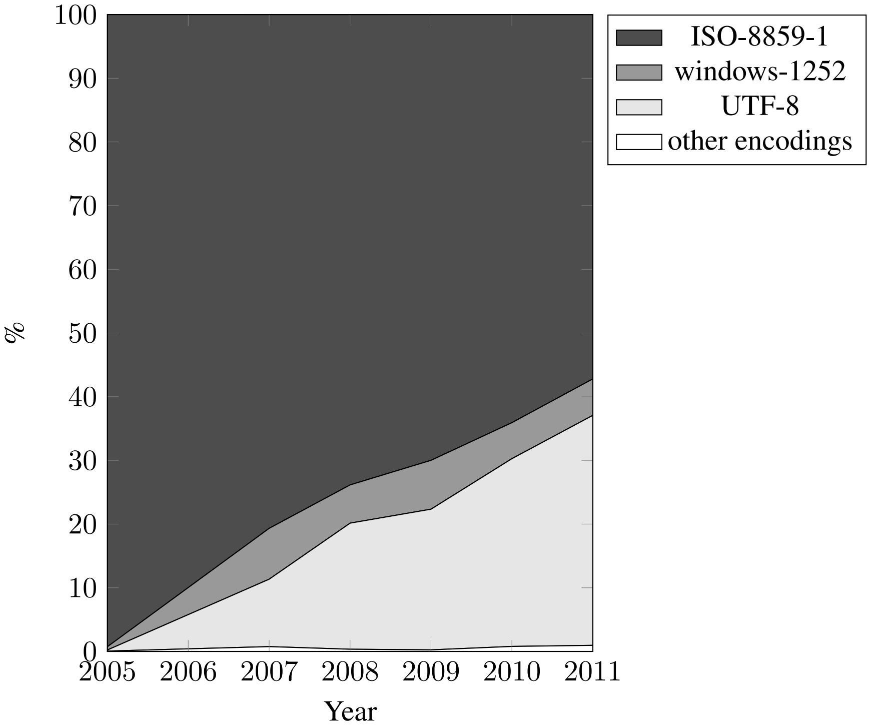


Figure 4.13: Mainly used HTML encodings

the used alphabet in the place where the HTML file was created. Chinese websites, for example, will use very different character encodings in most cases, so the findings here can only be seen as potentially valid for (Western) European countries. Nonetheless, there are some findings worth mentioning:

- ISO-8859-1 was used in more than 99% of the crawled files in 2005. Its usage constantly decreased by 5 to 10% each year, reaching about 55% in 2011.
- In contrast, UTF-8 usage increased. It lay under 1% in 2005 and increased by 5 to 10% each year to over 35% in 2011. UTF-8 was defined as an RFC standard in November 2003 [59], so it took a few years until the encoding was regularly used, but now it seems, that its usage will increase further and soon replace ISO-8859-1 as the most used character encoding.
- The usage of the windows-1252 character encoding was stable at 5 to 7% all years, which leads to the assumption, that it will also hold this usage in the next time.

Most of the other used character encodings can be separated into two types:

- Windows code pages, like windows-1256, which is used for Arabic letters, or windows-1251, used for Cyrillic languages.

- ISO/IEC standards. This family of encoding standards defines 15 different parts, that support different languages. The top used parts in this dataset were ISO/IEC 8859-15, used for Western European languages and ISO/IEC 8859-2, which is used for Eastern and South Eastern European languages.

4.8 Disappearing objects

This section focuses on objects (= (format,version)-pairs), that have disappeared in the crawled dataset (DS1) over the last few years. It is not impossible, that these objects are used again and re-appear, but as all of them have been replaced by a newer version, or by newer versions, the chance of happening is very little. Should these objects appear in other datasets, it is absolutely essential to take immediate preservation actions, because chances are, that they will also disappear soon there.

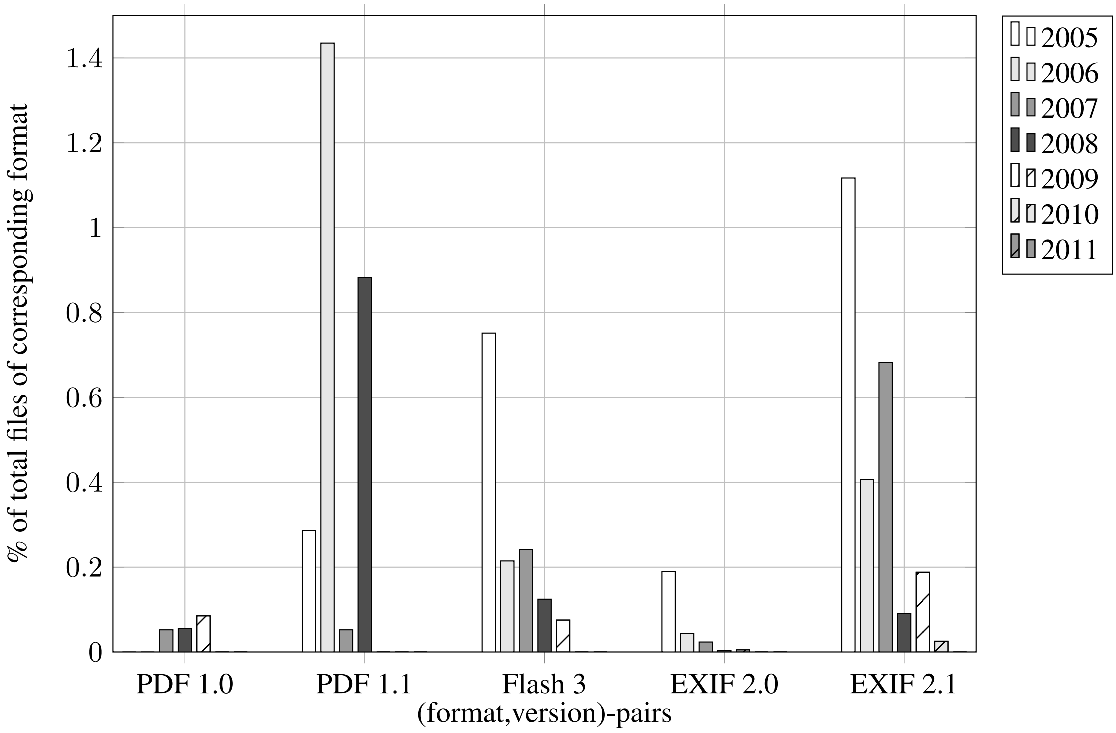


Figure 4.14: Objects, that disappeared in the analyzed years

The evolutions of these disappearing objects, that have been found in the crawled dataset, are displayed in Figure 4.14. The percentage values are computed by comparing the occurrences of the specific (format,version)-pair with the total files of the corresponding format in the current year. So, while the pair *(application/pdf, 1.0)* is compared with all *application/pdf* files, the pair *(image/jpeg, 2.0)*, for example, is compared with all found *image/jpeg* files.

- PDF 1.0 - (*application/pdf*, 1.0) has already been described in section 4.6. PDF 1.0 had already disappeared and wasn't crawled in 2005 and 2006. After returning in 2007 it was used in about 0.05% to 0.08% of all PDF files and it's usage stopped again in 2010.

- Like PDF 1.0, PDF 1.1 - (*application/pdf*, 1.1) has been described in the PDF section (4.6). It was used until 2008, with percentage values between 0.05% and even more than 1.4% in 2006. After 2008, PDF 1.1 wasn't used any more. It is possible, that in future years some PDF 1.1 files will be found, as PDF 1.0 did also return after disappearing, but this is very unlikely.

