



Universidade Nova de Lisboa
Faculdade de Ciências e Tecnologia
Department of Computer Science

Declarative Approach to Data Extraction of Web pages

By Ricardo Alves

Thesis submitted to Faculdade de Ciências e Tecnologia of the Universidade Nova de Lisboa, in
partial fulfilment of the requirements for the degree of **Master in Computer Science**

Supervisor: PhD João Moura Pires

Monte da Caparica, November 2009

Nº do aluno / Student number: 26084

Nome / Name: Ricardo João de Freitas Alves

Título da dissertação / Thesis title:

Declarative Approach to the Extraction of Relevant Data from HTML Files

Palavras-Chave / Keywords:

- Web Wrappers
- Web Data Extraction
- Graphical Wrapper Construction

Resumo

Temos, nos últimos anos vindo a assistir uma acentuada evolução da WEB, com a introdução de significativos aperfeiçoamentos a nível tecnológico como é o caso do aparecimento do XHTML, do CSS, do Javascript, e da Web2.0, entre outros. Este facto, aliado a outros factores tais como a expansão física da WEB, e o seu reduzido custo, têm vindo a motivar a adesão de organizações e do público em geral, com um conseqüente incremento no número de utilizadores e dessa forma influenciando decisivamente o volume do maior repositório de dados a nível global.

Deste modo, surgiu uma crescente necessidade de aquisições regulares de dados a partir da WEB, que pela sua frequência, duração e complexidade, só seriam viáveis de obter através extractores automáticos. Todavia, os extractores automáticos têm duas dificuldades essenciais. Em primeiro lugar, grande parte da informação da Web é apresentada em formatos visuais principalmente orientados para a leitura. Em segundo lugar, a introdução de páginas WEB dinâmicas, as quais são compostas em memória local a partir de diferentes origens, ocasionando que algumas dessas páginas não tenham um ficheiro fonte.

Assim, a presente tese vem propor um novo extractor mais moderno, capaz de suportar a evolução da Web, bem como ser genérico de forma a poder ser utilizado em qualquer situação, e com a capacidade de ser estendido e facilmente adaptável a utilizações mais específicas. Este projecto começou por ampliar um anteriormente existente com capacidade de efectuar extracções sobre ficheiros de texto semi-estruturados, todavia evoluiu para um sistema de extracção modular capaz de extrair dados de páginas Web, ficheiros de texto semi-estruturados e ser alargado de forma a suportar outros tipos de fontes de dados. O mesmo contém ainda um sistema de validação de dados mais completo e genérico e um novo sistema de entrega de dados capaz de realizar as anteriores entregas, bem como novas mais genéricas.

Foi ainda desenvolvido um editor gráfico para suportar as funcionalidades do sistema de extracção e permitir a um perito do domínio sem conhecimentos de informática de criar extracções com apenas algumas interacções simples e intuitivas sobre a página Web renderizada.

Abstract

In the last few years, we have been witnessing a noticeable WEB evolution with the introduction of significant improvements at technological level, such as the emergence of XHTML, CSS, Javascript, and Web2.0, just to name ones. This, combined with other factors such as physical expansion of the Web, as well as its low cost, have been the great motivator for the organizations and the general public to join, with a consequent growth in the number of users and thus influencing the volume of the largest global data repository.

In consequence, there was an increasing need for regular data acquisition from the WEB, and because of its frequency, length or complexity, it would only be viable to obtain through automatic extractors. However, two main difficulties are inherent to automatic extractors. First, much of the Web's information is presented in visual formats mainly directed for human reading. Secondly, the introduction of dynamic webpages, which are brought together in local memory from different sources, causing some pages not to have a source file.

Therefore, this thesis proposes a new and more modern extractor, capable of supporting the Web evolution, as well as being generic, so as to be able to be used in any situation, and capable of being extended and easily adaptable to a more particular use. This project is an extension of an earlier one which had the capability of extractions on semi-structured text files. However it evolved to a modular extraction system capable of extracting data from webpages, semi-structured text files and be expanded to support other data source types. It also contains a more complete and generic validation system and a new data delivery system capable of performing the earlier deliveries as well as new generic ones.

A graphical editor was also developed to support the extraction system features and to allow a domain expert without computer knowledge to create extractions with only a few simple and intuitive interactions on the rendered webpage.

Acronyms

<i>Acronym</i>	<i>Description</i>
API	Application programming interface
CSS	Cascading Style Sheets
CSV	Comma Separated Values
DOM	Document Object Model
DPM	Data Processing Module
ETD	Extraction, Transformation and Data Delivery
ETL	Extraction, Transformation and Loading
FFD	File Format Definition
FTP	File Transfer Protocol
GID	Global identifiers used on the SESS System
GUI	Graphical User Interface
HEL	HTML Extraction Language
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IL	Integration and Loading
JDBC	Java Database Connectivity
JDIC	Java Desktop Integration Components
JNI	Java Native Interface
MathML	Mathematical Markup Language
NSL	Nested String Lists
SEIS	Space Environment Information System

SESS	Space Environment Support System
SGML	Standard Generalized Markup Language
SMTP	Simple Mail Transfer Protocol
SVG	Scalable Vector Graphics
SWT	Standard Widget Toolkit
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
W4F	WysiWyg Web Wrapper Factory
WEB	World Wide Web (commonly abbreviated as "WEB")
WYSIWYG	What You See Is What You Get
XGML	Standard Generalized Markup Language
XHTML	Extensible HyperText Markup Language
XML	Extensible Markup Language
XPATH	XML Path Language
XPCOM	Cross Platform Component Object Model
XSL	Extensible Stylesheet Language
XSLT	Extensible Stylesheet Language Transformations
XUL	XML User Interface Language

Index

Chapter 1 - Introduction.....	1
1.1 Motivation.....	2
1.2 Problem Description.....	5
1.2.1 WebPages and HTML.....	5
1.2.2 The Deep Web.....	9
1.2.3 Web Data Extraction.....	9
1.2.4 Problem Description.....	12
1.3 Scope and Context.....	13
1.3.1 Presentation of the SESS Architecture.....	13
1.3.2 Scope and Context.....	17
1.3.3 Proposed Approach.....	18
1.4 Major Contributions.....	18
1.5 Thesis Structure.....	19
Chapter 2 - Related Work.....	21
2.1 HTML Data Extraction Projects and Products.....	22
2.1.1 Lixto Web Extraction Platform.....	22
2.1.2 Effective Web Data Extraction with Standard XML Technologies.....	23
2.1.3 WysiWyg Web Wrapper Factory (W4F).....	25
2.1.4 Data Extraction Products on the Web.....	28

2.2 Interactions on a Rendered Webpage	30
2.2.1 iPhone.....	31
2.2.2 HTML Editors.....	32
2.2.3 Proposed Interactions by W3C	32
2.3 Rendering Engines.....	34
2.3.1 JREx.....	34
2.3.2 Lobo: Java Web Browser.....	35
2.3.3 Internet Explorer trough Jawin Lib.....	36
2.3.4 Apple Webkit.....	36
2.3.5 JDIC (JDesktop Integration Components).....	37
2.3.6 SWT Library Browser Component.....	37
2.3.7 Rendering Engines Conclusion.....	38
2.4 Conclusions.....	38
Chapter 3 - SESS Extraction Detailed Implementation.....	41
3.1 Sections.....	42
3.2 Fields.....	43
3.3 Transformation.....	46
3.4 Data Delivery.....	47
3.5 The FFD Editor.....	49
3.5.1 FFD Extract Tab.....	49
3.5.2 FFD Transform Tab.....	51
3.5.3 FFD Data Delivery Tab.....	51
3.6 Conclusion.....	51

Chapter 4 - Architecture.....	53
4.1 Extraction System Overview.....	54
4.2 Extraction Module.....	56
4.2.1 Sections.....	57
4.2.2 Fields.....	60
4.3 Data Delivery Module.....	66
4.3.1 Metadata.....	67
4.3.2 Data Columns.....	68
4.3.3 Validation.....	68
4.3.4 XSLT Process.....	68
4.3.5 Save To.....	68
4.3.6 Output Delivery.....	69
Chapter 5 - Implementation.....	71
5.1 Supporting Multiple Source Types.....	72
5.2 Webpage Data Extractor.....	73
5.3 New Data Delivery	74
5.4 The Graphical Editor.....	75
5.4.1 The Extraction Editor.....	76
5.4.2 The Webpage Extraction Tester.....	84
5.4.3 The Data Delivery Editor.....	85
5.5 Validation.....	87
5.5.1 Stock Market.....	87
5.5.2 Electronic Stores – Kuantokusta Delivery.....	90

5.5.3 News Websites.....	94
5.5.4 Sports Websites – World Rally Championship.....	97
5.5.5 Educational Information.....	98
5.5.6 Sports Websites – Soccer.....	99
5.5.7 Electronic Stores – Amazon.....	101
5.5.8 Validations Conclusion.....	102
Chapter 6 - Conclusions and Future Work.....	105
6.1 Conclusions.....	106
6.2 Future Work.....	107
Chapter 7 - References.....	111

