TREBALL FINAL DE MASTER

Títol: Development of a generic test-bed for web scraping

Autor: Pedro Lorente Adamuz

Director: Jesús Alcober Segura

Data: 23 de gener del 2015
<table>
<thead>
<tr>
<th>Títol:</th>
<th>Develop a generic test-bed for web scraping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autor:</td>
<td>Pedro Lorente Adamuz</td>
</tr>
<tr>
<td>Director:</td>
<td>Jesús Alcober Segura</td>
</tr>
<tr>
<td>Data:</td>
<td>23 de gener del 2015</td>
</tr>
</tbody>
</table>

**Resum**

Aquest document conté la descripció i explicació del procés desenvolupat, per la creació d’un “Test-bed” de Web Scraping.

Aquest desenvolupament s’ha dut a terme en el marc d’una empresa anomenada IGLOR Soluciones Audiovisuales Avanzadas S.L. i amb un projecte concret com a referencia Trenditlive.

El document detalla els aspectes bàsics del concepte de web scraping així com de les eines de desenvolupament. Finalment s’explica el disseny i desenvolupament de la eina així com les principals implicacions d’aquesta des del punt de vista tècnic i legal.
Overview

This document contains the description and explanation process developed to construct a Web Scraping Tested.

This development has been guided by IGLOR Soluciones Audovisuales Avanzadas S.L. and takes as a reference the project Trenditlive.

Basic issues of web scraping and the development tools are detailed across the document. Also the design and development of the tool and its consequences are explained from both technical and legal point of view.
Index

INTRODUCTION ........................................................................................................ 1

CHAPTER 1. DEVELOPMENT TOOLS .................................................................... 4

1.1. Web scraping .................................................................................................. 4

1.2. Tools ............................................................................................................ 5

1.3. Capybara Library .......................................................................................... 7

CHAPTER 2. WEBPAGE STRUCTURE ANALYSIS ................................................. 8

2.1. Content structure ......................................................................................... 8

2.1.1. Section ..................................................................................................... 8

2.1.2. Category .................................................................................................. 9

2.1.3. Subcategory .............................................................................................. 10

2.1.4. Basic data structure ............................................................................... 10

2.2.1. Product Information ............................................................................... 11

2.3 Navigation structure .................................................................................... 12

2.3.1. Navigation methods ............................................................................... 12

CHAPTER 3. WEST ARCHITECTURE ................................................................. 14

3.1. How does it work? ....................................................................................... 14

3.2. Components ................................................................................................ 14

3.2.1. Configuration Parser ............................................................................. 15

3.2.2. Parser selector ....................................................................................... 17

3.2.3. Brand structure manager ....................................................................... 18

3.2.4. Product manager .................................................................................... 18

3.2.5. DB Manager ........................................................................................... 19

3.2.6. Normalizer ............................................................................................. 19

3.2.7. CSS Filters ............................................................................................ 19

3.3. Product’s state management ........................................................................ 22

CHAPTER 4. SOFTWARE ARCHITECTURE ..................................................... 25

4.1. Class diagrams ............................................................................................. 25

4.2. Flow diagrams ............................................................................................. 27

4.3. Performance ................................................................................................ 29

CHAPTER 5. LEGAL ISSUES ............................................................................. 31

CHAPTER 6. ENVIRONMENT IMPACT ANALYSIS .......................................... 34

CHAPTER 7. CONCLUSIONS .......................................................................... 36
ANNEX 1. BRANDS STRUCTURES ......................................................... 38

1.1. Brand navigation structures previews ............................................. 38
  1.1.1. Bershka ................................................................. 38
  1.1.2. Desigual ............................................................... 40
  1.1.3. Mango ................................................................. 42
  1.1.4. Massimo Dutti ......................................................... 44
  1.1.5. H&M ................................................................. 45
  1.1.6. Zara ................................................................. 47

ANNEX 2. NORMALIZER TAXONOMY .................................................... 49
List of figures
Fig 1. Trenditlive general structure ................................................................. 1
Fig 2. Scraping structure ................................................................................. 11
Fig 3. Relation between WEST components .................................................. 15
Fig 4 Configuration File example .................................................................. 16
Fig 5. HTML code for Zara colors .................................................................. 17
Fig 6. HTML code for Bershka colors ............................................................. 17
Fig 7. Example 1 css filters ............................................................................ 20
Fig 8. Example 2 css filters ............................................................................ 21
Fig 9. Example 3 css filters ............................................................................ 21
Fig 10. Information related to CSS and HTML tag ........................................... 22
Fig 11. Product Cycle diagram ...................................................................... 23
Fig 12. UML Class Diagram .......................................................................... 25
Fig 13. General Flow Diagram for WEST ...................................................... 27
Fig 14. Flow diagram to product information gather process .......................... 28
Fig 15. Electric Mix for Catalonia/Spain ......................................................... 34
Fig 16. Bershka general structure .................................................................. 38
Fig 17. Bershka product list .......................................................................... 38
Fig 18. Product Information page .................................................................. 39
Fig 19. Desigual general structure ................................................................ 40
Fig 20. Desigual product list ......................................................................... 40
Fig 21. Desigual product information ............................................................... 41
Fig 22. Mango sections and categories ............................................................. 42
Fig 23. Mango product list ............................................................................ 42
Fig 24. Mango product information ............................................................... 43
Fig 25. Massimo Dutti: sections, categories and product list in one page ........ 44
Fig 26. Massimo Dutti product information .................................................... 44
Fig 27. H&M sections and categories .............................................................. 45
Fig 28. HM Product List .............................................................................. 45
Fig 29. HM Product Information ................................................................. 46
Fig 30. Zara sections and categories ............................................................... 47
Fig 31. Zara Product list ................................................................................ 47
Fig 32. Zara Product Information ................................................................. 48

List of tables
Table 1 Features summary for Scrape Libraries .............................................. 6
Table 2. Brands and sections ......................................................................... 9
Table 3. Brands and categories ..................................................................... 9
Table 4. Type of navigation and content structure ......................................... 13
Table 5. Filter options .................................................................................. 22
Table 6. State bits for new product (empty DB) .............................................. 23
Table 7. Operation and bits summary ............................................................. 24
Table 8. Performance table .......................................................................... 30
Table 9. Consumption for development stage (equivalent time to 18 ECTS) ... 34
Table 10. Consumption for production stage (1year) .................................... 35
Introduction

This project has been developed within a Company called “IGLOR Soluciones Audiovisuales Avanzadas S.L”, which aim is to develop multimedia solutions over Internet and it is specialized in working with the cultural and creative sector. It provides customized solutions to experimental and innovative projects related to the networks and telecommunication worlds.

One of the key projects of IGLOR during the last semester of 2014 has been the development of Trenditlive, a test bed to design and develop a Fashion web aggregator. The aim is to group the products provided by the most recognized fashion brands in a unique webpage (www.trenditlive.com). This aggregator is designed to provide a high accurate search engine to the users helping them to reduce the time needed to find desired clothes.

This project develops a solution for one module consisting of a set of aggregators, called scrapping tool. Below the main structure of the aggregator is shown.

![Fig 1. Trenditlive general structure](image)

As seen in the image, the scope of the project is focused on the design and development of a WEB Scraping solution. This module allows the automatization of the process of gathering the information from different Fashion Band webpages.