- Flash 3 - (*application/x-shockwave-flash*, 3), the SWF file format, version 3, is used to deliver (animated) vector graphics, video, text and sound over the internet. These files can be embedded and rendered in the web browser. SWF is designed to meet goals like simplicity, scalability, scriptability and speed [32].

  Flash 3 was released in 1998 and was constantly replaced by newer versions. A usage rate of nearly 0.8% in 2005 dropped year by year, until it completely disappeared in 2010.

- EXIF 2.0 and EXIF 2.1 - (*image/jpeg*, 2.0) and (*image/jpeg*, 2.1) is a standard that specifies formats for images, sounds and tags used by digital still cameras, intended to record technical details that are associated with digital photography. The Extendable Image File Format, version 2.0, was defined by the JEIDA, the Japan Electronic Industries Development Association in 1997. The next update, EXIF 2.1, was released in December 1998 [3].

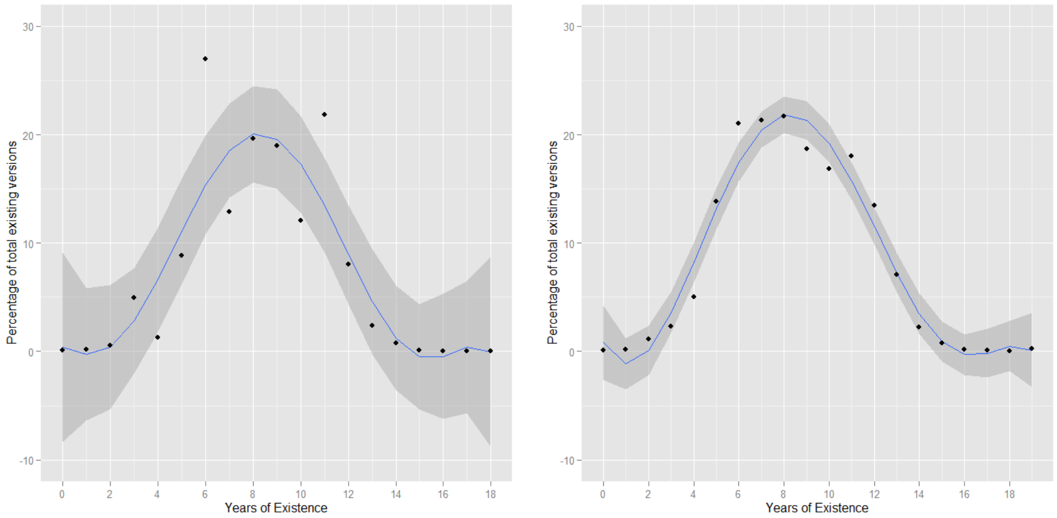
  EXIF 2.0 was used in 0.2% of all files with the *image/jpeg* media type in 2005. Then it nearly disappeared, with less than 0.05% in the years 2006 to 2009, and in 2010 it wasn't crawled any more. The slightly newer version, EXIF 2.1, started with a usage rate of over 1% in 2005. It decreased to 0.02% in 2010 and disappeared in 2011, a year later than the 2.0 version.

4.9 Comparison to Large Scale Study (DS3)

The results obtained through analyzing DS1 in the former sections can be compared to another, bigger, dataset, described as DS3 in section 4.1 on page 37. The queries, that were created for producing the results for DS1, can be reused to analyze this dataset and compare the results to the results presented in the former sections. The following large scale distributions do not show the obtained values for 2012, as they aren't available in DS1 and so cannot be compared.

A first approach is the comparison of the generalized development curve for file formats (which was explained and generated in Section 4.4). For both collections, three widely used file formats (PDF, HTML, Flash) were collected, the scale of the x-axis was increased by changing the years from the crawled to the years since release, and a polynomial regression was performed on the two collections independently.

This comparison is graphically shown in Figure 4.15, with DS1 on the left (4.15a), and DS3 on the right (4.15b) side. The lines in both figures show the polynomial regression lines, while the shaded areas



(a) DS1

(b) DS3

Figure 4.15: Regression analysis for the datasets compared

The generated regression lines look similar, both having their first raise after the second year, reaching the maximum at year 8 and then falling the next 8 years, until they disappear after 15 to 16 years. While DS1's regression line reaches a maximum of slightly less than 20%, DS3 reaches a peak of around 23%. Although the regression lines' similarity, the 90% confidence interval of DS1 is much wider than that of DS3. This could be explained by some outliers [8], that occurred (for instance showed by the dots in year 6 or 11 in 4.15a) in the smaller set (DS1).

After comparing the computed regression lines, the similarities of specific file formats are another interesting fact, that can be derived from these new data. Therefore, the PDF file format, whose different versions have been discussed for DS1 in Section 4.6, is used and compared to the distributions of these versions in the large collection (DS3).

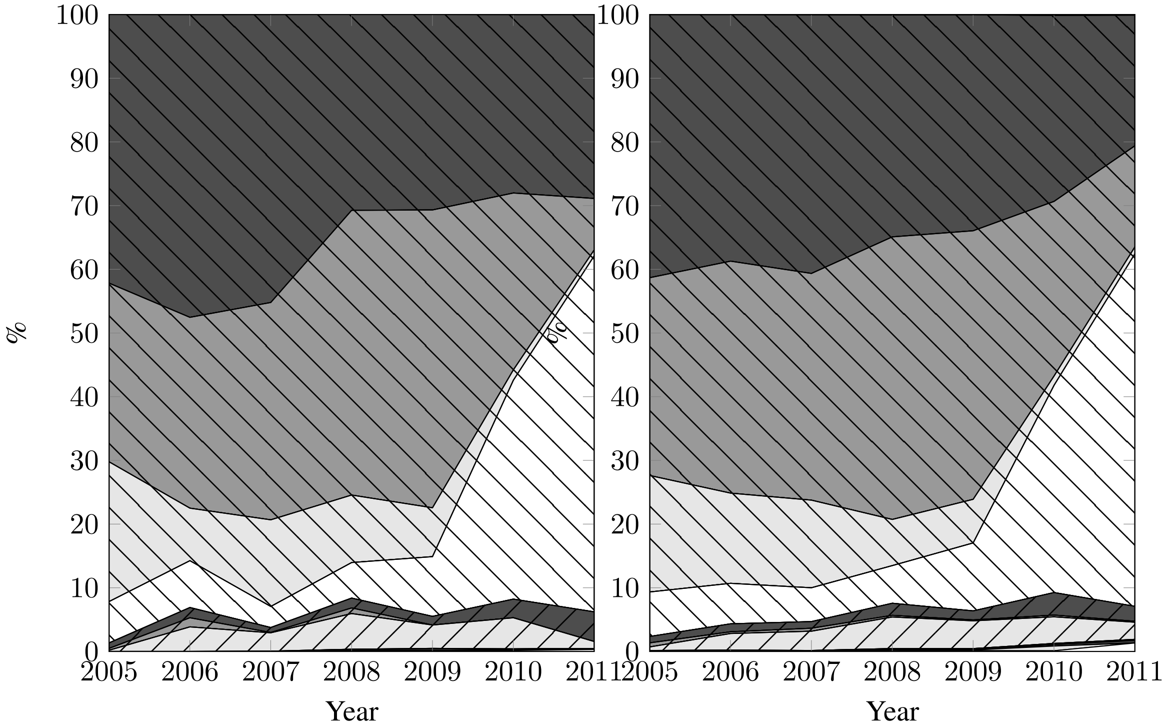
The distribution of the PDF versions found in data set DS1 has already been displayed in Figure 4.7 and can easily compared to the PDF versions found in DS3. Figure 4.16 shows a comparison between the collection, that has been analyzed in section 4.6 (Figure 4.16a) and the results that have been obtained through analyzing DS3 (Figure 4.16b). In this case, the x-axes represent the crawled years (2012 excluded, as it is only available for one collection), while the y-axes represent the percentual usage of the specific version of a file format, compared to the total usage of all versions of this format.