Figure Index


Figure 1: (A) Trains Departures & Arrivals (B) Flights Departures and Arrivals.....	3
Figure 2: (A) Formula 1 Example (B) WRC Example.....	4
Figure 3: Weather Examples.....	4
Figure 4: (A) Auction Example (B) E-Store Example.....	5
Figure 5: SESS Architecture Diagram.....	13
Figure 6: ETD + IL pipeline [1].....	14
Figure 7: Data Processing Module architecture [1].....	15
Figure 8: The File Format Definition model [1].....	16
Figure 9: The FFD Editor ETD tabs [1].....	17
Figure 10: Webpage Zoom Feature, present on the iPhone.....	31
Figure 11: Range selection example [27].....	33
Figure 12: Sample of a Semi-Structured Text File.....	42
Figure 13: Section XMLSchema.....	43
Figure 14: Single Value XMLSchema.....	44
Figure 15: Table XMLSchema.....	45
Figure 16: Data Typing and Validation Rules XMLSchema.....	46
Figure 17: Transformation XMLSchema.....	47
Figure 18: Data Delivery XMLSchema.....	48
Figure 19: Template XMLSchema.....	48

Figure 20: The FFD Editor Tabs.....	49
Figure 21: Extraction Tab ScreenShot.....	50
Figure 22: Architecture Basic Structure.....	54
Figure 23: FFD XMLSchema with the extraction languages separated.....	56
Figure 24: Section XMLSchema.....	58
Figure 25: Section Validation Rule definition XMLSchema.....	59
Figure 26: Field Element XMLSchema.....	60
Figure 27: Webpage Extraction Expression XMLSchema.....	61
Figure 28: Single Value XMLSchema.....	62
Figure 29: Special Value Field XMLSchema.....	62
Figure 30: Table Value XMLSchema.....	63
Figure 31: Value Validation Mechanism XMLSchema.....	64
Figure 32: Textual Validation Type XMLSchema.....	64
Figure 33: Numeric and Date Validation Type XMLSchema.....	65
Figure 34: User Define Validation Type XMLSchema.....	65
Figure 35: Field Missing Values XMLSchema.....	66
Figure 36: DataDelivery XMLSchema.....	67
Figure 37: Delivery output basic structure.....	69
Figure 38: Delivery output Data element XMLSchema.....	70
Figure 39: XSLT Process.....	70
Figure 40: Webpage extraction component interface.....	76
Figure 41: Addition of a new Section.....	78
Figure 42: Addition of a new Single Field.....	79


Figure 43: Addition of a new Table Field.....	80
Figure 44: Addition of a new Table Column.....	81
Figure 45: Special Field Wizard.....	81
Figure 46: Automatic extraction expressions generator mechanism Diagram.....	84
Figure 47: Webpage Extraction Tester interface.....	85
Figure 48: Data delivery editor interface.....	86
Figure 49: Site "bolsa.sic.pt".....	87
Figure 50: Creating the Stock Market Extraction.....	88
Figure 51: Stock Market Extraction Results.....	89
Figure 52: Manipulated webpage.....	90
Figure 53: KuntoKusta Sample File.....	91
Figure 54: Nuno Candeias Sample Product List.....	91
Figure 55: Visualizing the Extraction.....	92
Figure 56: Extraction First Test Results.....	93
Figure 57: Extraction Test Results on other pages.....	93
Figure 58: Delivery SaveTo type.....	94
Figure 59: Sample of the Resulting File.....	94
Figure 60: Engadget Webpage.....	95
Figure 61: Visualizing the Extraction.....	96
Figure 62: Engadget Extraction Results.....	96
Figure 63: Stage Times.....	97
Figure 64: Visualizing the Extraction.....	97
Figure 65: Stage Times Extraction Results.....	98

Figure 66: Webpage Containing School Subjects information.....	98
Figure 67: The Extraction Tree.....	99
Figure 68: Extraction Results.....	99
Figure 69: Soccer Player Details webpage.....	100
Figure 70: Visualizing the Extraction.....	100
Figure 71: Extraction Results.....	101
Figure 72: Amazon Digital Cameras Section webpage.....	101
Figure 73: Visualizing the Extraction.....	102
Figure 74: Amazon Extraction Results.....	102

Chapter 1 - Introduction



This chapter presents the motivation, problem description and the context of the thesis.



1.1 Motivation.....	2
1.2 Problem Description.....	5
1.3 Scope and Context.....	13
1.4 Major Contributions.....	18
1.5 Thesis Structure.....	19

The aim of this thesis is to build a system allowing a user without computer-science knowledge to create a data extraction process on webpages. To achieve this, it was decided to extend a previous project which already performed data extraction on semi-structured text files, but now creating a more complete system, holding the ability for data extraction on dynamic pages as those of Web 2.0.

In this chapter, we begin by explaining the motivation for this thesis in the first section, presenting two systems and particularly the solution for data extraction of one of them, which is shown as being a good platform that could possibly be evolved for the WEB. In the next section it is described the Web's evolution and the problematic of data extraction from webpages demonstrating the need for a new solution. This chapter ends with two sections, the first being a description of the proposed solution for the thesis, explaining in detail the system to be extended, and the last section highlights the thesis's major contributions.

1.1 Motivation

The “SEIS” (Space Environment Information System) and “SESS” (Space Environment Support System) systems, were developed mainly for the collection and integration of data on space weather conditions and space telemetry from the spacecrafts in "almost" real-time. They also enable access to historical data, with the purpose of providing it for analysis and subsequent decision-making. This telemetry and space weather data was available in many and different formats in semi-structured text files present on several data providers. Therefore to extract data from these files, a complete solution of "ETL" (Extraction, Transformation and Loading) was developed for the "SESS" system. This solution is described in the Master's thesis by Ricardo Fortuna Raminhos, entitled "Extraction and Transformation of semi-structured data from text files using a declarative approach" [1].

A classic “ETL” solution is an extracting system of data from multiple sources that transforms and returns it into a predetermined format. It normally requires procedures to be established by computer experts, which don't always have enough knowledge in the data domain, causing a rise in the task's time, and increasing the probability of errors. So in the referenced thesis, “ETL” has been divided into "Extraction, Transformation and Data

Delivery" (ETD) and "Integration and Loading" (IL). Creating therefore a separation between the tasks to be performed by data domain experts, like the extraction and transformation (ETD), from the more technical tasks to be performed by experts of computer technology, like the integration and loading of data into management systems (IL).

This project was a success in its scope, however it is considered that the real capabilities in the system are not fully exhausted in terms of application, as it involves concepts and features that could be used in other areas. An example is the Internet, which could be regarded as more attractive and appropriate in terms of usefulness and applicability with this solution, after all this mean of communication has the highest growth rates being the largest global existing data repository.

The question is primarily focused on the change needed to be made so that the system not only be based on plain text files, where its visualization is equal to its rendering (Raw = Render), but simultaneously supports extraction features from webpages, which are built to be viewed and understood by humans. In this aspect, the direct view of the source is different from the view of its rendering (Raw ≠ Render). This makes the automatic extraction of information from the webpages very difficult to be created by users without training in computer technology. Thus, there is a need to hide the technical details from all the extraction process so that it might be used by different users to extract different types of data. In the figures 1, 2, 3, 4 are presented some examples of webpages that contain data which would be interesting to extract.

Examples of Information on Public Transportation

LIVE DEPARTURES: LONDON KINGS CROSS (KGX)

Last updated: Thursday 10 July 2008 10:48:56 - automatically refresh this page

- View London Kings Cross Arrivals
- London Kings Cross Station Information
- Service Bulletin affecting London Kings Cross
- Call TrainTracker™ for London Kings Cross on 0871 200 49 59 or text KGX to 84550 to use TrainTracker™ Text

DESTINATION	TIMETABLE	EXPECTED	OPERATOR	TRAIN DETAILS
Cambridge	10:52	Starts here	East Coast Connect	Details
Glasgow Central	11:00	Starts here	National Express East Coast	Details
Cambridge	11:06	Starts here	East Coast Connect	Details
Leeds	11:10	Starts here	National Express East Coast	Details
Cambridge	11:15	Starts here	East Coast Connect	Details
Fife/Dunfermline	11:22	Starts here	East Coast Connect	Details
Newcastle	11:30	Starts here	National Express East Coast	Details
Leeds	11:35	Starts here	National Express East Coast	Details
Fife/Dunfermline	11:36	Starts here	East Coast Connect	Details
King's Lynn	11:45	Starts here	East Coast Connect	Details

(A)

ANZ Aeroportos

Damos vida aos aeroportos

Porto Algarve Açores

10.07.2008 10:36:30

Lisboa Partidas

Última actualização: 10 Jun 2008 10:34

Destino	Data	Hora	Voo	Companhia	Destino	Estado	Tracking
Lisboa	10/07	09:00	EZ31440	easyJet	Geneve	T1	Partiu: 09:25
Cafés e Restaurantes	10/07	09:00	TP 400	TAP Portugal	Isarville	T1	Partiu: 09:14
Guia Turístico	10/07	09:00	TP 402	TAP Portugal	Nova	T1	Partiu: 09:18
Sobre o Aeroporto	10/07	09:05	SN 3814	Brussels Airlines	Bruxelas	T1	Partiu: 09:23
	10/07	09:05	TP 500	TAP Portugal	Stockholm	T1	Partiu: 09:29
	10/07	09:05	TP 770	TAP Portugal	Bilbao	T1	Partiu: 09:20
	10/07	09:15	TP 746	TAP Portugal	Barcelona	T1	Partiu: 10:09

(B)

Figure 1: (A) Trains Departures & Arrivals (B) Flights Departures and Arrivals

Figure 1 presents two examples from sites that offer real-time information on arrivals and departures of public transportations. The first, relating to trains in London stations, and the second with flights at Lisbon airport.