The main requirements determined by the needs of Trenditlive platform are listed below:

- Collect information from a variety of fashion webpages
- Place the gathered information into a database to allow other modules to use it for compound webpages for analyze it
- Provide an automatic way of processing all the products contained in this webpages, gathering the most relevant information
- Manage the information in order to keep it up-to-date
- Perform the scraping using a common parameters from a configuration file
- Detect duplicated articles in order to avoid duplicated items into the database
- Perform a way of scraping that detects URL problems and avoid them, allowing the tool to continue with the scraping if problems occur
- Filter which products or information has to be gathered for a given webpage
- Normalize types of products using key words from the collected information

The tool developed in this project needs to follow certain rules in order to provide full integration with the whole platform. These rules will constrain the design:

- Work as daemon in the host system
- Provide reports to analyze how the process is going (number of errors, number of duplicated items, time spent…)
- Produce direct output to the database
- Develop a generic module easy to adapt to a wide range of webpages
- Maintain the status of the items in the database ensuring that all items are available in a certain period of time (the same time that are available on its source webpage)

The project report provides a full description of the full process of implementation including from the design to the development of the scraping module of Trenditlive. This module is called WEST as the acronym of WEb Scrapping Tool. The structure of this document is divided in seven chapters.

The first chapter will discuss about the required development tools. At the beginning, Web Scraping concept is introduced in order to provide a first idea of the aim of the project. Once web scraping concept is clarified, different possible alternatives available in the state of the art that can allow the development of a module for web scraping are analyzed and compared in order to select the most suitable one for developing WEST.

The design of WEST has a critical dependency on the structure of the target websites that have to be scraped. Furthermore, the second chapter analyzes the proposed websites provided by Trenditlive platform as a reference for the development. These websites are related to the following fashion brands: Bershka, Desigual, Mango, Massimo Dutti, H&M and Zara.

The third chapter explains the WEST architecture design process. This chapter takes as a reference the analysis performed on Chapter 2 and the requirements explained in the Project Scope. At the beginning of this chapter, it is shown a summary of how WEST will work and the descriptions of all independent components of WEST tools are provided.

The fourth chapter is dedicated for the technical part of the project and provides all the information related to the software design. In this chapter, class diagrams, flow diagrams and some information related to the performance of WEST can be found.
The fifth chapter is focusing on legal issues, which is one of the most polemic points of the project. This chapter provides a global vision on the legality of WEST, taking in consideration both the European Union and National Spanish laws.

The sixth chapter describes the environmental impact related to this project.

Finally the last chapter provides a summary of the main conclusions, explaining the impact of the project into the Trenditlive platform, including the main objectives reached into this project.

Also attached to the project, a bibliography reference and Annex section for additional information, to understand better the information used for design and develop WEST, can be found.
Chapter 1. Development tools

This chapter is focused on the development tools used to develop WEST. The starting point is to provide general information about web scraping and introduce the concept to the reader. Following this description a discussion about the different available developing options based on libraries is performed in order to justify the use of one of these.

1.1. Web scraping

The common definition of Web Scrapping, taken from Wikipedia (Scraping Wikipedia), is:

“Web scraping (web harvesting or web data extraction) is a computer software technique of extracting information from websites. Usually, such software programs simulate human exploration of the World Wide Web by either implementing low-level Hypertext Transfer Protocol (HTTP), or embedding a fully-fledged web browser, such as Internet Explorer or Mozilla Firefox.”

Remark that there are several techniques to perform web scraping that are listed below:

- **Human copy-and-paste.** This is the more traditional technique and it is based on the human behavior and analysis capability. In some cases it is the only method available.
- **Text grepping and regular expression matching.** The use of this technique is based on applying the powerful of grep tool combined with the regular expression management tools available in linux OS in order to extract information from the web.
- **HTTP programming.** Performs the scraping using socket programming and performing HTTP request to both static and dynamic webpages.
- **HTML parsers.** The use of parsers allows to deploy scrape systems to gather information in websites with a large collection of pages generated dynamically from an underlying structured source like a database.
- **DOM parsing.** This system is based on the use of embedded browsers like Mozilla Firefox or Explorer, using them to control and parse the web page DOM tree.
- **Web-scraping software** or **Vertical aggregation platforms.** This kind of software tools are provided by companies and usually works considering stable structures of the content and more static content, they also work with the data managing after the scrape process.
- **Semantic annotation recognizing.** This method is based on the semantic annotations embedded in the web pages, and can be considered it is a special case of DOM parsing.
- **Computer vision web-page analyzers.** It is a new strategy based on the machine learning and computer vision attempting to identify and extract the information as a human.
The strategy type that fits better for this project is **DOM parsing**. In this way, it is important to think the interaction between the web page and users. The web pages on the fashion sector often use Javascript to adapt the content according to the browser or using the actions done by the user in real-time. This results in highly dynamic content environment with the maximum capabilities to simulate the user behavior to obtain the desired information.

### 1.2. Tools

There are two main kinds of tools that can be used to develop WEST:

- Software platforms
- Programing libraries

These two options work in a similar way but for the purpose of this project programing libraries will be used. Software platforms are designed to interact with the user with special needs allowing high end customization. It is common that these platforms combine all the tools needed to gather the information, storage and in most cases performing analysis of the data. Even though WEST can be implemented using this platforms, it is important to notice that WEST will be part of another system and has special needs to adapt its behavior to expected requirements.

Examples of this kind of platforms are Google Chrome Web Scraper extension (Balodis) and Scrapy (Scr) for Perl. The first is maybe the most powerful and useful scrape platform but it is working from a web browser context and is not easy to integrate in another environment like Trenditlive. The other one, Scrappy is also a simple but powerful tool for scapping, although it has some limitations with the type of the content that can manage and work with. Scrappy is not able to manage dynamic content based on AJAX or Javascript (auto generated content). Of course, there are other similar platforms that can be found, but all of them have mainly the same limitations with the scraping content features and/or difficulties to be integrated with another running platforms.

Programing libraries allow to develop a customized piece of software fully integrated with the others modules of Trenditlive platform, they also are able to provide a compliant structure for scraping for the Fashion web pages.

For this reason, WEST is based in libraries. The main features that WEST needs from a programing libraries are listed below:

- Able to manage dynamic content (AJAX, Javascript, HTML 5...)
- Easy to develop and based on a programing language oriented to web development
- CSS/XPATH selectors (DOM navigation)
- Quick integration with common Database servers

The final choice will be on considering the most used and compatible libraries existing for the common web programing languages. The list of libraries that will
be compared base on the Java, PHP, Python, Perl and Ruby is given below in a Table 1. These languages are selected due to their special characteristics for web development. It is important to note, that the company’s developers did not have a previous experience with these languages, therefore the complexity of the given languages will be considered as an important metric for the final choice.

- Scrapy
- Goutte
- Capybara Tool Kit
- Jaunt
- Web::Scraper
- Mechanize

The next table shows a summary of the features provided by these libraries and according with the previous requirement established

<table>
<thead>
<tr>
<th>Library</th>
<th>Dynamic Content</th>
<th>DOM navigation</th>
<th>DB integration</th>
<th>Programing language</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scrapy</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>Python</td>
</tr>
<tr>
<td>Goutte</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>PHP</td>
</tr>
<tr>
<td>Capybara</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>Ruby</td>
</tr>
<tr>
<td>Jaunt</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>Java</td>
</tr>
<tr>
<td>Web::Scraper</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>Perl</td>
</tr>
<tr>
<td>Mechanize</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>Ruby</td>
</tr>
</tbody>
</table>

From the table, you can see that only two options from the six languages match the required criteria of ensuring the compatibility with Dynamic Content (Javascript, AJAX, ...). Also remark that, this compatibility for Dynamic Content scrapping is in all cases reached by the use of an external library called Selenium, that provides a way to create a headless browser able to make the interaction with the webpage. Once Selenium loads the page, then Capybara or Scrapy can start working.