This figure shows very similar distributions through the two collections for nearly all PDF versions. Most versions (amongst others, PDF 1.0, PDF 1.1, PDF 1.5 or PDF 1.7) have nearly

---

[8] Grubbs (1969) defines outliers as follows:

> „An outlying observation, or "outlier," is one that appears to deviate markedly from other members of the sample in which it occurs." [23]



(a) DS1

(b) DS3

Figure 4.16: Comparison of PDF Version Distributions

the same distributions over all years. PDF 1.3 developed very similar until 2010, only the last year differs (21% in DS3 to 28% in DS1). In contrast, PDF 1.4, which is also very similar until 2010, was more often crawled in DS3 in 2011 (15% to 8%).

Although the comparison between two collections is not sufficient to give a generally valid usage trend for PDF, it leads to the assumption that the usage of PDF versions distributes very stable along different collections.

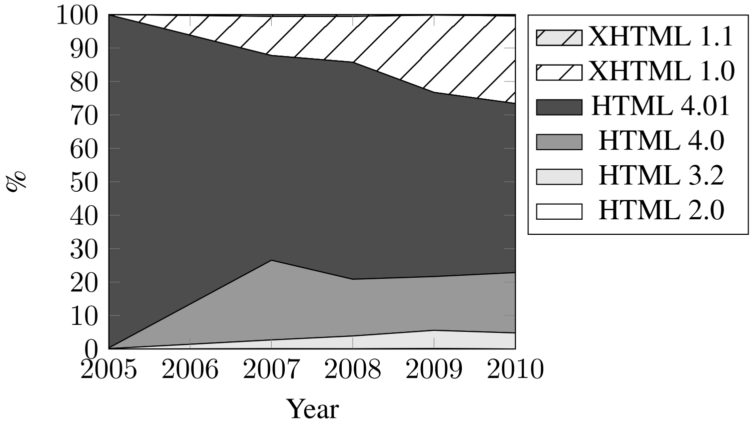
CHAPTER 5

Discussion and Conclusion

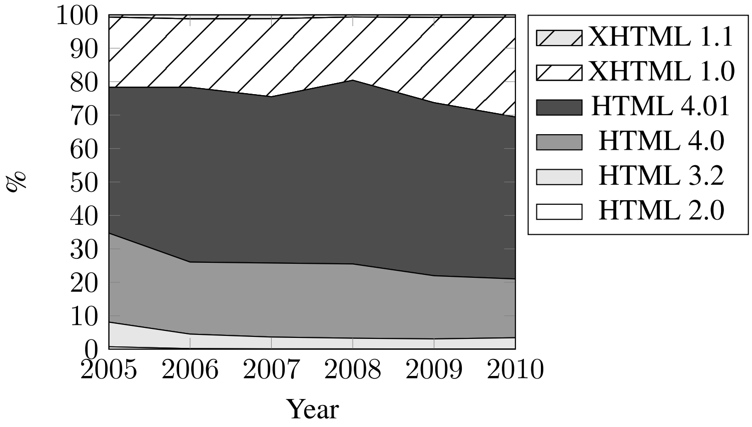
The following chapter discusses some of the results presented in chapter 4 and tries to compare them to other studies. Finally, the chapter draws a conclusion and provides an outlook to possible future work.

5.1 Comparison to other studies

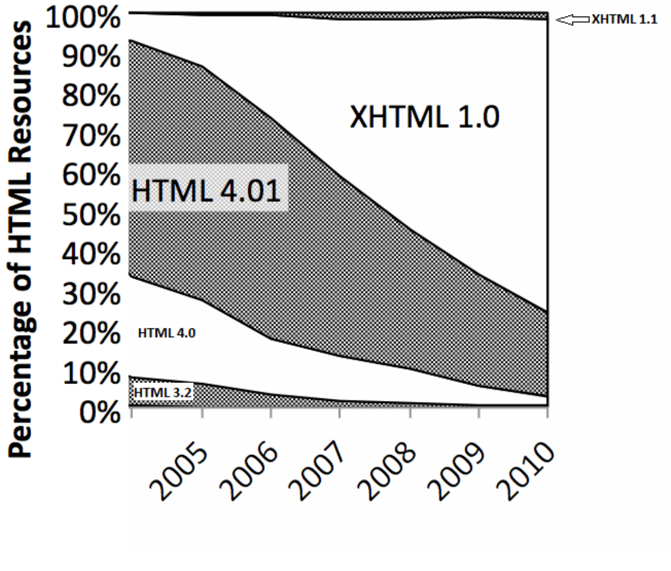
As the results of Jackson [35] overlap with my generated results for (X)HTML (DS2 - see section 4.7 on page 58), these findings can be compared. As an addition, the results are also compared to the generated (X)HTML distribution of DS3, and maybe a valid trend for different datasets (like UK and Danish) can be given. To be meaningful, the found HTML 5 data is not used in this comparison, as HTML 5 is not officially standardized yet, and that could be a possible cause, why it was not available in the UK Web Archive.



(a) DS2



(b) DS3



(c) UK Web Archive; Extracted from [35]

Figure 5.1: Comparison of HTML Versions Distribution

For this comparison, only the overlapping years for all three collections (2005 and the range from 2007 to 2010) can be compared, but as data for 2006 exist for at least two collections (DS3 and Jackson's results), these are also included in the figures. The displayed order of the versions in the diagrams were changed to match the order used by Jackson, which is, from top to bottom: XHTML 1.1, XHTML 1.0, HTML 4.01, HTML 4.0, HTML 3.2, HTML 2.0.

The HTML distribution, that was created from analyzing DS2 is displayed in the top Figure (5.1a), while the distribution of DS3 is shown in the middle (5.1b), and the essential years from Jackson's study on the bottom (5.1c).

Figure 5.1 displays big differences between the single collections, but also some similarities. One similarity is, that XHTML 1.1 is hardly used in all collections. As it was released as a W3C Recommendation in November 2010 ([39]), this fact is not really surprising.

The general replacement of HTML by XHTML is dramatically performed in the UK archive, with XHTML being used more and more per year, staring around 15% in 2005 until reaching estimated 70% in 2010. While the general XHTML usage also increased in the other two collections, this occurred far less. While collection DS2 (Figure 5.1a) saw an increase of around 25%, DS3's XHTML usage only increased from 20% to 30% in these 6 years.

The biggest difference in these sets, besides the raising of XHTML, is the usage of HTML 4.01. In DS2 it was the dominating format in 2005, with more than 99% usage. It then constantly decreased to about 64% in 2007 and finally 50% in 2010. Collection DS3(Figure 5.1b) even saw a little increase of the HTML 4.01 usage from 43% in 2005 to a top of 55% in 2008, and then 48% in 2010.

HTML 4.0, which decreased relatively linear in the UK Archive, until it's rate was around 5% in 2010, is was far more used in the other collections. In both of them, the rate was around 24% and slightly decreased to about 18% in 2010.