Examples of Information on Sports

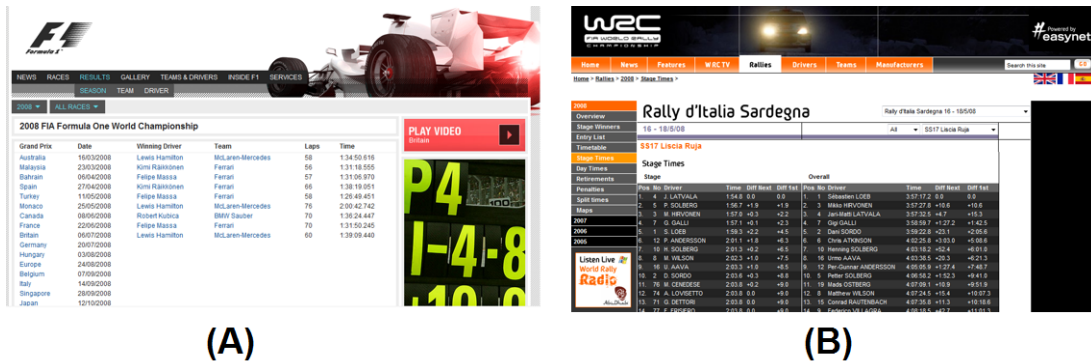


Figure 2: (A) Formula 1 Example (B) WRC Example

Figure 2, presents two examples of sites providing considerably extensive information about motor sports, Formula 1 and WRC. In the Formula1 webpage the pilots and builders championship points are listed and in the WRC webpage the pilots times in a given special stage.

Examples of Information on Weather

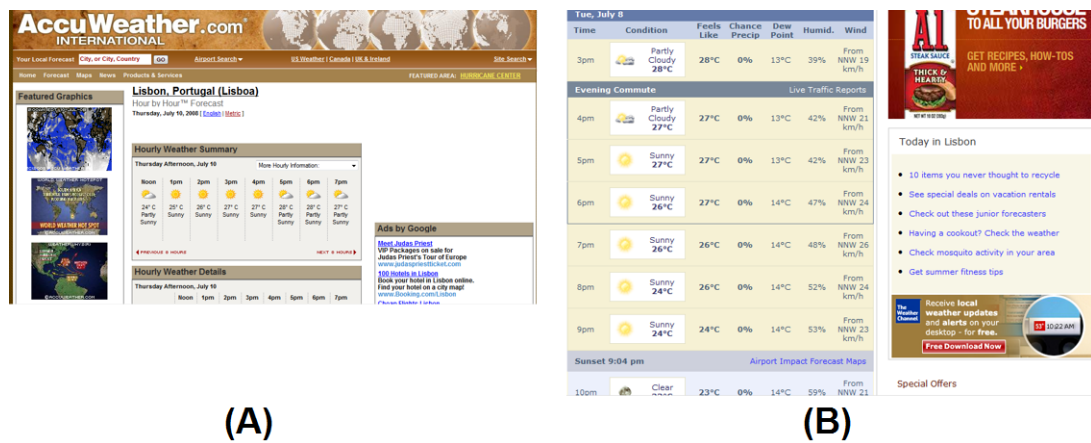


Figure 3: Weather Examples

The two examples, shown in figure 3, are from sites that offer a lot of information on climate of various geographical regions.

Examples of Electronic Market

(A) Auction Example

1,519 items found [Save this search]
View as: List [Customise view] Sort by: Best Match [v]

Featured Items	Price	Package to Post	Time Left
Maxon Start Lead V3 NEW FILTERS COPY PRO REMOVER DVD	£19.99	+£7.00	2h 26m
Marantz DV6001 DVD Player SACD CD Multi Region DV-6001 From Hong Kong	£182.00	not specified	3m
PHILIPS DVP1960 Multiregion DVD player "HDML & DivX"	£24.00	not specified	18m
Philips DVP1200 Multiregion DVD Player Ultra Slim MP3	£9.00	not specified	48m
Nextbase Region Free SDV37-S Portable DVD player	£80.00	not specified	1h

(B) E-Store Example

159 productos que corresponden a sua pesquisa: [Visualizar todos os produtos em uma tabela]

Produto	Preço	Disponibilidade	Venda	Oportunidade de compra	Preferir Comprar
FUJIFINEPIX S8000 IS Entregue com pilhas A Fujifilm apresenta a FinePix S8000 IS com uma objetiva de forte amplitude de 18x, num corpo ergonómico com design...	207 €	EM STOCK			
SONY Cyber-shot DSC-H50B - Preta A Sony apresenta a Cyber-shot DSC-H50B, uma bridge eficiente e um preço atractivo. Esta máquina digital tem todo o equilíbrio...	292 €	EM STOCK			
FUJIFINEPIX F50 IS preta Entregue com carregador, bateria lithium A Fujifilm apresenta a sucessora de sua famosa F3116 com a FinePix F50, que não só aumenta de suas capacidades, como também...	187,44	EM STOCK			
CANON Digital Ixus 860 IS prateada Entregue com carregador, bateria lithium, SD Card 32 mb No lugar de a sua famosa série Digital Ixus, a Canon lança uma máquina com funcionalidades avançadas que já é uma verdadeira...	239 €	EM STOCK			
CANON PowerShot A720 IS Entregue com pilhas, SD Card 16 mb Introduzindo uma tecnologia de vídeo de vídeo, a Canon PowerShot A720 IS é a sucessora da famosa A7100, conservando o seu...	185 €	EM STOCK			

Figure 4: (A) Auction Example (B) E-Store Example

Figure 4 presents two site examples for the sale of products. The first from an auction site and the second from an electronics store.

1.2 Problem Description

This section details the problematic that was previously outlined, highlighting the arising difficulties. The WEB evolution is described starting from the simple pages only written in HTML, leading to the current webpages that are dynamically composed from multiple sources on the cybernaut's computer. The technical hitches and difficulties of the web data extraction, like the need to extract data from the rendered page instead of its source file, are also explained in detail.

1.2.1 WebPages and HTML

HTML is an acronym for "HyperText Markup Language", which is one of the markup languages most often used on webpages. Markup languages are languages that use a set of notations in the text describing how their content should be structured and viewed or formatted. The HTML descends from the SGML (an older markup meta-language) permitting to denote specific text as links, header, paragraphs, lists, or others, enabling the addition of amongst other things; forms, images, scripts, and so on.

The HTML pages were made for showing a single graphical representation on screen, mixing

both contents and graphical appearance in a unique file. Therefore, to create a better separation between the content of the HTML page and its appearance the CSS ("Cascading Style Sheets") has emerged [2]. CSS is a language of styles created by the W3C (World Wide Web Consortium) [3] and used to describe the presentation of a document written in a markup language. This is a widely used language to define how webpages, written in HTML, XHTML and XML should be presented to the user. It can also be applied to other documents such as SVG (Scalable Vector Graphics), XUL (XML User Interface Language) which are XML languages[4].

Separating the contents from the presentation using CSS improves the contents accessibility by in one hand, increasing the flexibility and control over the design, and in the other hand, reducing the complexity and recurrence of the presentation rules. This separation also allows the same document to have several different appearances depending on the environment where it is being shown, in a screen, printer, or even in a Braille device.

At this moment the CSS2.1 is the current version, recommended since July, 19 of 2007 by the W3C, and it is implemented in most of the rendering engines used in web browsers.

One of the most common examples of usage of this language, is having several similar pages to publish on the Internet, the CSS allows us to use only a single file with the CSS presentation rules for all, yet permitting the change of their appearance through modification only in the CSS file.

Since 2000, the HTML was extended to a new language called XHTML (Extensible HyperText Markup Language) [5], which is based both on HTML and XML. Its new specification requires that these documents be well-formed XML documents while using the familiar HTML elements and always following the XML rules. The XML syntactic rules like for example the requirement of termination any element, the attribute values contained within double quotes, make the document more rigorous and robust, allowing them to be easily interpreted by any generic XML interpreter. This is an important advantage, since to interpret HTML it is necessary to use a specifically develop interpreter which has to do its job with incomplete, ill-formed code or even with errors. This means more work for the browsers which spend two thirds of processing time trying to recover and correct the HTML from the source file in order to be able to build the webpage rendering. Also, this problem extends to

all software which deals with webpages. To solve it, the programs have to correct it first using for example the HTML Tidy Library project, or incorporate in itself a web browser and get the target data through it (thus avoiding to deal with ill-formed html code). Another new feature of the XML-based language is the possibility of using "namespaces" within the document, providing the ability to integrate segments or sections with different XML-based languages, such as MathML [6] (which provides a foundation for the inclusion of mathematical expressions in Web pages), and SVG [7] (which is used for describing two-dimensional graphics and graphical applications in XML), inside the XHTML document. For the data extraction problem that is also important because there isn't a limited set of pre-known tags any more. Today these languages are in use but in the future depending on the advantages and business there will be other ones, making the work of automated programs more difficult.

Because the documents in HTML, XML and XHTML always create an hierarchy among its elements, it makes it possible to build an object model, when it is interpreted. The referred object model or DOM (Document Object Model) [8] is a neutral interface proposed by the W3C, which allows programs and scripts to dynamically access and modify the content, structure and style of documents. This even allows the processing of a document, incorporating the results of the operation back to the current page. Such easy usage makes it now the best method of processing and/or data extraction, becoming for this reason a widely used standard on programming.