Then if we focus on those that can give us this required functionality, the final choice from these two options bring us to Capybara. There are some reasons to take this decision:

- Components availability. Capybara is compatible with Selenium, and also includes a wide range of drivers that can provide the communication with the website (i.e. Javascript, Phantomjs, Poltergeist,…). However, to use Scrapy, a development of a special code to allow the use of Selenium or another webdriver is required.
- Language orientation. Scrapy is based on Python, which is a multipurpose language that has well known stability, although Capybara is based on Ruby, also a well recognized language, but is specially designed to use it
in web based application and has full featured tools management that will help due to the nature of the actions performed in WEST.

1.3. **Capybara Library**

As explained and decided in the previous section the project development will be based on Capybara library, based at the same time on Ruby. This library provides a full environment that fits perfectly with the aims of this project.

Capybara is initially designed to create web testing environments, in its website the following definition can be found:

> “Capybara is a library written in the Ruby programming language which makes it easy to simulate how a user interacts with your application. Capybara helps you test web applications by simulating how a real user would interact with your app. It is agnostic about the driver running your tests and comes with Rack::Test and Selenium support built in. WebKit is supported through an external gem.”

It is important to take into account this vision, as it allows the design and development of an application that acts as “a user”, performing the same interactions and simulating the action in a web browser instance. Thanks to this way of working, Capybara provides these relevant features:

- Intuitive API, which mimics the language an actual user would use.
- Wide variety of drivers from a fast headless mode to an actual browser.
  - RackTest
  - Selenium
  - Capybara-webkit
  - Poltergeist
- Synchronization thanks to this feature Capybara is able to wait for all the content to be loaded ensuring the execution of all the Javascripts
- High customization of filters based on XPATH of CSS and with several options, which allow controlling the exactness and the strategy to find the elements.

A quantitative and qualitative set of tests has been performed in order to determine the best default driver and take it as a reference in WEST. Analyzing the results, the conclusion is that Selenium is the quickest compatible driver that can be used. Selenium enables to create an instance of a dummy Firefox web browser running into a fake windows systems able to simulate the behavior of a real user.

Due to the readable way to write the filter the first choice for the type of filters (XPATH or CSS) is CSS, as it provides a more accurate way to access to certain elements taking advantage of the hierarchy of the DOM.
Chapter 2. Webpage structure analysis

The analysis of the structure helps to identify the common aspects regarding to the different brands. This analysis is focused on the main characteristics listed below:

1. Web content structure. This aspect is referred on how the information is organized for a given webpage. In this case, all the webpages in the fashion sector present similar ways to organize the product information; this result is an advantage for the tool design.
2. Web navigation structure. This point aims to determine how the webpage plans the navigation of the content and as will be shown later; there are more differences between the webpages in terms of navigation aspects. The navigation structure becomes an important aspect that concede to the webpage a special personality and that is why web designers spend many efforts to produce a significant differentiation from the competitors.

2.1. Content structure

According to the requirements explained in the Project Scope WEST is designed and developed to take 6 brands as a reference and start point. Across this section, the common data structure and the basic information will be shown. The list of analyzed brands include:

- Bershka
- Desigual
- Mango
- Massimo Dutti
- H&M
- Zara

A first look on the websites of these brands gives an idea of the structure followed to organized de data. This structure is based on three levels and gives a hierarchy that allows simplifying the design and development of WEST. These levels are:

- Section
- Category
- Subcategory

2.1.1. Section

The section is related to the gender. The three basic/main genders that are managed in these webpages are Woman, Man and Kids. However, other gender types can be found depending of the analyzed brand, on the table below can be seen the existing sections for each brand:
Table 2. Brands and sections

<table>
<thead>
<tr>
<th>Brand</th>
<th>Sections</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bershka</td>
<td>New, Bershka, Chico, BSK, Accesorios, Zapatos, Campaña, Blog</td>
</tr>
<tr>
<td>Desigual</td>
<td>Mujer, Hombre, Accesorios, Zapatos, Kids, Decoración, Outlet, Catalogo, Tiendas</td>
</tr>
<tr>
<td>Mango</td>
<td>Mujer, Hombre, Violeta</td>
</tr>
<tr>
<td>H&amp;M</td>
<td>Mujer, Hombre, Niños, Home, H&amp;M Life</td>
</tr>
<tr>
<td>Zara</td>
<td>Mujer, TRF, Hombre, Niño, Mini, Denim, Zapatos</td>
</tr>
</tbody>
</table>

This table shows a wide variety of different sections from each brand. Obviously, not all of these sections have to be processed by WEST. Furthermore, WEST must establish a protocol to the valid sections for a given brand. An example of this fact can be observed in Massimo Dutti where only the sections Women, Men, Boys and Girls would be processed by WEST, because the other sections are not important, as they do not contain special products to process.

2.1.2. Category

This level refers to the type of clothes for a given section. At this point is interesting to observe that each brand has different ways to designate the types of clothes. An example with the section “MUJER” can be seen on the table below.

Table 3. Brands and categories

<table>
<thead>
<tr>
<th>Brand</th>
<th>Category (Section “MUJER”)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bershka</td>
<td>New, Mid Season Sale, Básicos, Abrigos Y Cazadoras, Vestidos Y Monos, Camisetas, Camisas &amp; Blusas, Tops &amp; Bodies, Jerséis &amp; Chaquetas, Sudaderas, Jeans, Pantalones, Faldas, Shorts, Dressy Night Collection, Sport Start Moving</td>
</tr>
<tr>
<td>Desigual</td>
<td>Vestidos, Camisetas, Abrigos &amp; Chaquetas, Jerséis, Faldas, Pantalones &amp; Jeans, Camisas &amp; Blusas, Bolsos &amp; Accesorios, Zapatos, Bikinis, Pijamas, Perfumes, Sport, Fundas</td>
</tr>
<tr>
<td>Mango</td>
<td>Vestidos, Abrigos, Chaquetas, Cárdigans y jerséis, Blusas y camisetas, Camisetas y tops, Pantalones, Jeans, Faldas, Shorts, Monos, Lencería, Deporte, Moda baño</td>
</tr>
</tbody>
</table>
In general, all the brands have quite similar categories. As occurs with sections, a special protocol to select the valid categories has to be provided, to avoid those that doesn’t have products to process. An example of this could be the case of “New”. The products inside the category “New” usually are included into their own category and can produce some duplicate product errors. For example, the category of “Abrigos” include all of this kind of products and the new ones that are included inside the “New” category.

2.1.3. Subcategory

Subcategory is a special level that it is only used in a few cases. This level provides more details to the user in order to filter the contents. For the aim of WEST, this level is out of the scope. Although being analyzed, WEST will not consider it as a new category, since the section and category of a product is identified with enough detail to integrate it on the data structure of Trenditlive.

So, from the point of view of the treatment given to Subcategory is the same that with category providing also a method to select the valid categories.

2.1.4. Basic data structure

ANNEX I contains all the previews from webpages of the proposed brands and presents the general structure used for each one to organize the information (Section, Category and Subcategory). In addition, a common hierarchy is used, as it can be seen.