While HTML 3.2 nearly disappeared in Jackson's analysis, it was still used in the other collections in 2010, with a rate of about 3% to 4%.

HTML 2.0 has nearly disappeared in all collections. It is no longer visible in the graphs for all collections in 2010 (but it is still there, with less than 0.1%), and also hardly used in 2005 with a maximum of 0.7% in DS3 (Figure 5.1b).

Summarizing these findings, the differences between DS2 and DS3 are not very big, except the year 2005, where HTML 4.01 made nearly the whole HTML usage in DS2.

The differences from both sets to the UK Archive (5.1c) are much bigger. To explain this gap, more information about Jackson's analysis method would be needed, but he mentioned on his blog, that he used two tools, DROID and Apache Tika [1] for the identification of the files, while this thesis used the whole FITS, including the HTML Parser tool, which results in a total of 7 tools analyzing each file. This could be a possible reason, but as the difference between the collections is so big, this could also mean, that no common usage distribution for collections of different countries can be given.

---

[1] `http://tika.apache.org/`

5.2 Interesting Findings

This section gives an overview about some interesting findings, that have been derived through analyzing DS1's available data for this thesis, and may be of some interest to the reader.

The produced results can be set in contrast with Arlitt and Williamson (1996)'s study of web servers [2], if just the main IANA media types are taken into account. This may give an insight in the change of rates of the main media types over the last 15 years.

As a recapitulation, Arlitt and Williamson (1996) used 7 generic types (namely *HTML*, *Images*, *Sound*, *Video*, *Dynamic*, *Formatted* and *Other*). As the type *Dynamic* seems to be unclear, and it's memberships are inadequately documented in their work, the type is removed for this comparison, and new percentages are created.

The results, that are presented in chapter 4 are mapped to best fitting these generic types, so the best possible comparison is guaranteed, as follows:

- ***HTML*** - this generic type contains exactly two media types, *application/xhtml+xml* and *text/html*, as they are the only valid media types for HTML files.
- ***Images*** - this generic type contains all media types, that belong to in the main IANA type *image*, like *image/png* or *image/gif*.
- ***Sound*** - this generic type contains all media types, that belong to the main IANA type *audio*, like audio/midi.
- ***Video*** - this generic type contains all media types, that belong to in the main IANA type *video*, like *video/quicktime*.
- ***Formatted*** - this generic type contains all kinds of formatted documents, like *application/pdf* or *application/vnd.ms-powerpoint*.
- ***Other*** - this generic type contains all media types, that did not fit one of the other types. It contains, for instance, media types like *application/x-rar* or *text/x-c*.

The resulting diagram is presented in Figure 5.2. The usage of HTML files increased from 41 to 46%, while image files were much less used (decreasing more than 16%). In contrast, formatted objects did increase by more than 12%, while the other file types stayed relatively stable. Although this comparison is not significant, as there exist a lot of video and image hosting platforms, where the percentage rates seem to be completely different, and not all files have been assigned adequately to the media types, these findings are still interesting.

Some of the generated and analyzed data for DS2 can also be compared to the works of Woodruff et al. [58] and Nanavati et al. [42], who worked, amongst others, on the distribution rate of specific HTML tags in HTML documents.

Figure 5.3 shows these results. The tags are compared for the years 1996 (Woodruff et. al. [58]), 2004 (Nanavati et. al. [42] and 2011 (DS2). A lot of the common tags, like `<body>`, `<title>` or `<html>` decreased by 5 to 10% since 2004. This could be explained by the massive usage of templates and templating engines in the last years. Other tags, like `<div>` or `<li>` increased, maybe through the invention of new CSS functionalities.

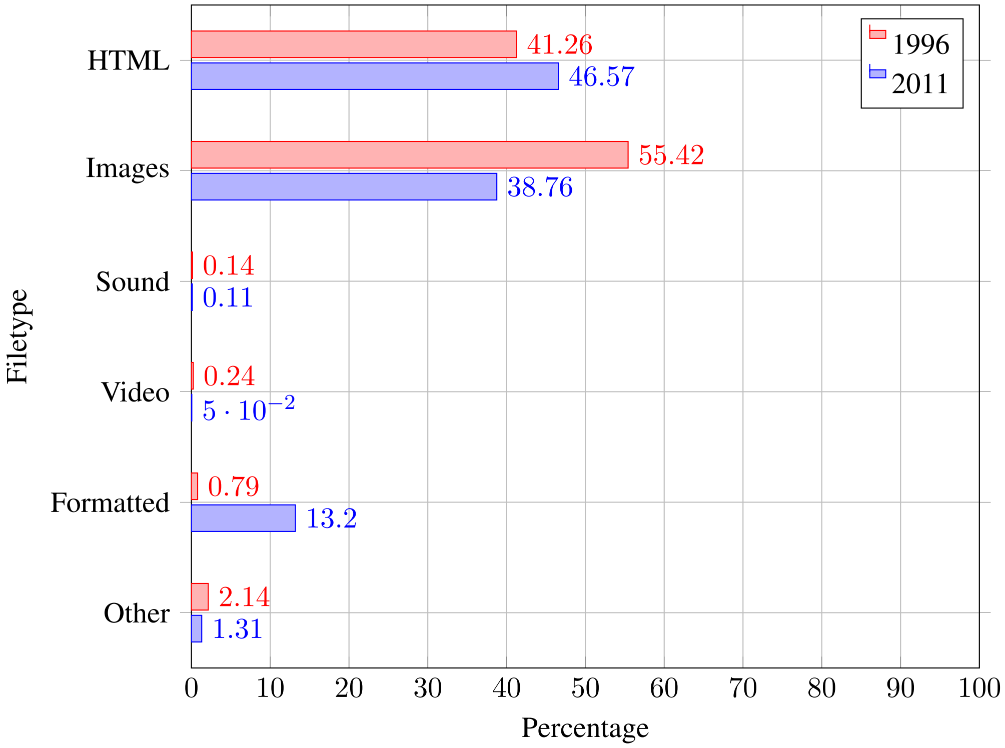


Figure 5.2: Comparison of Arlitt and Williamson's results [2] (1996) and the new found results (2011)

This thesis also analyzed tags from 2005. These tags are not displayed in the figure, but as the results are very similar to Nanavati et. al.'s from 2004, it can be asserted, that the overall structure was nearly the same in the files found in Denmark (this thesis), and those found through a representative crawl by using the Stanford WebBase project [33]. This leads to the assumption, that the usage of HTML tags is nearly the same in many different countries and cultures.

5.3 Conclusion and Future Work

This thesis analyzed digital preservation and web archiving techniques, as well as some previous studies on the distribution of file formats and on the usage of tags inside HTML documents, that were done in the last years. Although several studies existed, none of these covered all necessary parts (longer time series, valid identification, sufficient granularity) to present the distribution of different versions of specific file formats over a couple of years.

As such data can be a necessary information for understanding and predicting file format obsolescence, this thesis analyzed file format distributions over a longer time series and created a generally valid life cycle predictor for file format versions.