However, to allow some interaction between the website and the user, it is necessary that the code of a programming language be introduced inside it, in order to be interpreted and executed by the browser, thus enabling richer and more dynamic pages. The most used language for this purpose is the Javascript. Being a "scripting" language created by Brendan Eich [9] in 1995, who worked for "Netscape Communications Corporation", currently "Mozilla Foundation". It is an interpreted multi-paradigm language, based on prototype, and influenced by several other already existing languages, such as Self, C, Scheme, Perl, Python and Java. Although designed to be similar to Java, its name caused some confusion and was considered by many as a marketing trick [10] from Netscape.

Javascript is specially known for being widely used in webpages and executed completely by

the client, thus creating interactivity with the user. Some examples are the opening of menus, new windows, validation of forms, use of responsive images to the cursor movement, sections of pages that are displayed or hidden by action on a link, and others. This is possible because the browsers allow scripts to access, interact and modify the page object model that is being rendered. The referred feature creates a great dynamism on its graphical representation making it more interesting for the users visualizing it. However the object model used to graphically display the page begins to differentiate itself from the initial hierarchy its file source had. This becomes a problem when processing or extracting the data contained in the page, being aggravated in the WEB2.0 as explained next.

The next step in the web evolution was the Web 2.0. The term Web 2.0 [14] was first introduced by Tim O'Reilly in the conference entitled "The Web as Platform" in 2004, later known as the "Web 2.0 conference" [15]. Although the term suggests a new version of the World Wide Web it doesn't refer to an upgrade of the network technology, but in fact to a major change of the Web usage by programmers and Internet users. This change promotes creativity, information sharing and collaboration amongst internauts. Some examples of these new technologies are the "weblogs", "wikis", "podcasts", "Social software", and web APIs, which improved the read-only static sites enabling its users to interact and perform many other actions.

All of these can be built on top of the interactivity features supported by the currently designated "Web 1.0", providing the "Network as a platform" and enabling users to run applications through their own browsers. Most of their "user-friendly" interfaces are implemented using Ajax¹ or Flex². These are supported on scripts that modify the object model of the rendered page, and so in the simplest case differences are created between this model and the one in the original source file. Furthermore these scripts can even start requests to HTTP servers incorporating dynamically and in real time the resulting answers in the

1 Ajax, or AJAX (Asynchronous JavaScript and XML), is a group of interrelated web development techniques used to create interactive web applications or rich Internet applications. With Ajax, web applications can retrieve data from the server asynchronously in the background without interfering with the display and behavior of the existing page.

2 Flex is a collection of technologies released by Adobe Systems for the development and deployment of cross-platform rich Internet applications based on the proprietary Adobe Flash platform.

object model of the webpage being rendered. Thus, composing it through a variety of information from several different sources on the Internet, eventually leading to the fact, that the concept of the source file from the webpage become inexistent.

1.2.2 The Deep Web

The "Deep Web" or "Hidden Web" [11], is the content present in the World Wide Web that is not indexed by search engines. It includes content which is only accessed by filling out forms, pages that have no link to themselves from other pages (backlinks), content of Private Webs (which are sites that require the users be registered), content of Contextual Web (which is content that varies depending on the user), Limit access content (which are pages that restrict the content), Scripted content (pages that are only accessible through JavaScript, Flash or Ajax), or content that is in multimedia formats not interpreted by search engines. It was in 1994 that the first reference to this issue appeared with the term "Invisible Web" by Jill Ellsworth in the book "The Internet Business Book" [12]. Only in 2001 the term "Deep Web" appeared in the entitled study "The Deep Web: surfacing Hidden Value" written by Michael K. Bergman. It is known that the size of the Deep Web is much larger than the Surface Web. By the year 2000, its size was estimated to be about 550 times larger. By 2005 it was guaranteed as the sector with the fastest growth of the entire Web, containing 99% of the information on the Internet [13].

Thus, it is clear that it is not always possible to access a website from a URL (Uniform Resource Locator), but instead having to surf the Internet by following links, filling out forms, providing valid credentials for access to private sites, and running actions on pages to get the desired one. This becomes particularly difficult for automated systems to browse these pages without human aid.

1.2.3 Web Data Extraction

Given the huge amount of data on the Internet, automated processes become nearly indispensable to enable the extraction of such data, Therefore emerging the web wrappers.

In programming, the design of an adapter (often called a "wrapper") 'adjusts' the interface of a class to another one that the customer is waiting for. An adapter allows various classes to

work together whereas before it wasn't possible because of the incompatibilities among their interfaces. To achieve this, the interface of an existing class is involved, providing a new one which no longer contains incompatibilities. The adapter is also responsible for implementing all the necessary logic to transform the information in order to conform to what the consumer expects. For example, if you store multiple boolean values in an integer variable, but our consumer wants the returned values to be "true" or "false", the wrapper is responsible for extracting the appropriate values from the integer variable converting and returning them to the consumer.

In databases, a wrapper is a software component that converts data and queries from one model to another. For instance, when developing a program we create several modules or classes where all the necessary code to interact with the database management system is placed, converting the obtained results so that they can be processed by the application.

In the Web environment, the wrappers purpose is to extract and process the information stored in a HTML document, returning one other document with the structured data, possibly in XML to be subsequently processed.

1.2.3.1 Data Extraction from the Source File Versus Rendering of Contents

Since most of these current WEB data extractors do its own extraction on the data contained in the source file, this is similar to "Screen scraping". This is a technique consisting in a program to extract data from the output of another program, being the one making the extraction called "screen scraper". The key element that distinguishes "Screen scraping" from simple data interpretation, is that the output being processed was designed to be presented to a human user and not to be the input of another program, hence being unusual to have convenient structure for its interpretation.

There are many synonyms for "screen scraping", like "Data scraping", "web scraping", "page scraping", "web page wrapping" or "HTML scraping", being these last specific for WEB data extraction. However, given the possibility of being able to run JavaScript on webpages (when they are being viewed through a Web browser), as explained in section 1.2.1 the presented content may be different from the contents of the source file. So, there is a need of extracting information from webpages after its rendering, and to build a current and modern web

extraction tool it can not be based on the existence of a webpage source file. This is due to the discrepancy between an object model built using the data from the source file and an object model built for the web page viewing and rendering. This makes the extraction scripts (wrappers) created using a graphical tool that allows the selection of the elements (that we want to extract) within a webpage being viewed, very good for extracting information on a rendered webpage. However we can't guarantee that these scripts work properly when using them directly on the webpage source file.

1.2.3.2 Web Navigation

When we want to retrieve data from the web, the extraction of data from the HTML pages is only part of the problem. Before making the actual extraction we must learn how to surf to the pages that contain the data we are looking for. This stage is called "fetching" and that's where we found a series of new problems, such as password-protected sites, which need session identifiers, filling interactive forms on dynamic pages, among others. Only after overcoming these problems we get to the desired page, which is then transferred to the extraction module. This navigation problem becomes more serious in the extraction of data from the Deep Web, and the most common way to deal with it, is to record the user interactions on the pages, (e.g. driven links, text entered in the searches, or user name and password used to access a private area) in order to get the navigation path to the page containing the data to extract.

1.2.3.3 Graphical Interfaces for Building Web Wrappers

To allow the use of Web wrappers by users without computer training, it is necessary to hide some technical details. Thus, there should be a graphical interface to allow its construction, simplifying the whole process. However, several applications or systems for web data extraction don't provide this interface for creating web wrappers, requiring that the user knows the language and its rules used for the extraction. Others have very poor graphical interfaces, not giving the necessary support to the user. This mainly happens due to the implementing complexity a good graphical interface. They have to resolve, or to consider the following issues:

- It should allow the construction of wrappers, using all the capabilities of the extraction language.
- It shouldn't force the user to know the extraction language.
- It shouldn't create only absolute extraction expressions, it should interact with the user in order to improve the extraction expression as much as possible, trying to get one that would be resistant to small changes or updates on the target page.
- It should allow the user to view exactly the extraction target on the web.
- It should allow testing the "extraction script" in other pages, to ensure the script correctness and robustness.
- It should provide a validation mechanism in order to improve the quality of the extracted data.

1.2.4 Problem Description

Thus, the problem is to study, propose and implement a solution for data extraction on the WEB taking into account:

- That the specification of data to extract, as the extraction of data should work at the DOM object model level created by the browser, considering the new paradigms of the Web 2.0.
- The specification of the data to extract should be sufficiently generalized to allow the processing for the data extraction to be sufficiently robust to deal with several instances of the same webpage. Small changes in the format of the different instances of the same webpage should allow continuing with an accurate extraction of data or at least to permit the detection that a change is invalidating the process.
- The solution should allow a very active participation of domain experts (without knowledge of computer-science) in the specification of data to extract and if possible on the entire construction of extractors.

It should be noted that the related problem of browsing is outside the scope of this work and should be assumed as previously solved.

1.3 Scope and Context

This section presents an initial analysis of the ETD solution developed for the SESS System, which we intend to extend, not only improving it, but also to take advantage of the existing good capabilities incorporating them into the new web extracting system created in this thesis. After, the scope and purpose of this thesis in terms of the new proposed coverage as well as aspects related to implementation will be defined .

1.3.1 Presentation of the SESS Architecture

Data from space environment is composed of information on space weather, as well as data on the spacecraft, which are stored in semi-structured text formatted files, available in HTTP or FTP servers.

Both data types are the basis of two decision support systems. The “Space Environment Information System” (SEIS), which is a prototype system for singular missions. And the "Space Environment Support System" (SESS), which is a support system for multiple missions, providing spacial environment data "almost" in real time as well as historical data to the flight control teams. These systems are composed by several modules connected to each other like shown in the figure 5. It is in the "Data Processing Module" where the extraction system resides, which gathers all the input files, extracting, transforming and delivering the needed data to the other inner systems.