A first conclusion is that all the webpages are using Sections, Categories and a product list, and only in a few cases Subcategories. Using this as a base, WEST will go over all wanted sections and try to assign categories for each section making the same procedure for subcategories when exist. Diagram below shows the tree structure that are used to perform the scraping over each webpage.
2.2.1 Product Information

The product information is the most important point to define, as it will be the key element (unit) when integrating WEST into Trenditlive, it needs to use coherent description of every product. The relevance on the definition of this information and its structure is determined by the functions of the search engine of Trenditlive. This information has to allow:

- Database model coherent definition
- Linked information (photos, colors, and other attributes)
- Type of product automatic detection
- Normalized categories or product type
- Provide enough information to perform complex search sentences over the database
- Provide information to manage the status of the products:
  - Product on sale
  - Old product
  - New product
  - Duplicated product
  - Wrong product (it can be caused by several different factors: bad information, it does not exist, problems with photos…)
The main problem related to the product information gathering process is the differences between the available product information for each brand. The relevant factor is that the information provided for the product of each brand is a little bit different. In this way nobody follows a standard, producers are promoting and highlighting information that represents an added value for the image of the brand. Therefore, a list of minimum fields related to product description has been defined in WEST

- Product Name
- Product Reference
- Brand
- Department
- Season
- Description
- Product URL
- Photo/s URL
- Price
- Old price
- Sizes
- Type
- Section
- Category
- Subcategory

All these fields are fetched directly from the webpage of the product, although there are some fields that are post processed in order to obtain homogeneous information between different brands. Type, section, category and subcategory are grouped and analyzed to perform a standard taxonomy. This allows WEST to assign shared tags to each clothes. For the case that some brands that do not include information for some fields, WEST just assigns to the field a default value “-“.

2.3 **Navigation structure**

This section makes the analysis of the navigation method followed by the different brands in their websites. One of the most important things of WEST is the capability for navigate across the entire websites. Thanks to the knowledge of the content structure, WEST is able to follow the links in order to perform a systematic walk across the whole website.

2.3.1 **Navigation methods**

As mentioned in the last point, all the webpages have the same way to organize the information but they use different strategies to present the product’s list to the final users. All of these strategies have to be considered in order to provide the right resources to WEST to follow these lists. These strategies are called:
– **Page based (PB).** In this case, a user can navigate through the product list using a page structure, where can be found a certain number of products per page.

– **Infinite scroll (IS).** In this case, the user can see the products when he/she scrolls down the webpage. This scroll is detected through JavaScript mechanism and load a certain number of blocks. This method can be interpreted as a per block load system, where a block is a number of products. Sometimes Infinite scroll is implemented through a page based scheme hidden to the user and managed using JavaScript.

– **All products (AP).** In this case the webpage prompts all the products to be accessed when the user scroll down the page. The main difference between infinite scrolling is that all the products are loaded from the beginning.

The table below shows how each brand presents the information and how much detailed is the structure.

**Table 4. Type of navigation and content structure**

<table>
<thead>
<tr>
<th></th>
<th>IS</th>
<th>PB</th>
<th>AP</th>
<th>Sections</th>
<th>Category</th>
<th>Subcategory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bershka</td>
<td>-</td>
<td>YES</td>
<td>-</td>
<td>YES</td>
<td>YES</td>
<td>-</td>
</tr>
<tr>
<td>Desigual</td>
<td>-</td>
<td>-</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>-</td>
</tr>
<tr>
<td>Mango</td>
<td>-</td>
<td>YES</td>
<td>-</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Massimo Dutti</td>
<td>-</td>
<td>-</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>-</td>
</tr>
<tr>
<td>H&amp;M</td>
<td>-</td>
<td>YES</td>
<td>-</td>
<td>YES</td>
<td>YES</td>
<td>-</td>
</tr>
<tr>
<td>Zara</td>
<td>YES</td>
<td>-</td>
<td>-</td>
<td>YES</td>
<td>YES</td>
<td>-</td>
</tr>
</tbody>
</table>

WEST implements these three strategies for gathering every product and is able to go over the right data structure depending on the sections, categories and subcategories.
Chapter 3. WEST architecture

This chapter describes WEST’s architecture. At first, a few lines explaining the how WEST is working are attached. Then all the WEST components will be described.

3.1. How does it work?

As explained in Chapter 1, the development is based on Ruby and several libraries that involve the web testing tools. The developing idea is a tool that acts like a user navigating across the whole target webpage collecting the expected data.

WEST takes advantage of the powerful Capybara + Selenium to go over all the webpage and collect the product information. Below, the main steps to scrape a web (assuming a webpage without subcategories) can be followed:

1. Visit the main URL
2. Filter using the corresponding CSS code for sections and store the section’s list
3. Visit the first section and filter by category CSS code and store the categories list
4. Visit the first category and filter by the product CSS and store the product’s list
5. Iterate over the list of product applying the different CSS codes for each field of information defined
6. Take the full list of products with all the information and insert them to the DB
7. Go to point 3 and start the next section if such exists. This point is repeated until no section is available to process.

Related to this process, and as can be interpreted the sequence, the sequence is quite simple and goes over the different elements using recursive algorithm. The sequence just finishes when no more sections (with their related categories and subcategories) are available.

The main difficulty in WEST is the filter definition and its customization to the different brands. As can be seen on the ANNEX 1, all the webpage structures are quite similar but they follow different HTML code and also are using Javascript adding complexity to detect common patterns between them. Defined filters have to be strongly accurate in order to fetch desired information.

3.2. Components

Several functional parts dividing the main functions and functionalities compose WEST. The idea is to isolate the main functions in order to obtain a comprehensive and easy software architecture that can be “easily” modified and adapted according to the needs of the different brands.
The main components of WEST are:

- **Configuration parser.** Its main function is to read the configuration file and set all the custom variables to allow the creation of the right parser version for each brand.
- **Parser Selector.** Its main function is to create the instance of each brand specific parser.
- **Brand structure manager.** This component maps the first level of navigation to parse a webpage, it means the sections, categories and subcategories.
- **Product Manager.** This component lists and gathers information of all the products via applying the navigation method defined in the configuration file.
- **Database Manager.** It is the interface between WEST and the Database, its function is to push the collected data from Product Data Manager in to the Database in a coherent way.
- **Normalizer.** This component is making the information post processing in order to carry out the required standardization of the several fields from the collected data for a given product.

On the next diagram, the relation between these components can be seen.

![Diagram showing the relation between WEST components](image)

3.2.1. **Configuration Parser**

The configuration files are defined using YAML (Evans, 2001). Ruby has a specific library to manage YAML files and it is easy and comprehensive to define and to implement. Below an example of a standard configuration file:
It is important to note that through this file all the needed information is provided to WEST, especially all the patterns used to filter the webpage content. All the fields containing the keyword “pattern” are related to the CSS codes to gather specific information related with the structure, navigation mode or product information.

Main parameters are:

- **Brand** is used to know the needed instance of the parser. It is used on Parser Selector
- **URL** is related to the shop online of the brand, frequently there are brands where the homepage is different from the shop online webpage.
- **Capybara_driver** allows to select what kind of driver has to use the parser. In WEST all the parser are using selenium due to its compatibility with Javascript and its stability.
- **Log_file** turns on/off the log subsystem
- **Insert_to_db** allows to run the parser without modifying the database for debugging purposes
- **Pagetype** selects the navigation mode to follow the structure of the webpage

### 3.2.2. Parser selector

Once the previous component read configuration file Parser Selector, it uses this information to create the proper instance of a parser in order to obtain a customized way to get the information fields. This is important, due to the existing differences between brands. The product information gathering functions have to be implemented using different strategies. There are one parser class for every brand allowing customization of the needed fields.

This product information differences make difficult to work with universal CSS filters, as an example a color collection from Zara and Bershka can be seen below.