For this purpose, a framework was proposed, that uses some external, publicly available tools, and is able to crawl, identify and analyze parts of the world wide web in 3 steps:

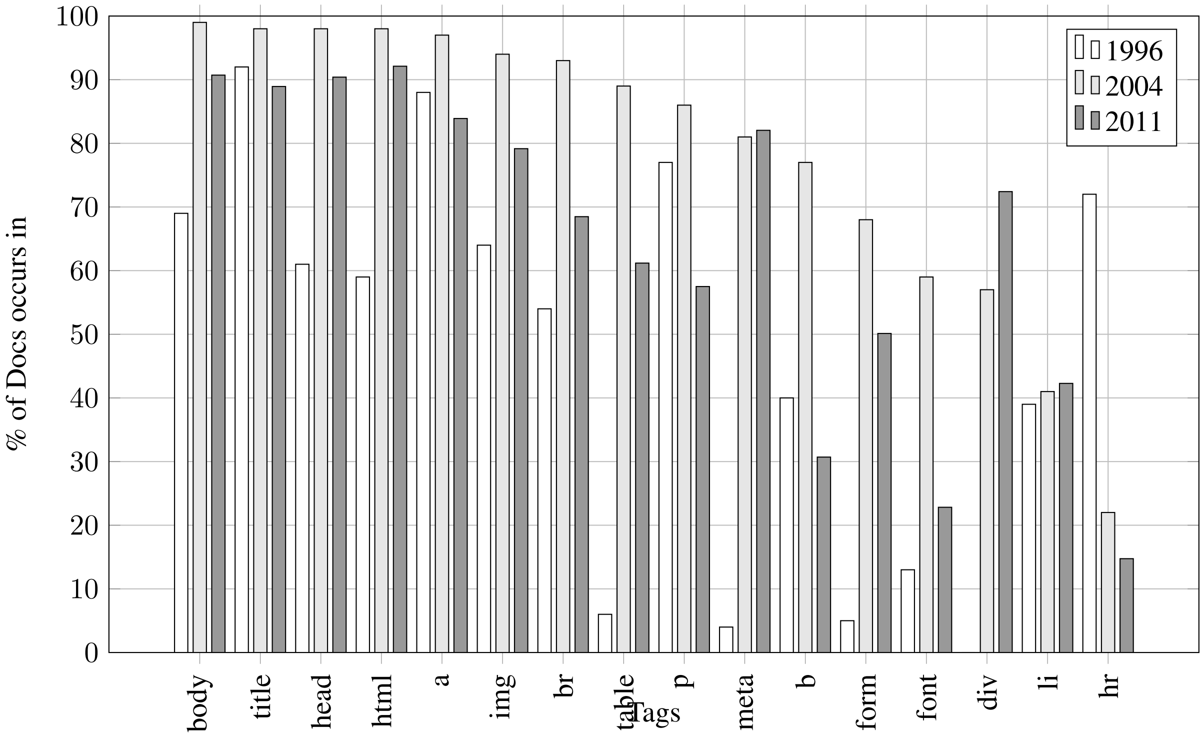


Figure 5.3: Comparison of Documents containing specific Tags, 1996 [58], 2004 [42] and 2011 (DS2)

1. Crawl specific parts of the world wide web, filter the necessary information out of the created WARC files, and reconstruct the single files out of the WARC container
2. Identify the created files with FITS, parse the created XML files with c3po and save them to a NoSQL database
3. Query the database with *map/reduce* and use R to compute statistics of the resulting data

The HTML Parser Tool, that has been created for this thesis, turned out to be a powerful tool than can support the HTML identification of the File Information Tool Set in terms of counting tags, identifying encodings or analyzing internal and external links, and does not slow down the whole FITS identification process, by just taking 10% to 15% of the whole time FITS needs for one file.

The datasets used for the analysis were achieved from the StatsBiblioteket (The State and University Library) in Aarhus, Denmark [2], as they had already crawled the Danish web in the last years, and so, data from 7 to 8 years was available for this thesis. It turned out, that a longer time series was needed to analyze the trend of the usage of file format versions (which is the usage of a specific version of a specific file format compared to the usage of all versions of this

---

[2] `http://www.statsbiblioteket.dk/`

file format), so the data was transformed to comprise the years since the single versions have been released.

As a result, an aggregated curve for some common used file formats was computed, that can be used as the origin for future work and for comparing other file formats or other data sets to this curve. It was shown, that a specific version of a file format is used at least 15 years, although the heavy usage covers 6 to 10 years (around 4 to 13 years after release).

Additionally, some common file formats, like PDF, the Portable Document Format, or HTML, were analyzed in detail. Some findings were, that PDF was distributed very equally in different collections, which can offer valuable clues, but to prove this as a generally valid fact, the analysis of more different data sets is necessary. The findings for HTML differ from findings that were derived from other datasets, so no valid assumption for the distribution of HTML versions can be given; again, future work on other datasets will be required.

An interesting task for future work would be the comparison of disappearing versions of file formats to the availability of tools for the creation and maintenance of these versions. Does the usage of file formats versions rapidly decrease, when newly released tools lack support for these versions, or do these tools support older versions, as long as they are still in use?

As some file format versions, that disappeared in the dataset, were discovered, it would be interesting to find, if these objects are still supported by newly invented or updated tools. This does also deal with the question, how long it takes for objects to disappear, when supporting tools are no longer available. A concrete issue for this context would be: If a new version of Adobe Acrobat [3] is released, that saves files as a new PDF format by default, does this lead to heavily increased use of this version?

---

[3] Adobe's proprietary software for creating, editing and converting PDF files

APPENDIX A

Code

A.1 Example FITS File

Listing A.1 displays an example XML file that was created from FITS by analyzing a PDF document. This file contains, amongst others, the tools that were used for the analysis, the versions reported by the tools, and additional file informations and metadata.

```
<?xml version="1.0" encoding="UTF-8"?>
<fits xmlns="http://hul.harvard.edu/ois/xml/ns/fits/fits_output" xmlns:xsi="
    http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://hul
    .harvard.edu/ois/xml/ns/fits/fits_output http://hul.harvard.edu/ois/xml/
    xsd/fits/fits_output.xsd" version="0.6.0" timestamp="11.06.12 11:45">
  <identification>
    <identity format="Portable Document Format" mimetype="application/pdf"
        toolname="FITS" toolversion="0.6.0">
      <tool toolname="Jhove" toolversion="1.5" />
      <tool toolname="file utility" toolversion="5.03" />
      <tool toolname="Exiftool" toolversion="7.74" />
      <tool toolname="Droid" toolversion="3.0" />
      <tool toolname="ffident" toolversion="0.2" />
      <version toolname="Jhove" toolversion="1.5">1.5</version>
      <externalIdentifier toolname="Droid" toolversion="3.0" type="puid">fmt
          /19</externalIdentifier>
    </identity>
  </identification>
  <fileinfo>
    <size toolname="Jhove" toolversion="1.5">275031</size>
    <creatingApplicationName toolname="Jhove" toolversion="1.5">Microsoft
        Office Word 2007/Microsoft Office Word 2007</creatingApplicationName>
    <lastmodified toolname="Exiftool" toolversion="7.74" status="
        SINGLE_RESULT">2012:06:11 11:43:14+02:00</lastmodified>
    <created toolname="Exiftool" toolversion="7.74" status="SINGLE_RESULT">
        2012:03:14 14:57:20+01:00</created>
    <filepath toolname="OIS File Information" toolversion="0.1" status="
        SINGLE_RESULT">D:\TU\temp-files\Studien_an_der_TUWien.pdf</filepath>
    <filename toolname="OIS File Information" toolversion="0.1" status="
        SINGLE_RESULT">D:\TU\temp-files\Studien_an_der_TUWien.pdf</filename>
    <md5checksum toolname="OIS File Information" toolversion="0.1" status="
        SINGLE_RESULT">9563b1d778b3e3fba5b3cf1ad348def8</md5checksum>
    <fslastmodified toolname="OIS File Information" toolversion="0.1" status=
        "SINGLE_RESULT">1339407794495</fslastmodified>
  </fileinfo>
  <filestatus>
    <well-formed toolname="Jhove" toolversion="1.5" status="SINGLE_RESULT">
        true</well-formed>
    <valid toolname="Jhove" toolversion="1.5" status="SINGLE_RESULT">true</
        valid>
  </filestatus>
  <metadata>
    <document>
      <title toolname="Jhove" toolversion="1.5">STUDIENRICHTUNGEN AN DER
          TECHNISCHEN UNIVERSITAET WIEN</title>
      <author toolname="Jhove" toolversion="1.5">POUSEK  Wolfgang</author>
      <language toolname="Jhove" toolversion="1.5" status="SINGLE_RESULT">de-
          AT</language>
      <pageCount toolname="Jhove" toolversion="1.5">2</pageCount>
      <isTagged toolname="Jhove" toolversion="1.5" status="SINGLE_RESULT">no<
          /isTagged>
      <hasOutline toolname="Jhove" toolversion="1.5" status="SINGLE_RESULT">
          no</hasOutline>
      <hasAnnotations toolname="Jhove" toolversion="1.5" status="
          SINGLE_RESULT">no</hasAnnotations>
      <isRightsManaged toolname="Exiftool" toolversion="7.74" status="
          SINGLE_RESULT">no</isRightsManaged>
      <isProtected toolname="Exiftool" toolversion="7.74" status="
          SINGLE_RESULT">no</isProtected>
    </document>
  </metadata>
</fits>
```