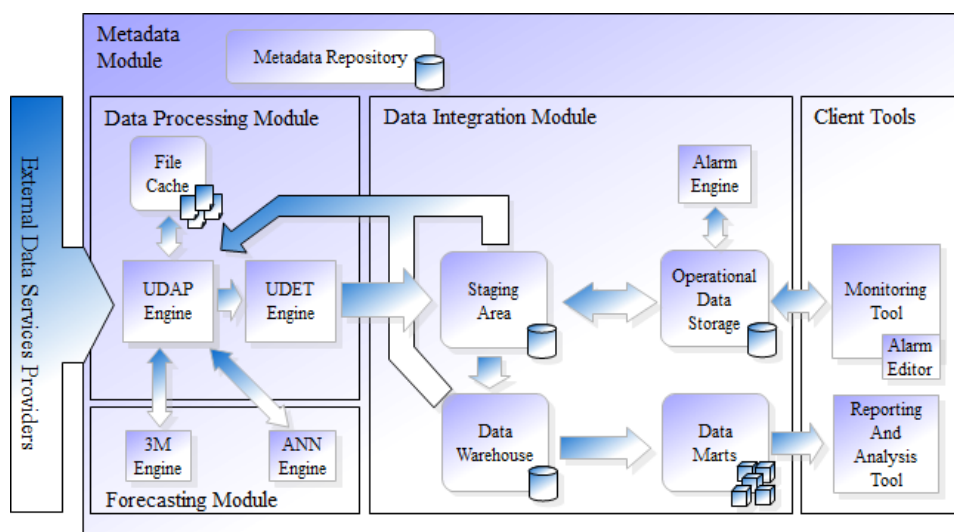


Figure 5: SESS Architecture Diagram

This extraction system presents a separation in its solution of Extraction, Transformation and Loading of data (ETL). A classic system for ETL extraction, extracts data from multiple data sources, ensuring their quality as well as their consistency during transformation, returning them in a predetermined format. This is not the most suitable approach for web data extraction, due to the amount of files in different formats requiring in turn different extraction rules. Forcing this work to be done by computer experts as well as data domain experts in strong cooperation among them. Their cooperation will necessarily result in an increase in the time spent for the job, as well as the possibility of errors. Also as the extraction is implemented by a computer expert, it is almost impossible for a domain expert to review or reuse it, since he certainly may not have the necessary computer skills. To solve these problems and to create a clear separation of concepts, the ETL has been divided into "Extraction, Transformation and Data Delivery" (ETD) and "Integration and Loading" (IL), as shown in figure 6.

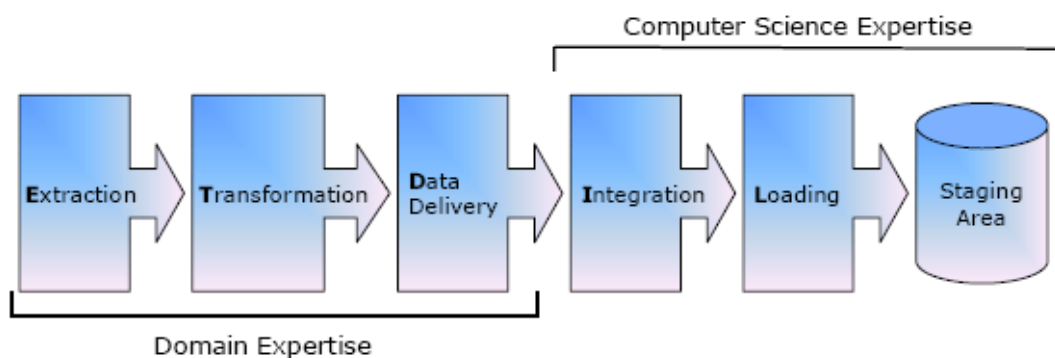


Figure 6: ETD + IL pipeline [1]

ETD is mostly composed by tasks that have to do with the raw data, which should only be performed by the domain expert being supported by a declarative extracting language and a graphical application that should make the language usage completely transparent to the user. IL is composed of tasks with technical operations, such as the introduction of these results in different data management systems, and thus more appropriate for the experts in information technology. So, with the differentiation between the data domain tasks from computer tasks, the time for the extraction solution is substantially reduced and the quality of extracted data is significantly improved, since the extraction is fully enforced and monitored by the domain expert. To implement the system's ETD solution, the "Data Processing" module was created.

This module is divided into six parts, showed in blue in figure 7.

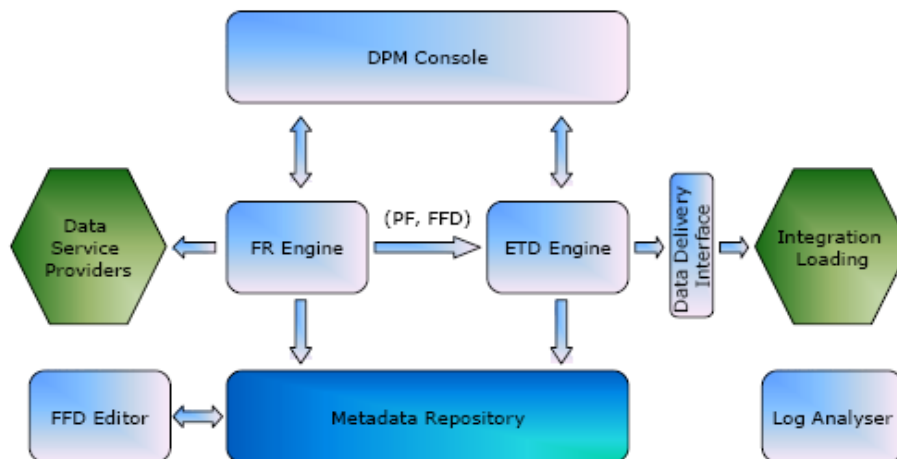


Figure 7: Data Processing Module architecture [1]

The remaining components in green are those of external service providers with which the data-processing module communicates. The dark blue represents the component of Metadata Repository implemented in the thesis by Ricardo Ferreira, also external to this module. The six components that constitute the data-processing module are:

File Retriever: This component is responsible for acquiring the data files from external service providers.

ETD Engine: This component is responsible for making the extraction and data transformation.

Data Delivery Interface: This component is responsible for returning the extracted and transformed data from the ETD Engine component, into an external loading data module of a third party service provider.

File Format Definition Editor (FFD Editor): This component is a graphical application that allows a user without computer training, to create, fix and test extraction scripts, which can be later interpreted and implemented by the three previous components, thus obscuring all the technical details from the extraction system.

DPM Console: This component is a graphical application that allows control and monitoring over all download actions and data processing within this module.

Log Analyzer: This component is a graphical application that allows analysis on the recorded information.

The File Retriever and ETD Engine components create a processing chain responsible for the download, extraction, and data transformation. They use web services to continuously implement this chain, allowing it to be working in the same or different machines, according to the policies of their organization, or the processing needs. This chain in operation uses a settings file, previously specified by the user through the FFD Editor, which is internally separated into distinct areas. Each of these divisions is a procedure of the model shown in figure 8.

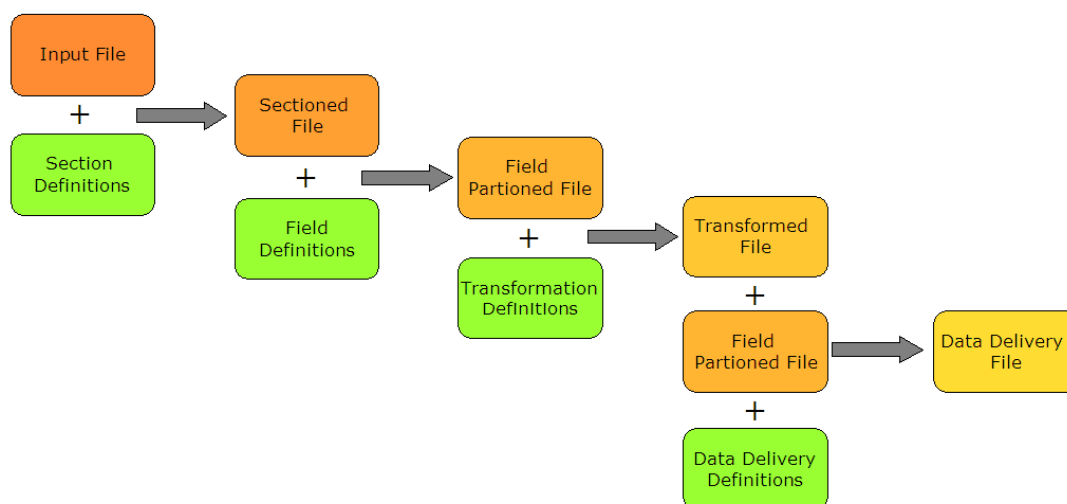


Figure 8: The File Format Definition model [1]

The proposal of dividing the extraction process, transformation and delivery of data in smaller cases having a corresponding section in the definitions file, was to follow the objective of "divide to conquer", thus facilitating its architecture, extensibility and implementation.

The extraction process is started by using the raw data file and the definitions contained in FFD to select the sections to process. From this first stage, it results in a sectioned data file, which is introduced in the second stage, as well as the definitions of the values to be extracted contained in the FFD. From the second one, extracted values within their sections have resulted, because they can only be defined in the context of a section. At this point the

extraction process is completed, transformation being the next procedure. There the inputs are the processed data from the previous extraction stage and the specifications of data transformations in the file FFD, resulting in the transformed data. Finally in the last stage, the settings for delivery of data are applied to both the results of the extraction and transformation, resulting in the data delivery file, which is the final output of the whole ETD solution.