---

**Fig 5. HTML code for Zara colors**

```html
<!div id="colors-296286" class="colors">
  <div class="img" title="Negro" data-colorcode="800">
    <img src="https://static.zara.net/photos/1915/11/22/p/5977/360/800/0/1/5/1/5.jpg?timestamp=141184524105" class="checked">
  </div>
</div>
```

**Fig 6. HTML code for Bershka colors**

```html
<!div class="colors_detail">"Negro id="shortColor0">
  <img src="http://static.bershka.net/photos/2014/11/22/p/7453/160/800/0/1/5/1/5.jpg?timestamp=141184524105" alt="Negro">
  <div id="color_detail" target="colorDetail" class="productDetail" data-parent="colorDetail">
    <div class="productColor">
      <div class="colorName">Negro</div>
      <div class="colorValue">Negro</div>
    </div>
  </div>
</div>
```

Zara website uses a CSS that is different to Berhska’s website. Zara colors use this CSS filter “#colors .colorEl” rather than Bershka is “.color_detail”. As can be observed, the difference between these two examples is not only about the CSS, two brands are using the attribute title to put the name of the color but Zara’s
Develop a generic test-bed for web scraping

website puts it inside of a div called “imgCont” and Zara’s inside a div “color_detail”.

For this reason, a customized parser has to be implemented for each brand and related to specific fields of the product information.

3.2.3. Brand structure manager

The main function of this component is to prepare and adjust the navigation structure needed for each brand. The main reason to add this component to WEST is specific customization needs for each brand in terms of sections, categories or subcategories and for issues related with the initial homepage.

Frequently brands that not allow to access directly to its shop page or maybe it has not properly structured the products following the defined hierarchy can be found. Brand structure manager allows defining the proper way to customize and map all of these issues to be sure that a list of products can be fetched.

In addition, as said before, this component also needs to allow the customized process to enter to the website. Nowadays is common to find websites showing at first point a form to fill the country and the language or to get another information from the user. In any case, thanks to this component customization, this process can be performed according to the special requirements of each brand.

3.2.4. Product manager

This is the main component of WEST and it has several functions. The result of this component is a full list of the products with all of information properly filled in. It is connected to DB Manager and Normalizer in order to produce coherent results and ready to use for other parts of the platform.

The main functions of a product manager are:

- List the products and get its URLs
- Gather all the information related to a product
- Create statistics of the process

The first task is to create the list of the products, performed according to the navigation mode provided on the configuration file. As explained before these modes are Infinite scroll, Page based and All products. Following these modes, the component constructs a list of the products obtaining each URL. The next step is the iteration on this list gathering the production information.

All these functions are controlled in order to provide a minimum statistics that allow to know:

- Number of successfully processed products
- Number of wrong processed products
Number of duplicated products that could not be inserted into the database

During the development phase a protection control from the webpages was detected which was trying to prevent flooding attacks or scrapings. This protection control caused deny of request from the target webpage. Product Manager implements a strategy based on 120 seconds waiting in case product information page could not be fetched. Thanks to this mechanism, in most cases the protection controls can be avoided.

3.2.5. DB Manager

The function of DB Manager is to insert the products in the database. This operation involves few checks to find duplicated products. In addition, this component is in charge of the state management of each product.

Product duplication is detected using a hash code based on three fields of the product information:

- Name
- Product URL
- Default photo URL

3.2.6. Normalizer

Normalizer takes care about how the brands are classifying the different types of clothes. As explained before, the final result of WEST is used as the content for Trenditlive platform, and a common and specific classification has to be defined in order to group the clothes through unique keywords.

The final aim of this component is to use a basic taxonomy that allows the classification of the product. That can be used to tag a product regardless of its brand. The full development of this Normalizer and its taxonomy is out of the project’s scope but to provide a first approach of the component a rudimentary model of taxonomy has been developed. This model contains one level for product classification based on a hash table (more information can be found on ANNEX 2).

In addition, this component performs another standardization operation with the objective to obtain a common format from all the products regardless its brand. These two operations are down case all the fields and format price to “0.00”

3.2.7. CSS Filters

Filters are an important part of this project and this section will explain it in more detail in or to provide a wide view to the user.
These filters help WEST to extract and navigate through a given webpage. On the navigation side, WEST is using the filters to obtain:

- List of sections
- List of categories
- List of subcategories
- List of products
- Elements involved in the navigation method (infinite scroll, web page, all products)

These filters are also used to get the specific information for a product, each field of information has a filter associated in order to provide an accurate way to fetch the information.

Sections, categories, subcategories and products are stored using a data structure composed by a text field with name and another text field containing the URL for each element. Navigation Method elements use a filter way that allow WEST to move through the pages using “Next page” or switch the mode to “Show all products” or even perform the right scroll to obtain a list of all the products.

The use of CSS filters are limited to the visible HTML/CSS tags. Javascript is present on the major part of the webpages, these results in the use of hidden elements that are coded in HTML page using the visibility attribute. Thanks to the capabilities of Capybara, these hidden elements can be filtered to get the needed parameter to navigate the webpage properly. An example of the kind of filters used for a webpage is given below:

**Sections CSS Filter:** `div#menu li a`

**Fig 7. Example 1 css filters**

**Categories CSS Filter:** `div#menu ul li ul li a`
Products CSS filter: `div#products a.product-name`

CSS filters to get product information:
Develop a generic test-bed for web scraping

- Reference: `p#product-ref`
- Price: `.prices`
- Description: `h2#product-long-desc`
- Color: `div#product-colors ul li img`
- Size: `.product-sizes ul li`

![Fig 10. Information related to CSS and HTML tag](image)

As can be seen, the filters are constructed using `div` and other tags of the HTML code also integrating the class and the id of the tags. Briefing the next table shows how a filter can be constructed.

<table>
<thead>
<tr>
<th>Filter</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using generic html tag</td>
<td><code>div,h2,ul,...</code></td>
</tr>
<tr>
<td>Using a div with special id</td>
<td><code>div#product-colors</code></td>
</tr>
<tr>
<td>Using a div or another HTML tag with special class</td>
<td><code>.product-sizes</code></td>
</tr>
<tr>
<td>Combine div id with special class</td>
<td><code>div#product-colors .active</code></td>
</tr>
</tbody>
</table>

3.3. Product’s state management

The product’s state management has 4 main functions and is in charge of the maintenance of products in the database. Each time WEST is executed, it obtains a list of the products of the websites and has to perform operations regarding existing products in the database. These operations are:

- Add new products
- Modify information of existing products. This information could be price, available sizes, colors and alike
- Mark as discontinued those products that are no longer in the scraped website but still exist in the database.
The product cycle according to this operation is drawn in the following next diagram

![Diagram showing the product cycle](image)

Fig 11. Product Cycle diagram

The proposed strategy to cover these product states is using 4 bits to manage different situations. These bits are:

- Published (pub)
- New (new)
- Discontinued (dis)
- Notify (not)

Regarding to the product cycle diagram each product state has proper combination of these bit that can be seen on the next table:

<table>
<thead>
<tr>
<th></th>
<th>pub</th>
<th>new</th>
<th>dis</th>
<th>not</th>
</tr>
</thead>
<tbody>
<tr>
<td>New</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Modified</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Up-to-date</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Discontinued</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
There are two different initial situations according to the database. The first is when DB has not items related to the given brand. The second one considers that already exist items of the brand into the DB.

**No items into DB**

When WEST is executed for the first time for a given brand all the products/items added to the DB will inserted with the combination corresponding to New. In this state, a product can be published and displayed on Trenditlive frontend, and it can be highlighted as new content.

**Items into DB**

When WEST is executed and there are products in the database related to the parsed brand, the next situation can be found:

- If the product does not exist then WEST marks it as New
- If the product exists and there are no changes related to its information, then WEST marks it as Up-to-date
- If the product exist and there changes in the information, an update will be needed for this product, then WEST marks it as Modified.