Listing A.1: An example FITS file

A.2 C3PO Output

Listing A.2 shows the c3po output for a HTML file. All information obtained from the XML file created by FITS are parsed into a JSON like structure.

```
{
   "_id" : ObjectId("5180f0aee4b028d73662572c"),
   "name" : "120-9-20050704082337-00080-kb-prod-har-002.kb.dk.arc:66088053:
       text-html",
   "uid" : "/net/halley/scape/workingbjarne-fits-tests/runall
       -9/120-9-20050704082337-00080-kb-prod-har-002.kb.dk.arc-org
       /120-9-20050704082337-00080-kb-prod-har-002.kb.dk.arc:66088053:text-
       html",
   "collecton" : "nf2005",
   "metadata" : {
      "format_version" : {
         "status" : "OK",
         "value" : "4.01",
         "sources" : ["5ab705ec-154c-4785-bbbd-883c1adffbae"]
      },
      "puid" : {
         "status" : "OK",
         "value" : "fmt/100",
         "sources" : ["5ab705ec-154c-4785-bbbd-883c1adffbae"]
      },
      "format" : {
         "status" : "OK",
         "value" : "Hypertext Markup Language",
         "sources" : "005358b1-70ae-483e-a7bf-88fa5b0741ff",
         "ddfadb31-f9f0-455c-a9ae-99307fa4905c",
         "5ab705ec-154c-4785-bbbd-883c1adffbae",
         "6b5c7807-20384752-95dc-5110acc34ea6",
         "2e815e89-1b69-4bd8-ab35-86cf00f835ba"]
      },
      "mimetype" : {
         "status" : "OK",
         "value" : "text/html",
         "sources" : ["005358b1-70ae483e-a7bf-88fa5b0741ff", "ddfadb31-f9f0
             -455c-a9ae-99307fa4905c", "5ab705ec-154c-4785-bbbd-883c1adffbae"
             , "6b5c7807-2038-4752-95dc-5110acc34ea6" "2e815e89-1b69-4bd8-
             ab35-86cf00f835ba"]
      },
      "size" : {
         "status" : "OK",
         "value" : NumberLong(21515),
         "sources" : ["005358b1-70ae-483e-a7bf-88fa5b0741ff"]
      },
      "lastmodified" : {
         "status" : "SINGLE_RESULT",
         "value" : ISODate("1970-01-13T23:14:19Z"),
         "sources" : ["ddfadb31-f9f0-455c-a9ae-99307fa4905c"]
      },
      "checksum_md5" : {
         "status" : "SINGLE_RESULT",
         "value" : "8875f063cb723395b317893290cd7c4c",
         "sources" : ["d286e469-8e86-4bb0-83b4-d9579e33ab87"]
      },
      "lastmodified_fs" : {
         "status" : "INGLE_RESULT",
         "value" : ISODate("1970-01-13T23:14:19Z"),
         "sources" : ["d286e469-8e86-4bb0-83b4-d9579e33ab87"]
      },
      "wellformed" : {
         "status" : "SINGLE_RESULT",
         "valu" : false,
         "sources" : ["005358b1-70ae-483e-a7bf-88fa5b0741ff"]
      },
      "valid" : {
         "status" : "SINGLE_RESULT",
         "value" : false,
         "sources" : ["005358b1-70ae-483e-a7bf-8fa5b0741ff"]
      },
      "message" : {
         "status" : "SINGLE_RESULT",
         "value" : "TokenMgrError: Lexical error at line 166, column 14.
             Encountered: \"\\\"\" (34), after : \"\"",
         "sources" : ["005358b1-70ae-483e-a7bf-88fa5b0741ff"]
      },
      "charset" : {
         "status" : "OK",
         "value" : "iso-8859-1",
         "sources" : ["ddfadb31-f9f0-455c-a9ae-99307fa4905c"]
      },
      "markupbasis" : {
         "status" : "OK",
         "value" : "HTML",
         "sources" : ["005358b1-70ae-483e-a7bf-88fa5b0741ff"]
      },
      "markupBasisVersion" : {
         "status" : "SINGLE_RESULT" "value" : "4.01",
         "sources" : ["6b5c7807-2038-4752-95dc-5110acc34ea6"]
      },
      "created" : {
         "status" : "SINGLE_RESULT",
         "value" : ISODate("2005-07-04T06:23:37Z"),
         "souces" : []
      }
   }
}
```

Listing A.2: The c3po output generated for a single file

A.3 JavaScript Method for fuzzy classification

Listing A.3 displays the JavaScript function that was created for the fuzzy classification of a file. It returns triples of mediatype, format and probability.

```
function getMediaTypeProbabilityIncludingVersion(object) {
   var jsonObj = [];
   var mediatype = "";
   var probability = 0.0;
   var version = "-1";  // -1 = undefined

   // just one mediatype reported
   if(object.metadata['mimetype'].value)
   {
      mediatype = object.metadata['mimetype'].value;
      probability = 1.0;
      if(object.metadata['format_version'])
      {
         if(object.metadata['format_version'].value)     // version
         {
            version = object.metadata['format_version'].value;
         }else if(object.metadata['format_version'].values)
         {
            version = object.metadata['format_version'].values[0];   // possible, because no "real" conflicts
         }
      }
      jsonObj.push({
         "mediatype" : mediatype,
         "probability" : probability,
         "version" : version
      });
   }
   // multiple (different) media types
   else if(object.metadata['mimetype'].values)
   {
      var arrLen = object.metadata['mimetype'].values.length;

      // loop over media types
      for (var i = 0; i < arrLen; i++)
      {
         mediatype = object.metadata['mimetype'].values[i];
         probability = 1 / arrLen;

         if(object.metadata['format_version'])
         {
            if(object.metadata['format_version'].value)     // version
            {
               if((contains(object.metadata['format_version'].sources, object.metadata['mimetype'].sources[i])) != -1)
                  version = object.metadata['format_version'].value;
            }else if(object.metadata['format_version'].values)
            {
               // loop over versions
               version = object.metadata['format_version'].values[0];   // possible, because no "real" conflicts
               var index = contains(object.metadata['format_version'].sources, object.metadata['mimetype'].sources[i]);
               if(index >= 0)
                  version = object.metadata['format_version'].values[index];
            }
         }
         jsonObj.push({
            "mediatype" : mediatype,
            "probability" : probability,
            "version" : version
         });
      }
   }
   return jsonObj;
}
```

```

function contains(a, obj) {
   var i = a.length;
   while (i--) {
      if (a[i] === obj) {
         return i;
      }
   }
   return -1;
}
```