In order to it make possible for a user who is only an expert in his domain to create an extraction, the FFD Editor component has been developed. This component allows setting all the details of the extraction, transformation and data delivery, without having to know the technical parts of the declarative language that supports the whole system. This is a graphical application divided into three tabs, corresponding each of them to one of the steps in this process: extraction, processing and data delivery, as shown in figure 9.

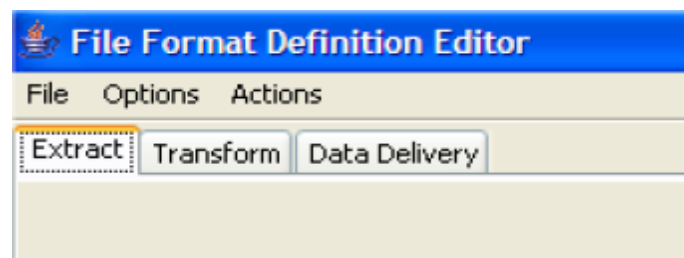


Figure 9: The FFD Editor ETD tabs [1]

On the "Extract" window tab, the user interacts with the area on which is presented a file with trial data, firstly dividing that file into sections and then indicating the values to extract within each one, making it possible to select single values as well as complete tables. On the "Transform" tab, the user can build a graph representing the transformations that will be applied to the values previously specified during the extraction phase. On the "Data Delivery" tab, the user organizes and selects the relevant data that are to be delivered to the recipient system.

1.3.2 Scope and Context

As previously mentioned, the aim of this thesis is the development of a modular data extraction solution, able to extract data from different source types in particular from the

WEB, which considering its evolution should be able to extract data from the document object model created for the web page rendering. As such the data type extracted in this system doesn't include images, videos, other multimedia formats and binaries, however the URL to those types of data are presented in the web page DOM and it is possible to extract them.

1.3.3 Proposed Approach

This project, which is intended to extend the existing work of Ricardo Raminhos, will be implemented on the same platforms and technologies such as Java, the graphic library SWT, XML and Web services.

In order to display a webpage rendering, the rendering engine available in the SWT library will be used. To allow an interaction between the user and the webpage in order to select the elements to extract, a layer will be developed in Javascript which will be introduced on the page after it is rendered. This layer will communicate with the main Java application through a path provided by the SWT browser component, transferring the information that the application needs to build the extraction script.

The extraction language in the previous project will have to be improved and the webpage extraction language module added. In this new module, XPath expressions will be used in junction with regular expressions, in order to specify the elements to extract from the inside of the object model.

1.4 Major Contributions

With this thesis we aim to create a generic and updated data extractor for webpages that considers in its development the entire evolution of the Web, and thus able to extract information on the more innovative web pages, as it is the case of some Web 2.0 sites which are dynamically composed in the Internet user's computer from several sources.

It will also be generic enough to be able to extract any information (present in any site on the Internet), allowing the selection of information to be extracted in an interactive way on an example page, thus making the construction process of the extractions straight forward and

intuitive, allowing its usage by any user, even if the person hasn't any computer training. Thus, the number of people as well as the time for the extraction solution is substantially reduced, and the quality of extracted data is significantly improved, being this task fully implemented and monitored by experts in the data domain.

Integrated with this extraction system, a very comprehensive, flexible and precise web page extraction language, a generic Data Delivery engine as well as the language that allows the deliveries definition, and a complete and adaptable Validation system which can be used on other data processing systems were also created.

This project extends from the earlier mentioned system by Ricardo Raminhos, thus continuing and allowing the same operating mode, on transformation and preparation for data delivery. Thereby ending with a powerful tool for extracting, capable of being used in semi-structured text files as in web pages.

Further, a survey on web pages rendering engines capable of being integrated in a Java application was conducted evaluating their capacity, performance, security and if it is frequently updated.

1.5 Thesis Structure

The content of this thesis is thus divided into seven chapters: Introduction, Related Work, SESS Extraction Detailed Implementation, Architecture, Implementation, Conclusions and Future Work and References

Chapter One Introduction	This chapter presents the motivation, problem description and the context of the thesis.
Chapter Two Related Work	This chapter presents the state of the art in WEB data extraction. Several projects and products in data web extraction are discussed. It also presents the rendering engines available for Java Integration.

Chapter Three
SESS Extraction Detailed
Implementation

As we intended to extend the previous extraction project to the WEB, this chapter describes its most significant implementation details.

Chapter Four
Architecture

This chapter presents the architecture of the WEB Extraction System, explaining the base language that supports the Extraction and Data Delivery systems

Chapter Five
Implementation

This chapter presents the implementation details of the WEB Data Extraction and Data Delivery systems, as well as the graphical editor supporting them.


Chapter Six
Conclusions and Future
Work

This chapter draws final conclusions on the design and implementation of this thesis as well as presents future work activities.


Chapter Seven
References

This chapter comprises all the bibliographic contents referenced throughout the report.

Chapter 2 - Related Work



This chapter presents the state of the art in WEB data extraction. Several projects and products in data web extraction are discussed. It also presents the rendering engines available for Java Integration.



2.1 HTML Data Extraction Projects and Products.....	22
2.2 Interactions on a Rendered Webpage	30
2.3 Rendering Engines.....	34
2.4 Conclusions.....	38

This chapter focuses on the state of the art in the area of WEB data extraction. The first section, presents an analysis of the architecture implemented by some projects for web data extraction. In the second section is included a discussion about interactions on web pages allowed by various types of applications or suggested by organizations, with special relevance to aspects related to selection of graphical elements within a web page. Finishing up the chapter is an analysis of several rendering engines, which can be integrated within a Java application.

It should be noted, that all projects and products for WEB data extraction analysed in this chapter, and all others the author found but are not presented in this manuscript, make their extraction on the content of the source file, and not on the rendered web page, thus not allowing extraction of data on some pages, in particular dynamic pages developed more recently.

2.1 HTML Data Extraction Projects and Products

In this section several Data Extraction projects are analysed and their architecture described. In these particular analysis we are interested in the ideas, functionalities and implementation details of the projects. We try to use them in order to improve the design and development of a more modern and robust solution.

2.1.1 Lixto Web Extraction Platform

Robert Baumgartner, Michal Ceresna, Gerald Ledermüller [16] describe the approach that the company "Lixto Software GmbH" took on building its platform for Internet data extraction. The system covers both, the navigation on WebPages in order to reach those containing the data to extract, and the effective extraction of the requested data.

In the first issue, an approach is proposed based on collecting all actions on WebPages, so that later the navigation on them could be reproduced. Therefore, mouse and keyboard inputs should be recorded, thus allowing access to information protected by password or only accessible using search forms. This information would not be possible to access, by keeping only a sequence of URLs.

In the second issue, Elog is used, which is a proprietary language for data extraction, developed specifically for the platform Lixto. The language operates on Web Objects, which are HTML elements or text, being identified by internal, contextual and range conditions, expressed in XPath. Its rules can be completely specified visually without the knowledge of the language itself.

To allow working on WebPages the FireFox rendering engine on a "Java-to-XPCOM bridge" is used, embedding it in the Java application using the SWT library (which will be further discussed in section 2.3), thereby creating the page rendering and permitting interaction with itself. The possible interactions with the rendered page are based only on the choice of elements to extract through the drawing of rectangles around them. However there is no obligation to choose all the elements to extract, as it allows the selection of a single one of those repeated several times on the page, or having brothers with the same name tag, thus defining a group. Then, all the selected elements inside will be related to it and not to the root of the document, therefore being possible to get several grouped elements found repeated on the page.

It should be noted however, that this platform was developed as a solution for extracting data from a system belonging to the same company, which aims to optimize the business, covering the whole process from the Internet data extraction to the decision-making and subsequent implementation of price strategies. Therefore, data extractions are not generic, being only possible to extract information concerning the marketing of products on sites of electronic marketplaces. It's not equally possible to make any transformation to the extracted data before delivery.

2.1.2 Effective Web Data Extraction with Standard XML Technologies

This project was implemented by IBM, and describes the ANDES system, a platform for the production of applications for WEB data extraction. This system was designed to use only the XML standard technologies, consisting of two parts, the navigation, and the extraction [17].

Navigation in this system is done in an automated way, being only necessary to provide the initial links (called "Seed links") and filters for URLs of pages containing information to be extracted. Then the system starts on the first pages, browsing automatically until it gets all the

valid URL's for the filters mentioned above, sending these pages for the extraction module. To make this self navigation, the crawler needs a list of hyperlinks to travel to. To build such a list, first all links contained on the page are gathered, and then using a special module which by analysing HTML Forms and Javascript code creates "Synthetic Hyperlinks" that are also added to the list. It also allows other settings such as, links to follow, links that should not be followed, and the level of depth to which to navigate.