As a summary of the operations below, a table with the actions the bit corresponding bit combination is shown

<table>
<thead>
<tr>
<th>Actions</th>
<th>pub</th>
<th>new</th>
<th>dis</th>
<th>not</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEST execution without items into DB</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>WEST execution with items into DB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>___Before parsing</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>___During the parsing. Item exists and no changes needed</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>___During the parsing. Item exists and need to be changed</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

When a product is already in the DB but is not parsed it means that this product is discontinued. The detection of this situation is performed setting the bits on “Before parsing”. When WEST finishes all the products that has this, bits will be put in the discontinued state to prevent it to be displayed by Trenditlive.
Chapter 4. Software architecture

4.1. Class diagrams

Fig 12. UML Class Diagram
The previous image shows the class diagram used to develop WEST. It is based on 5 different classes with different relations between them. The main class is Parser in charge of the scraping tasks. This main class uses the others to adapt the program structure for gathering the needed information of the webpage. The class regarding to Database management is not included in this diagram because it is provided by the Trenditlive in order to avoid compatibility problems coming from the design of WEST.

Below can be seen the detailed description of these classes and their role in the global structure.

- **Parser**. It is the general structure containing the navigation functions and the map of the structure of the webpage. Also is in charge of the Database communication and logging functions of WEST.
- **ParserBrand**. This class is an inheritance of Parser and allows developing special functions to get information from a webpage. As said before, in general terms the target webpages are quite similar but all of the brands use different ways to produce the same result, this point does not allow to use the same code to get the same information (for example the color) for all the brands. ParserBrand allows creating special parser for each brand with adapted code to get several information.
- **Normalize**. It is the class in charge to produce a homogenous format for any product regardless of the brand. This class creates a common section and category structure used by the other modules of the system. For example in case of pricing, it converts all the prices to the same format 00.00 (it is common to find different notations for prices i.e. 19,99)
- **ProductInfo**. It is a class that maps all the items needed to describe a product.
- **BlockInfo**. It is the minimum block of information that contains the Name of a Product, a ProductInfo structure and the URL where it can be founded.
- **BlockArray**. This class is a special case of an Array based on BlockInfo. The main reason to define this class is to allow several functions to work with the BlockInfo.
4.2. *Flow diagrams*

This block explains the tasks and functions that WEST performs to scrape the information from a target webpage. The beginning of this is to present general flow diagram with the process performed for each brand. This process is shown in the figure below.

![Flow Diagram](image)

**Fig 13. General Flow Diagram for WEST**

As can be observed WEST works with recursive functions, processing the information through the sections, categories, subcategories and finally across the products. When a product list belongs to a subcategory or category, it begins the process to gather the information for each product. Once the entire list is processed, it goes to the next category or if category is the last in the list - it goes to next section. WEST follows this process until all of the subcategories, categories and sections are processed.
The main part of the process is performed into “Gather Product’s Inf”. At this point all the information defined for a product is collected, normalized and finally stored in the database. This process is described with another flow diagram that can be seen below.

Fig 14. Flow diagram to product information gather process

Gathering of product information is done using a loop over the list of products as can be seen in the flow diagram. This loop extract all the information related to a
product (title, price, description...) until no more products are available. It is important to note that sometimes there are some errors when a product URL is visited. These errors could occur due to three main reasons:

- The webpage detects the activity of WEST as an unusual source of traffic and disable the access temporally from the attached IP to the WEST’s machine.
- There is a problem with the webpage and the target product is listed but its URL is not available due to a management problem of the brand maintenance. This error usually redirects to a page with this message “This product doesn’t exist” or similar.
- There is a problem loading the URL of the product due to the time needed to load images and other html elements breaking timeouts of Capybara.

Problems are listed in order of appearance. The most important issue is the first, when the webpage identify the WEST’s activity as a Deny of Service attack. Each time this issue appears WEST enters in a loop waiting 200 seconds before trying again to visit the URL. This time lapse is not random, it has been chosen because usually the firewall that protects the webpages use a block time over 120 seconds. The iteration loop works until the page is reached or until WEST has tried to visit 5 times. This retries ensure that WEST can increase the likelihood of avoid the block performed by the website’s firewall.

The other two problems are hard to detect and distinguish because they appear in the phase of “Product Process Information”, for this reason these two problem are treated as the same. Several running tests done with WEST points that likelihood of occurs this problem is really low than appears the first one. The mechanism to avoid or reduce the impact of these two problems is a loop of 3 iterations with a sleep of 3 seconds. Thanks to this strategy, WEST waits a reasonable time to allow all elements to be loaded.

All this strategies result in the same behavior when the problem could not be avoided, WEST skips a current product and tries to continue with the next one. Only if WEST wait for 120 seconds 5 times the execution is interrupted with an error code.

**4.3. Performance**

This project is based on gathering of the information from different websites, the performance involves several issues that are hard to evaluate. As an exercise this section provides a list of those variables affecting the final performance, but a very detailed analysis of its impact are not covered by the project scope.

Main variables affecting to the performance:

- Network congestion
  - Web scraping server
  - Target website
- Size of the images
- Type of navigation mode
- Number of categories/subcategories
- Number of concurrent parser running at the same time
- Product contents (number of colors, number of photos,…)

In general terms, the most relevant variable is the size of images, always assuming good quality on the network. In addition, it is important to take into consideration that websites can allow a maximum speed per user in order to protect their platform from collapse. The table below shows the results of measuring the time spent to perform a scrape of the target websites.

**Table 8. Performance table**

<table>
<thead>
<tr>
<th>Items</th>
<th>Errors</th>
<th>Total Time (h)</th>
<th>Time per item (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bershka</td>
<td>3360</td>
<td>2,96</td>
<td>3,2</td>
</tr>
<tr>
<td>Desigual</td>
<td>1126</td>
<td>1,12</td>
<td>3,7</td>
</tr>
<tr>
<td>Mango</td>
<td>4600</td>
<td>3,88</td>
<td>3,0</td>
</tr>
<tr>
<td>Massimo Dutti</td>
<td>1185</td>
<td>0.88</td>
<td>2,7</td>
</tr>
<tr>
<td>H&amp;M</td>
<td>3433</td>
<td>2,14</td>
<td>2,2</td>
</tr>
<tr>
<td>Zara</td>
<td>5746</td>
<td>5,3</td>
<td>3,3</td>
</tr>
</tbody>
</table>

This results show an average of over 3 seconds per product that is the reference time for each product. Another important aspect of this table is the difference between the numbers of products processed for each brand. As can be seen WEST process 5746 from Zara vs 1126 of Desigual. The main reason for this difference is due to WEST performs de scrape without previous information of each processed product. WEST unknowns if a product was processed or not the only information about a product is done when a product is inserted in the DB, but at this point it is already processed.
Chapter 5. Legal Issues

Web scraping or screen scraping is an activity used by many websites in order to provide aggregate results to end users. These websites are called web aggregator for example: Atrapalo, Lyst, Kelkoo, etc.

Legal aspects of web scraping depend on the treatment of the information and how this activity concerns the business of the scraped webpages. From the information treatment, the behavior of web scraper can be divided in two types:

- Scraped information is stored in a Database
- Scraped information is created on demand

WEST is in the first group because all information is stored in a database. This type is most sensitive case as targeted websites can see this activity as illegal copying of their contents.

Before we start to evaluate the legal aspects of WEST, it is important to understand how Trenditlive will use the collected information. At this point, the data gathered from the different brands is stored in a database, which is used to show as results of the searches performed by the users. When a user clicks on a product it is automatically redirected to the original webpage, it means that Trenditlive is not involved into the sell process.