Listing A.3: JavaScript methods that are used for fuzzy classification

A.4 JavaScript Method for Tag Filtering

Listing A.4 displays the JavaScript function for the computation of an array with the sums of valid HTML tags for a specific object. The resulting object array holds all found valid tags, along with their according number of occurrences.

```
function getTagOccurrences(object){
   var validTags = ["a", "abbr", "acronym", "address", "applet", "area", "article", "aside", "audio", "b", "base", "basefont", "bdi", "bdo", "big", "blockquote", "body", "br", "button", "canvas", "caption", "center", "cite", "code", "col", "colgroup", "command", "datalist", "dd", "del", "details", "dfn", "dir", "div", "dl", "dt", "em", "embed", "fieldset", "figcaption", "figure", "font", "footer", "form", "frame", "frameset", "h1", "h2", "h3", "h4", "h5", "h6", "head", "header", "hgroup", "hr", "html", "i", "iframe", "img", "input", "ins", "isindex", "keygen", "kbd", "label", "legend", "li", "link", "map", "mark", "menu", "meta", "meter", "nav", "noframes", "noscript", "object", "ol", "optgroup", "option", "output", "p", "param", "pre", "progress", "q", "rp", "rt", "ruby", "s", "samp", "script", "section", "select", "small", "source", "span", "strike", "strong", "style", "sub", "summary", "sup", "table", "tbody", "td", "textarea", "tfoot", "th", "thead", "time", "title", "tr", "track", "tt", "u", "ul", "var", "video", "wbr"];
   var tags = []; // declare tag array
   for (key in object.metadata) {
      var tagPos = key.lastIndexOf("TagOccurences");
      if (tagPos != -1) {
         if(in_array(key.substr(0, tagPos), validTags))  // check for valid html tag
         {
            tags.push({
               "tag" : key.substr(0, tagPos),
               "count" : parseInt(object.metadata[key].value)
            });
         }
      }
   }
   return (tags);
}
```

Listing A.4: JavaScript method for filtering of the valid tag occurrences

APPENDIX B

Release Years

The following Table B.1 gives an overview of the release years of some common file format versions, that were needed for this thesis. These years can be imported in R to create queries and statistics.

Media Type	**Version**	**Release Year**
application/pdf	1.0	1993
application/pdf	1.1	1996
application/pdf	1.2	1996
application/pdf	1.3	2000
application/pdf	1.4	2001
application/pdf	1.5	2003
application/pdf	1.6	2004
application/pdf	1.7	2006
application/pdf	1a	2005
application/pdf	1a:2003	2003
application/pdf	1b	2005
application/x-shockwave-flash	1	1996
application/x-shockwave-flash	2	1997
application/x-shockwave-flash	3	1998
application/x-shockwave-flash	4	1999
application/x-shockwave-flash	5	2000
application/x-shockwave-flash	6	2002
application/x-shockwave-flash	7	2003
application/xhtml+xml	1.0	2000
application/xhtml+xml	1.1	2001
image/jpeg	2.0	1997
image/jpeg	2.1	1998
image/png	1.0	1996
image/png	1.1	1998
image/png	1.2	1999
text/html	2.0	1995
text/html	3.2	1997
text/html	4.0	1998
text/html	4.01	1999

Table B.1: Release Years of some common Media Type Versions

List of Figures

Listings

Bibliography

[1] Roy A. Allan. *A History of the Personal Computer: The People and the Technology*. Allan Publishing, 2001.

[2] Martin F. Arlitt and Carey L. Williamson. Web server workload characterization: the search for invariants. In *Proceedings of the 1996 ACM SIGMETRICS international conference on Measurement and modeling of computer systems (SIGMETRICS '96)*, pages 126–137, New York, NY, USA, 1996. ACM.

[3] Camera & Imaging Products Association. Exchangeable image file format for digital still cameras: Exif version 2.3, April 2010.

[4] Mark Baker and Peter Stark. Rfc 3236: The 'application/xhtml+xml' media type, January 2002.

[5] Michael K Bergman. The deep web: Surfacing hidden value. *The Journal of Electronic Publishing*, 7(1), August 2001.

[6] T. Berners-Lee, R. Fielding, and H. Frystyk. Hypertext transfer protocol – http/1.0. Technical Report RFC1945, Internet Engineering Task Force, May 1996.

[7] Tim Berners-Lee and Dan Connolly. Rfc 1866: Hypertext markup language - 2.0. Internet RFC 1866, November 1995.

[8] Krishna Bharat and Andrei Broder. A technique for measuring the relative size and overlap of public web search engines. *Computer Networks and ISDN Systems*, 30(1-7):379–388, April 1998.

[9] Jean-Luc Bloechle, Denis Lalanne, and Rolf Ingold. Ocd: An optimized and canonical document format. In *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR2009)*, pages 236–240, Barcelona (Spain), July 2009. IEEE Computer Society.

[10] Adrian Brown. Automatic format identification using pronom and droid. Technical report, The National Archives, March 2006.

[11] Kristina Chodorow and Michael Dirolf. *MongoDB - The Definitive Guide: Powerful and Scalable Data Storage.* O'Reilly, 2010.

[12] Dan Connolly and Larry Masinter. Rfc 2854: The 'text/html' media type. Internet RFC 2854, June 2000.

[13] The World Wide Web Consortium. Recommended list of doctype declarations, 12 2011. `http://www.w3.org/QA/2002/04/valid-dtd-list.html`. Accessed: 2012-02-09.

[14] Microsoft Corporation. Microsoft office word 97-2007 binary file format specification [*.doc], 2007.

[15] Maurice de Kunder. Geschatte grootte van het geindexeerde world wide web. Master's thesis, Universiteit van Tilburg, 2006.

[16] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. In *Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6 (OSDI'04)*, pages 137–150, San Francisco, CA, December 2004. USENIX Association.

[17] Marilyn Deegan and Simon Tanner. *Digital Preservation*. facet publishing, 2006.

[18] Susanne Dobratz. Grundfragen der digitalen Langzeitarchivierung für den edoc-Server. *Cms-Journal*, 32(32):93–98, 6 2009.

[19] Roy Fielding, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, and Tim Berners-Lee. Hypertext transfer protocol – http/1.1, 1999.

[20] The Harvard University Library Office for Information Systems. Introduction to FITS. `http://code.google.com/p/fits/wiki/general`. Accessed: 2012-04-29.

[21] Ned Freed and Nathaniel Borenstein. Rfc 2046: Multipurpose internet mail extensions (MIME) part two: Media types, 1996.

[22] John Gantz and David Reinsel. Extracting value from chaos. *IDC iView*, pages 1–12, June 2011. `http://idcdocserv.com/1142`. Accessed: 2012-05-28.

[23] Frank E. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1–21, 1969.