The extraction of data in this system is made in the "Data Extractor" module, which firstly passes the file by a filter to fix the inadequately formatted HTML, resulting in XHTML. This is then passed through a pipeline of XSLT's, defined by the administrator, which will refine the data gradually until the final XML. The system also allows the use of regular expressions in order to identify or extract some information in the text within the page elements, such as paragraphs of text, for instance in the description of a product, where its various specifications reside. The final XML is then passed to another module called "Data Checker", where the data is checked and validated by means of syntactic and semantic tests. As a last step and finishing the extraction process, the data is stored in a database in the form of already processed results. The actual process of storing the data, is performed by inserting the XML results into a relational database, making the extracted data available to database applications and tools (e.g., for data mining). If an error is detected during any stage of the extraction process, the administrator is alerted and the data where the error occurred is transferred to a "Staging Area" where it may be revised. After the extraction of information on each individual page, the system allows the concatenation of several partial outputs, using an XSLT, to create the XML output, thus resolving the problem of the desired information originally divided into several pages.

To export the already processed data, the system allows only its transformation into comma-separated values, which are then inserted through JDBC into a relational database, however other approaches are under study for data export purposes.

This system doesn't have an automatic generation of wrappers, thus forcing each one of them to be manually written, and according to its authors resulting in a more suitable wrapper to each page, they could even admit the existence of some changes on its own page. Mentioning for example changes of position of the elements to extract, which are the most frequent.

However the manual process requires that the user be also an expert in XML technologies, and particularly XSLT's.

The system also has the ability to schedule extractions, enabling them to be executed in a previously set time by the administrator, as well as to configure all its features through a Web interface, thereby optimizing the use of the system.

2.1.3 WysiWyg Web Wrapper Factory (W4F)

The W4F[18], developed in cooperation with the Department of Computer and Information Science at the University of Pennsylvania and the National Superior School of Telecommunications in Paris, France, is a complete platform for WEB data extraction. The process is composed of 3 stages, first to obtain the web pages, secondly the extraction of the information and finally the third where the data is structured and returned.

This project was initiated with the aim to improve the construction of wrappers so that the developers are mainly focused on:

- Architecture of wrappers by layers – Being divided into three layers, one for each phase of the process under a specific language.
- Declarative specification - So that they become independent of implementation, perceptible, reusable, and easy to maintain.
- Multi-grain extraction – In case of more than one type of extraction, such as extraction of elements within the hierarchy of the HTML file, or the extraction of values separated by commas, being different extraction types and they are considered as complementary.
- Easy to use and reuse.

In this project, none of the created declarative languages are XML based, but instead formed under a more specific syntax similar to a programming language, allowing something similar to methods and variables. Also it should be noted that XML technologies are not used during the whole process.

The process itself starts with the phase of "Retrieval", consisting on getting the web page through a HTTP request to the remote server, like in any browser. This is the phase where all requests, connections, redirections, and permissions, among others, are addressed and resolved. For this phase, a language has been created allowing the definition of the URL, the type of request, and the parameters in case of the "POST" type, thus allowing access to pages that would only be possible to achieve through the filling out of a form, or through the presentation of credentials.

After obtaining the document, the second phase of the process is started with the extraction of information contained in it. Firstly, it is interpreted and a tree object model is created in accordance to the hierarchy in the HTML file, and then the desired information is captured using a specific language called HEL (HTML Extraction Language). This language allows several ways to navigate within the object model in order to reach the elements containing the wanted information. The user can give the full path from the root to the element, following the document hierarchy through the dot separation of the element names, or by following the source of the document using for this purpose the operator "->". In such cases it is possible to make use of variables to select multiple elements within the tree, and not just one. To remove the information from the selected component, the language allows the use of two different types of operators. Those who return the data from the object model, such as "getAttr," and the operators of regular expressions, which are the "split" and "match", which follow the syntax of the Perl language.

This system also allows the structure to be extracted simultaneously with the data in order to group together simple information that otherwise would not make sense. These groups of information are created by the operator "#", allowing multiple paths to be followed within the object model of the document. For instance when extracting information on products in an on-line shop, we need to group the data of each product together, in order to obtain several groups where each one contains information on a single product. In the final extraction stage, the information is returned in a structure designed for that purpose, called NSLs (Nested String Lists), which is defined by NSL = null | string | list-of (NSL). As a model it is very flexible, being automatically created for each extraction according to the use of the operator "#", cited above.

To complete the process, the data within a NSL structure are passed to the last stage, called the Mapping stage. Here is where the data is processed so high-level applications can consume it. At this stage there isn't a declarative language, unlike the other stages, since the data can be automatically converted to Java classes if it is not a very complex structure. Otherwise, the data will be passed on NSLs for Java classes defined by the user. The existing system had an automatic export of data to a external processing system named K2 System [19] but not yet a direct export to XML files, which was still in development.

For this project, there was a great attention in creating editors to generate the wrappers automatically, thus not requiring the user to be an expert in the used technologies and on the languages created for the system, facilitating its use, and allowing almost anyone to use it. The following wizards are included in the system: The wizard for navigation, called "form-wizard", which generates the wrapper code corresponding to the filling in of the forms used to access the pages that contain the required data. The wizard to extract, called "extraction-wizard" which generates the wrapper code for the extraction of data contained on pages. The "mapping-wizard" which generates the wrapper code that allows the conversion of the data contained in the NSLs, to the structures defined by the user. This last wizard was planned but not yet implemented by the date of the publication of this paper .

In this system were also included some tools which receive the wrappers code generated by earlier wizards, and create Java classes so they can be performed as independent system applications, or classes that can be called from another Java application.

This is a fairly complete project, despite of not making use of existing standard technologies and having been fully and specifically created for it. This aspect has caused several problems. Among them is the difficulty of updating and maintaining the software, and the need of having to acquire the indispensable technical training for the staff on the specifically created languages for the project. Moreover the increased workload, resulting from having to implement everything from scratch, as opposed to using standard technology such as XML, XPATH, XSLT, which already have their interpreters implemented, adjusted and properly tested.

The graphic interactions with pages permitted by wizards, are simple, easy to use and understandable by the user. In recent years these interactions became the most common in

extractors applications and wrapper developers. They are however too simple to be able to generate robust extracting expressions resistant to changes or adjustments in the pages, and can only generate absolute terms by the name of the item and its position linked with the above hierarchically. Optimizing expressions requires user intervention on them, forcing the person to have a good knowledge of the language used for extraction.

2.1.4 Data Extraction Products on the Web

In recent years there has been a significant increase in the volume of information that companies need. To meet this demand, they turned to the web as one of the largest repositories of information, which led to a major expansion of the companies providing such services and products. So we analysed the most popular on the Internet, though based only on the features and functionality advertised by its creators, since they are projects of private companies and their architectures are not revealed.

Several companies are building specific extractors on customer's request for pages holding the information they would like. Such as "Ficstar" [20], which has an extractor of WEB information called "Ficstar Web Grabber", that is configurable and adaptable to most web pages, but they also can build specific extractors, or apply a new configuration on its own extractor, so that customers can extract the information they want from the WEB.

Other companies offer their own data extraction services, requiring only specifications from their customers about what information they want, the pages where it's located, and how they wish to integrate within their databases. So, what is done is an effective extraction and data transformation with further loading into customer's databases. We may quote as example the companies "KnowleSys Software"[21] and "Fetch Technologies"[22].

There are yet several generic extractors that allow the extraction of data from most web pages, with just some tweaking, or the creation of a script to perform the extraction. Some of them are available free on the Internet, others are sold by software companies. All described here, don't create any wrappers or extractors, as their only purpose is the extraction of specific types of information out of webpages by a single user.

2.1.4.1 Link Web Extractor 2.5

The “Link Web Extractor”[23] was a tool built in Brazil, by “LinkWS”[24] in order to extract predefined information from webpages, such as E-mails, Phones, Faxes, URLs, Company names, Page titles, Meta Descriptions and Meta Keywords. It can extract information from any page that the user wishes, however it was specially designed to extract information from directory pages of the better known search engines, like Google and Yahoo.

After the extraction process, this tool allows the results to be completely or partially exported to text files, XML or CSV (Comma Separated Values). Yet allowing those results to be managed within the program. For that purpose it offers several features such as advanced searches, filtering, massive replacements and specific deletions, thus facilitating their visualization and maintenance.

2.1.4.2 HTML Extractor 1.0

This extractor [25], was built in order to automatically get certain elements of a large set of HTML documents.

Its most developed function is the web crawling, letting this go and get the HTML pages of many forms, as well as the filtering of which ones should or shouldn't be accessed, allowing this access to take place locally or remotely.

Its extraction features are somewhat limited for the user to specify what he wants in term of extraction, as the system only presents the graphic tree of the document object model. Thus making it only possible to create an extraction of complete HTML elements, and to only allow the use of absolute expressions.

2.1.4.3 Iconico Data Extractor

This extractor [26] was developed and is currently sold by the company "Iconico". It is more advanced than the last studied extractor, but continues to be directed to a single user. The interface is divided into three areas: "where to extract", "what to extract" and "extraction results". The first two relate to the extraction settings, and the third refers to the area where the results of the extraction are placed.

When specifying the pages to be processed, it allows the setting of a list of pages or the directory where they are, or still yet to define the use of the displayed pages in Internet Explorer.

The extraction settings are created through a proprietary interface, containing all the available options, which makes it easy to use, yet it doesn't present the viewing of a sample page from which they could create the necessary settings for this process. Nevertheless it's still possible to extract any kind of information from a webpage, being it of a predefined type as URLs and E-mails, or other specified information through wildcards or regular expressions, or even through the execution of JavaScript code over the webpage.

It is always necessary that the user has a good knowledge on the language and of the rules for extraction, because he will be the one who is going to define them.

At the end of the extraction, the results can be saved to disk, exported to Microsoft Excel, or printed. The application also allows these data to be refined, allowing the removal of duplicates and sorting it alphabetically.