Affiliation programs
There are websites allowing the use of their products for third party platforms. This kind of activity is developed under affiliation programs. Thanks to affiliation programs, third party platform can redirect the traffic to final website offering a product list and taking a fee if the user finally performs a purchase. That program is offered by some brands like Nike, Custo, Springfield etc. This program works using code associated with the redirected URL that is used to know from where the user arrives to their webpage, when the user buy the product then a fee is generated to the source of the content.

Trenditlive includes brands with and without affiliation programs. In terms of laws, the brands providing affiliation programs are implicit allowing the use of their data for a third party platform. The problems related to the laws are focused on the scrapped brands without an explicitly authorization.

Positioning of Trenditlive
This part aims to clarify the position of Trenditlive regarding to the scraped websites. This position would help Trenditlive to appear as an advantage from the point of view of these websites.

Trenditlive intends to be considered as a new source of users for the brands and does not want to appear as a competitor. Trenditlive main value is to offer an easy and accurate way to find the desired items for the users, without the need of spend a lot of time browsing across different websites. As is explained before, once the user click on product it is automatically redirected to the brand website.
Legality

Previous information describes which kind of Database access performs Trenditlive and the position that it would take regarding the scraped websites.

The initial discussion about the legality starts with this question: What is the law that have to be applied according to Trenditlive activity? The answer is European Directive 96/9/EC “Legal protection of Databases”, this law is covering the uses that can be performed regarding to databases. Spanish law that makes reference to this directive is “Ley 5/1998 del 6 de marzo – Protección Jurídica de las Bases de Datos” and adds directly the content of the directive. Due to this, the main references are made to the European Directive instead of the Spanish law.

Nowadays Spanish legislation is subject to a deep transformation, due to the recently approval of “Ley 21/2014 – Ley de la Propiedad Intelectual” this law is specially focused on the authors and the rights related to their works and the regulation of the databases is made through the law “Ley 5/1998”. Even this is important to take in consideration the latest shut down of the google news service in Spain thanks to this law. Focusing on the project scope this law is taking effect due to this main point is to protect the re-distribution of the contents related to news contents and as can be imagined Trenditlive is out of this scope.

European Directive 96/9/EC Legal protection of Databases

The Directive 96/9/EC object is the legal protection of databases. Trenditlive’s activity is covered specially in the article 7 related to the “SUI GENERIS RIGHT”, that is covering maker of database right and protection issues, at the beginning can be read this paragraph:

“1. Member States shall provide for a right for the maker of a database which shows that there has been qualitatively and/or quantitatively a substantial investment in either the obtaining, verification or presentation of the contents to prevent extraction and/or re-utilization of the whole or of a substantial part, evaluated qualitatively and/or quantitatively, of the contents of that database.”

It is important to link to the explanation of the terms used in this laws concerning to extraction and re-utilization:

“(a) ‘extraction’ shall mean the permanent or temporary transfer of all or a substantial part of the contents of a database to another medium by any means or in any form;
(b) ‘re-utilization’ shall mean any form of making available to the public all or a substantial part of the contents of a database by the distribution of copies, by renting, by on-line or other forms of transmission. The first sale of a copy of a database within the Community by the rightholder or with his consent shall exhaust the right to control resale of that copy within the Community;”

Related to this right the obligations are explained on article 8:
“1. The maker of a database which is made available to the public in whatever manner may not prevent a lawful user of the database from extracting and/or re-utilizing insubstantial parts of its contents, evaluated qualitatively and/or quantitatively, for any purposes whatsoever. Where the lawful user is authorized to extract and/or re-utilize only part of the database, this paragraph shall apply only to that part.

2. A lawful user of a database which is made available to the public in whatever manner may not perform acts which conflict with normal exploitation of the database or unreasonably prejudice the legitimate interests of the maker of the database.

3. A lawful user of a database which is made available to the public in any manner may not cause prejudice to the holder of a copyright or related right in respect of the works or subject matter contained in the database.”

Also exceptions of these rights are provide in Article 9:

“(a) in the case of extraction for private purposes of the contents of a non-electronic database;
(b) in the case of extraction for the purposes of illustration for teaching or scientific research, as long as the source is indicated and to the extent justified by the non-commercial purpose to be achieved;
(c) in the case of extraction and/or re-utilization for the purposes of public security or an administrative or judicial procedure.”

The conclusions extracted from the analysis of previous information:

- Trenditlive activity is not included in the exceptions provided by Article 9
- Trenditlive activity is not conflicting with the main business of the database’s owners
- Trenditlive makes a copy of the database and for this performs an illegal use of the databases
- Trenditlive has to obtain a permission of use of this databases in order to perform a legal use of the target databases
- Trenditlive affiliation program activity is made under legal condition has not effects in terms of legacy

The legacy of the operation of Trenditlive was not analyzed beforehand and these results are so important in order to design a good strategy that allows Trenditlive to operate in a lawful way. In addition, it is important to consider that in terms of impact Trenditlive will not have problems in the initial phase due to little number of users that will manage. As Trenditlive starts growing the legal aspects of the use of the database will take more importance and have to be treated with a right foresight considering all the aspect mentioned in this document.
Chapter 6. Environment Impact Analysis

The environment impact of the software design and development is difficult to analyze. The impact of this project will be analyzed following these two considerations:

1. Power consumption during the designing and development in terms of:
   - Computer use
2. Power consumption related to the production period of the developed module

The environment impact will be calculated in form of footprint carbon. The calculation of this footprint is performed following the conversion rules related to the Spain electricity production system that is the platform providing electricity to Catalonia’s network. The table below is obtained from (Oficina Catalana del Canvi Climàtic, 2012)

<table>
<thead>
<tr>
<th>MIX ELÉCTRICO (g CO2/kWh)</th>
<th>2008</th>
<th>2009</th>
<th>2010</th>
<th>2011</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>313</td>
<td>297</td>
<td>206</td>
<td>267</td>
</tr>
</tbody>
</table>

Fig 15. Electric Mix for Catalonia/Spain

For the purpose of this project a value of 248 g CO2/KWh is taken from the corresponding document regarding to the year 2014 (Climàtic, 2014).

Table 9. Consumption for development stage (equivalent time to 18 ECTS)

<table>
<thead>
<tr>
<th>Development Stage</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PC power consumption</td>
<td>300 W</td>
</tr>
<tr>
<td>PFC ECTS</td>
<td>18 ECTS</td>
</tr>
<tr>
<td>PFC Hours (ECTS -&gt; 30hours)</td>
<td>540 hours</td>
</tr>
<tr>
<td>PFC correction factor (20%)</td>
<td>648 hours</td>
</tr>
<tr>
<td>Total Consumption</td>
<td>194,4 kWh</td>
</tr>
<tr>
<td></td>
<td>0,017 tep</td>
</tr>
<tr>
<td></td>
<td>48,21 kg CO2</td>
</tr>
</tbody>
</table>
Environment Impact Analysis

Table 10. Consumption for production stage (1 year)

<table>
<thead>
<tr>
<th>Production Stage</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Server power consumption</td>
<td>600 W</td>
</tr>
<tr>
<td>Weeks per year</td>
<td>52</td>
</tr>
<tr>
<td>Runs 2 days per week</td>
<td>104</td>
</tr>
<tr>
<td>1 Run -&gt; 12h</td>
<td>1248</td>
</tr>
<tr>
<td><strong>Total consumption</strong></td>
<td><strong>748.8</strong> KWh</td>
</tr>
<tr>
<td><strong>Total CO2 emission</strong></td>
<td><strong>185.70</strong> kg CO2</td>
</tr>
</tbody>
</table>

This values can be compared with the average of consumption of a Home in Spain that is around 3500 KWh (IDAE). The development stage represents 5.5% of this consumption and also de production one is over 21%.
Chapter 7. Conclusions

The first conclusion that can be made after analysis is the viability to develop a custom set of parsers based on Capybara, based on the information shown in the first chapter.