[24] Antonio Gulli and A. Signorini. The indexable web is more than 11.5 billion pages. In *Special interest tracks and posters of the 14th international conference on World Wide Web (WWW '05)*, pages 902–903, New York, NY, USA, 2005. ACM.

[25] Stephanie Hackett and Bambang Parmanto. A longitudinal evaluation of accessibility: higher education web sites. *Internet Research: Electronic Networking Applications and Policy*, 15(3):281–294, March 2005.

[26] John Erik Halse, Gordon Mohr, Kristinn Sigurdsson, Michael Stack, and Paul Jack. Heritrix developer documentation. `http://crawler.archive.org/articles/developer_manual/index.html`. Accessed: 2012-02-11.

[27] Eric Hamilton. Jpeg file interchange format version 1.02, September 1992.

[28] Margaret Hedstrom, Seamus Ross, Kevin Ashley, Birte Christensen-Dalsgaard, Wendy Duff, Henry Gladney, Claude Huc, Anne r. Kenney, Reagan Moore, and Erich Neuhold. Invest to save: Report and recommendations of the NSF-DELOS working group on digital archiving and preservation. Technical report, Report of the European Union DELOS and US National Science Foundation Workgroup on Digital Preservation and Archiving, 2003.

[29] Ian Hickson and David Hyatt. Html 5. A vocabulary and associated APIs for HTML and XHTML, January 2008. `http://www.w3.org/TR/2008/WD-html5-20080122/`. Accessed: 2012-10-27.

[30] Adobe Systems Incorporated. PDF Reference - Adobe portable document format, version 1.3, July 2000.

[31] Adobe Systems Incorporated. PDF Reference - Adobe Portable Document Format, 6th edition, November 2006.

[32] Adobe Systems Incorporated. Swf file format specification, version 10, November 2008. `http://wwwimages.adobe.com/www.adobe.com/content/dam/Adobe/en/devnet/swf/pdf/swf_file_format_spec_v10.pdf`. Accessed: 2012-11-14.

[33] The Stanford InfoLab. The Stanford webbase project, 2012. `http://diglib.stanford.edu:8091/~testbed/doc2/WebBase/`. Accessed 2012-03-17.

[34] ISO. *Information and documentation - The WARC File Format (ISO 28500:2009)*, 2009.

[35] Andrew Jackson. Analysing the formats in the uk web archive, 8 2012. `http://www.openplanetsfoundation.org/blogs/2012-08-17-analysing-formats-uk-web-archive`. Accessed: 2012-11-16.

[36] Brian Kelly, Kevin Ashle, Marieke Guy, Edward Pinsent, Richard Davies, and Jordan Hatcher. Preservation of web resources: The JISC PoWR Project. *5th International Conference on Preservation of Digital Objects*, August 2008.

[37] Steve Lawrence and C. Lee Giles. Searching the world wide web. *Science*, 280(5360):98–100, 1998.

[38] Steve Lawrence and C. Lee Giles. Accessibility of information on the web. *Nature*, 400:107–109, July 1999.

[39] Shane McCarron and Masayasu Ishikawa. Xhtml 1.1 - module-based xhtml - second edition, 2010. `http://www.w3.org/TR/xhtml11/`. Accessed: 2012-08-14.

[40] Filippo Menczer. *Web Data Mining. Exploring Hyperlinks, Contents and Usage Data*, chapter Web Crawling, pages 273–321. Springer-Verlag Berlin Heidelberg, 2nd edition, 2008.

[41] Gordon Mohr, Michael Stack, Igor Ranitovic, Dan Avery, and Michele Kimpton. An introduction to heritrix. an open source archival quality web crawler. In *Proceedings of the 4th International Web Archiving Workshop (IWAW '04)*, September 2004.

[42] A. Nanavati, A. Chakraborty, D. DeAngelis, H. Godil, and T. D'Silva. An investigation of documents on the world wide web. Technical report, The University of Texas at Austin, December 2004.

[43] Umara Noor, Zahid Rashid, and Azhar Rauf. A survey of automatic deep web classification techniques. *International Journal of Computer Applications*, 19(6):43–50, April 2011. Published by Foundation of Computer Science.

[44] Alexandros Ntoulas, Junghoo Cho, and Christopher Olston. What's new on the web? the evolution of the web from a search engine perspective. In *Proceedings of the 13th International Conference on World Wide Web (WWW'04)*, pages 1–12, New York, NY, USA, May 2004. ACM.

[45] Digital Preservation Department of the UK National Archives. The technical registry Pronom. `http://www.nationalarchives.gov.uk/help/PRONOM/faq.htm`. Accessed: 2012-02-19.

[46] Gautam Pant, Padmini Srinivasan, and Filippo Menczer. Crawling the web. In *Web Dynamics: Adapting to Change in Content, Size, Topology and Use.*, pages 153–178. Springer-Verlag, 2004.

[47] Karl Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine*, 50:157–175, 1900.

[48] Steven Pemberton, Murray Altheim, Daniel Austin, Frank Boumphrey, John Burger, Andrew W. Donoho, Sam Dooley, Klaus Hofrichter, Philipp Hoschka, Masayasu Ishikawa, Warner ten Kate, Peter King, Paula Klante, Shin'ichi Matsui, Shane McCarron, Ann Navarro, Zach Nies, Dave Raggett, Patrick Schmitz, Sebastian Schnitzenbaumer, Peter Stark, Chris Wilson, Ted Wugofski, and Dan Zigmond. Xhtml 1.0: The extensible hypertext markup language, 2000. `http://www.w3.org/TR/xhtml1/`. Accessed: 2012-08-17.

[49] Petar Petrov. C3PO. Clever, Crafty Content Profiling of Objects. `http://ifs.tuwien.ac.at/imp/c3po`. Accessed: 2013-04-23.

[50] Petar Petrov and Christoph Becker. Large-scale content profiling for preservation analysis. In *Proceedings of the 9th International Conference on Preservation of Digital Objects (iPRES 2012)*, Toronto, 2012.

[51] Dave Raggett. HTML 3.2 reference specification. *World Wide Web J.*, 2(1):29–73, 1997.

[52] Dave Raggett, Arnaud Le Hors, and Ian Jacobs. Html 4.01 specification. W3C Recommendation, December 1999. `http://www.w3.org/TR/html401/`. Accessed: 2012-10-29.

[53] Dave Raggett, Arnaud Le Hors, and Ian Jacobs. Html 4.0 specification. W3C Recommendation REC-html40-971218, December 1997. `http://www.w3.org/TR/REC-html40-971218/`. Accessed: 2012-10-29.

[54] Jeff Rothenberg. Ensuring the longevity of digital information. Technical report, RAND Corporation, February 1999.

[55] Simon Butcher. The vnd.microsoft.icon MIME subtype. `http://www.iana.org/assignments/media-types/image/vnd.microsoft.icon`. Accessed: 2012-11-09.

[56] Stephan Strodl, Peter Paul Beran, and Andreas Rauber. Migrating content in WARC files. In *Proceedings of the 9th International Web Archiving Workshop (IWAW 2009)*, Corfu, Greece, October 2009.

[57] Rob Weir. Opendocument format: The standard for office documents. *IEEE Internet Computing*, 13(2):83–87, March 2009.

[58] Allison Woodruff, Paul M. Aoki, Eric Brewer, Paul Gauthier, and Lawrence A. Rowe. An investigation of documents from the world wide web. In *Computer Networks and ISDN Systems*, pages 963–979, 1996.

[59] Francois Yergeau. Rfc 3629: Utf-8, a transformation format of iso 10646. RFC 3629 (Standard), November 2003.