Even if the results of the project compounds a fully functional tool there are few issues that could not be observed due to time limitation. These points are:

- Management of the product’s stage even a strategy to maintain the products each execution of WEST the development of this strategy could not be developed.
- Work as daemon. All the parsers developed regarding to the brands are prepared to run in daemon mode (it means with start/stop options and running in the background) but it is still needed to develop a manager in charge of launching each parser every defined time period.

The results of the project in terms of software architecture are detailed quite enough for the scope of the initial work plan. According to the evolution of this project, it is important to highlight that it is recommended to add some more classes to divide and modularize more the design provided. Thanks to the evolution of the code and the addition of new features, add to the platform new brands will be easy in the future.

The performance of WEST has not been a main issue in the project requirements due to the low frequency of WEST execution (initially one time every week). Nevertheless it is important to considerate the high number of contained products in each website and perform as soon as possible the management of the product’s state properly. In order to reach a better performance, the main recommendation is to try to insert each product to the DB each time is gathered instead of make it in blocks. This strategy will allow only gathering the items related with the duplication detection instead of gather all the information for check it later.

As explained in CHAPTER 5 Legal Issues related to this project have an important impact and will be relevant, not only for the evolution and viability of WEST, but also of entire platform. It is important to try to solve the problems regarding to European directives obtaining the right agreements in order to keep the activity of Trenditlive in a legal place.
Bibliography


**Scrappy** [En línea]. - http://search.cpan.org/~awncorp/Scrappy-0.62/.


ANNEX 1. Brands Structures

1.1. Brand navigation structures previews

1.1.1. Bershka

Fig 16. Bershka general structure

Fig 17. Bershka product list
Bershka structure is based in a horizontal bar with section, categories appear when mouse pointer is over each section unfolding a menu where the user can click into the desired category to access. Once an user click in a category he access to the product list page, with all the products appears in different ways, in this case all the products are loaded and the user only can switch if he want to see 3 products per row or 4 products row.

At the end the user access to the product information page and it is from here where WEST can gather all the information related to the product.

Fig 18. Product Information page
1.1.2. Desigual

Fig 19. Desigual general structure

Fig 20. Desigual product list
Fig 21. Desigual product information

Desigual has exactly the same structure for sections and categories, the main different for this brand is in the product list page. The navigation through the products are ordered following a scheme based on pages, the user can move through the pages using the navigation bar.

Related to the product information page is quite similar to Bershka, in any case, the kind of information available in each proposed brand will be explained later.
1.1.3. Mango

Fig 22. Mango sections and categories

Fig 23. Mango product list
Fig 24. Mango product information

Mango’s structure agree with Bershka and Desigual referred to sections and categories. Also in this case a difference can be observed in the product list due to Mango strategy is to load all the products form a category at the same time and in only one page. This difference in the navigation of the product list will produce different ways to implement the go across the products.
1.1.4. Massimo Dutti

Massimo Dutti change the strategy of sections and categories order, in this there is a full page where a vertical bar on the left, contains a hierarchy of sections and categories. This difference would not affect to the scraping implementation of this part of the webpage but in any case is better in order to filter the needed information due to all the objects are visible in the page. In the other cases the function to auto-display using mouse over need and special way to filter in order to find hidden elements (this issue will be discussed in depth later) Massimo Dutti has the same way to present the product list than Mango, all products in the same page.
1.1.5. H&M

Fig 27. H&M sections and categories

Fig 28. HM Product List
Fig 29. HM Product Information

The structure of HM is quite similar to Mango and Desigual it has a main bar where can be found sections and a submenu is activated when is clicked with mouse. In this case there is an important difference in the way that is presented the product list, HM uses an infinite scroll system to present the items to customer. This difference is important because involve another kind of scraping to gather all products of category.

Product list information is presented in the same way.
1.1.6. Zara

Fig 30. Zara sections and categories

Fig 31. Zara Product list
Develop a generic test-bed for web scraping

Zara is the last brand analyze and follow the structure explained before, in particularly sections and category structure follow the same structure as Massimo Dutti and list the products putting all in the same page as Mango.

Fig 32. Zara Product Information
### ANNEX 2. Normalizer taxonomy

<table>
<thead>
<tr>
<th>Clothe type</th>
<th>Normalized type</th>
<th>Clothe type</th>
<th>Normalized type</th>
<th>Clothe type</th>
<th>Normalized type</th>
<th>Clothe type</th>
<th>Normalized type</th>
</tr>
</thead>
<tbody>
<tr>
<td>chaqueta</td>
<td>jersey</td>
<td>vestido</td>
<td>monedero</td>
<td>carteras</td>
<td>fragancia</td>
<td></td>
<td></td>
</tr>
<tr>
<td>abrigo</td>
<td>sudadera</td>
<td>body</td>
<td>zapato</td>
<td>collar</td>
<td>pañuelo</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cazadora</td>
<td>sudaderas</td>
<td>bodies</td>
<td>botín</td>
<td>pendientes</td>
<td>diadema</td>
<td></td>
<td></td>
</tr>
<tr>
<td>parka</td>
<td>camiseta</td>
<td>polo</td>
<td>slipper</td>
<td>anillo</td>
<td>mochila</td>
<td></td>
<td></td>
</tr>
<tr>
<td>americana</td>
<td>camisa</td>
<td>mono</td>
<td>botín</td>
<td>pendientes</td>
<td>pendientes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>blazer</td>
<td>camisas</td>
<td>monos</td>
<td>tacon</td>
<td>pendientes</td>
<td>pendientes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>beisbolera</td>
<td>cardigan</td>
<td>top</td>
<td>tacon</td>
<td>pendientes</td>
<td>pendientes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>chaleco</td>
<td>cardigans</td>
<td>tops</td>
<td>cuña</td>
<td>brazalete</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chaqueton</td>
<td>pantalon</td>
<td>faldas</td>
<td>stiletto</td>
<td>cinturon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gabardina</td>
<td>bermuda</td>
<td>vaqueros</td>
<td>bambas</td>
<td>gafa</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>trench</td>
<td>short</td>
<td>short</td>
<td>blucher</td>
<td>reloj</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bolero</td>
<td>jeggin</td>
<td>legging</td>
<td>blusones</td>
<td>rojo</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>póngines</td>
<td>shorts</td>
<td>blusones</td>
<td>reloj</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>jeggin</td>
<td>vaqueros</td>
<td>bolso</td>
<td>gafa</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>pantalones</td>
<td>vaqueros</td>
<td>bolso</td>
<td>gafa</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>bolsos</td>
<td>gafa</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>accesorios</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>tarjetero</td>
<td>chal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libreta</td>
<td>albornoz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bolsas</td>
<td>mochila</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>espejo</td>
<td>auriculares</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gobi</td>
<td>tarjetero</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tarjetero</td>
<td>chal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>albornoz</td>
<td>bolsas</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pulsera</td>
<td>marroquinería</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>esponja</td>
<td>neceser</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gorro</td>
<td>gorro</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pamela</td>
<td>gorro</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>llavero</td>
<td>albornoz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>llavero</td>
<td>albornoz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cuello</td>
<td>broche</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>broche</td>
<td>guantes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chal</td>
<td>alcayares</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>albornoz</td>
<td>bolso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tarjetero</td>
<td>chal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libreta</td>
<td>albornoz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>espejo</td>
<td>tarjetero</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>