2.2 Interactions on a Rendered Webpage

Data extraction is an essential activity in many different areas, and is usually performed by users with knowledge in the data domain, which mostly doesn't have sufficient understanding of computer technology. This makes data extracting a very difficult process, when it is necessary to define the extraction on a system's specific language, or even on a computer standard language, requiring that the user be trained, or that a computer technician be contacted. A graphical interface capable of interacting with a user without computer knowledge is required, for hiding the extraction language and all other technical details from the user. Furthermore, as the user selects the elements to be extracted on a sample page, the program should create robust extractions that are resilient to small changes in web pages.

In the next subsections we discuss some examples of graphical interactions.

2.2.1 iPhone

The new phone from Apple, the iPhone, holds more capabilities than just single phone calls. Among other important characteristics for its marketing, is the viewing of webpages. This view is achieved through a browser based on the safari specifically developed for the iPhone. Being very specific, it contains some interesting features that are not present in other browsers. What interests us most is the ability to zoom on parts of the page, which is shown in the figure 10.



Figure 10: Webpage Zoom Feature, present on the iPhone

What this interaction allows us to do is the selection of an element within a page with a simple tap in its interior, which is automatically analysed determining which is the element that you want to view, and making the corresponding expansion as a result. For example, if the user pressed on a cell of a table, most probably wants to view the entire table and not only the cell, or when the user chooses an image to enlarge containing associated text, the expansion should be done on the image with the text and not just the image.

Thus, the chosen element to be expanded is determined through the dimensions of each element, its position in the hierarchy of the document, and the context where it is inserted.

This interaction permits the viewing of the page in small parts that are automatically determined by the browser.

2.2.2 HTML Editors

There are many HTML Editors, some of them free, some commercial. Among several editor types we are just interested for this case in the so-called WYSIWYG ("What You See Is What You Get"), such as "Macromedia Dreamweaver", "CoffeeCup HTML Editor" or "Microsoft FrontPage". These allow the construction of HTML pages only with gestures and interactions on a graphical canvas, as well as inserting and deleting elements or modify their properties. These actions are carried out after selecting an element within the page. This simple interaction consists only in clicking on the element that we intend to select, so that the application will display it in a different colour to show that has been selected. Thus it is possible that a user who is not a specialist, can easily build webpages without having to engage with the code needed to produce them. Even though this interaction wasn't created with the intend to graphically select a DOM webpage element (since the HTML code generated by these editors is built from the webpage graphical design present in the editor's canvas component and any changes in it causes the fully re-built of html code), it could be used with this purpose creating a link between the webpage DOM elements and their graphical representation.

2.2.3 Proposed Interactions by W3C

The World Wide Web Consortium (W3C) also presented a proposal for interaction with the web pages, which is described in the section "Ranges" of the proposal for the second level of the document object model [27]. That interaction was designed to be able to select an area within the page using the hierarchy of elements within it. This is based on a selected area within the page, where two points are created ("boundary-points"), the first indicates the beginning of the selection and the second indicates the end. Each of these two points contains the information that corresponds to the node and the offset of the beginning or end of the selection, being the information indicated by the starting point included in the selection but excluded in the ending point. The node can point to a text element or a non-text element, where in both cases the offset indicates a zero-based index on an array. However if it is a text element the array is composed by the text contained in the element, whereas if it is not a text

element the array is composed by the child elements. The set of two points creates a selection range. In figure 11 there are four different range examples. The boundary-points of each Range are labelled with *s#* (the start of the Range) and *e#* (the end of the Range), where # is the number of the Range. For example, in the Range 2 the selection starts in the "BODY" element and immediately after the "H1" element (child indexed by "0") and immediately before the "P" element (child indexed by "1"), and ends in the "BODY" element immediately after the "P" element. This is an example of a selection of the whole element, however it is also possible to select partial elements like in the Range 1, which starts right before the selection in the third character (index "2" of the array) of the "H1" element and ends right before the second character (index "1" of the array) of the "P" element.

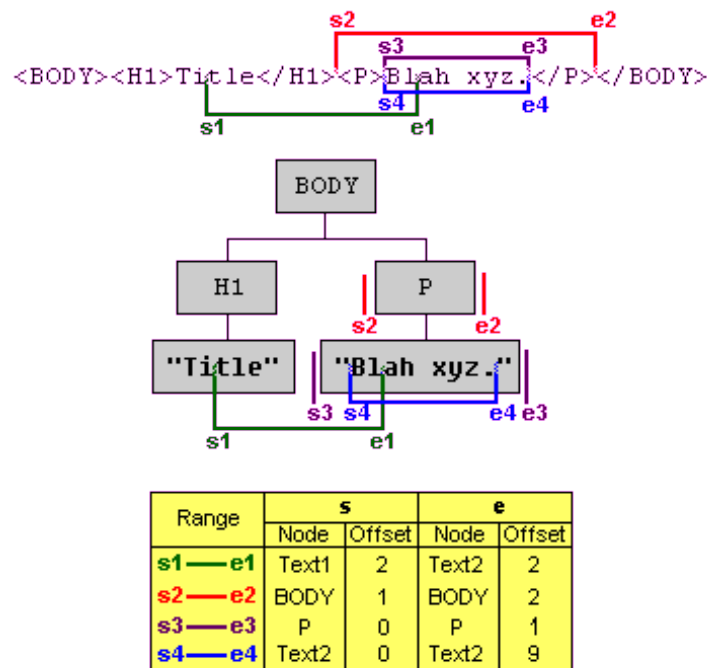


Figure 11: Range selection example [27]

As shown, we can select more than just text contained in a single element, we can even select more elements in order to retain the text format, or include images and other objects such as Flash presentations, provided that the application that creates the rendering of the document supports it.

This representation of the selection also makes it relatively easy to build XPointer

expressions or XPath expressions using the information contained in the boundary-points that make up the range of that selection. However this interaction is only capable of selecting data to be extracted accordingly to its position on the page, not being flexible enough, or even robust.

2.3 Rendering Engines

In order to provide the user with a sample webpage where he may select the items to be extracted, thus automatically creating a script for extraction, it becomes necessary the use of a rendering engine. Furthermore as this project intends to extend the Ricardo Raminhos project, which was implemented in JAVA, a research has been made about the use of rendering engines within a JAVA application.

2.3.1 JREx

The JREx [28] is a project developed by CNMeppada, which creates a new JAVA component for an Internet browser. This was implemented as a wrapper around the "Gecko" engine built by the "Mozilla Foundation", providing an API to work in Java with this engine, which is fully implemented in C++.

The first version was launched on February 2004 and the most recent on January 2006. Since then there hasn't been further developments on this project, where it can be quickly noticed some flaws derived from this lack of updates. This software is only supported up to version 1.5 of Java, and it is only compatible with the version 1.7 of the engine "Gecko", which is no longer used by any other commercial software since it doesn't have the latest security updates.

After a brief search on the Internet, it is notorious the consensus about the installation difficulties of this component. One of the reasons is that their extensive documentation, actually contains little useful information. This greatly hinders the work of the programmer who has to seek information given by other programmers outside the project. The availability for clarification is minimal, since the project has been implemented by only one person now evolved in other projects. However, to increase the functionality of a Java application adding

the ability to view Web content, is one of the most used solutions since the rendering engine is well known, reliable, with considerable stability, and supports a large number of standards.

To evaluate the solution, an example application was created using the JREx. The prototype has revealed a very good performance while rendering web pages, supporting all standard technologies such as JavaScript, Flash, CSS, and so forth, not having experienced any problems while using the rendering engine to view Web Pages. To test the access to the internal document object model, some calls were made to simple functions that the API offers, and it was then when some problems arose. It was found that when more than one function is called or the same function more than once, the results were not consistent, and with too many calls to its methods the component stops working.

Thus, although a very good rendering engine is being used, there are still serious flaws in the wrapper preventing its use, especially when it becomes necessary to access the internal document object model.

2.3.2 Lobo: Java Web Browser

The Lobo: Java Web Browser [29] is a project for a Web browser component, developed entirely in Java. This architecture has many advantages in the integration of this component into Java applications, becoming a lot easier and with a much greater extensibility. It is also safer since Java is less prone to buffer overflow attacks, as it runs inside a virtual machine and is continuously within a closed environment, making it secure for the user. It also has the portability advantage, since there is a Java Virtual Machine for almost all operating systems, thus allowing it to be executed on any of them.

This project uses the rendering engine "Cobra" implemented by the same group, being the performance of this engine quite acceptable. It also contains an extensive API providing access to its internal document object model. The project is yet in active development, having a very good and easy to use documentation. This component has been submitted to tests not revealing any errors or problems when their internal methods, provided by the API, were called. The rendering engine is not in its final version and didn't create a correct view of all the tested pages, particularly those using "Ajax" as the "Gmail" web site.

This solution has one last advantage. Since the whole project was built as modules, it is possible to make the interpretation of the HTML file, execute any JavaScript and apply all CSS styles contained on the page, resulting in the document object model for that page without creating its graphical representation. This makes the automatic extraction process faster, because it has fewer tasks to perform, thus using less memory.

2.3.3 Internet Explorer trough Jawin Lib

The project Jawin is a facilitator of interconnection among native Win32 applications XPCOM virtual machine and Java, achieving this through the use of JNI. The aim was to use this project to provide an interface to manipulate the Internet Explorer rendering engine, as well as showing the rendered webpages in a Java application. However, it wasn't possible to meet the proposed objective by ourselves, so in a search for more information in order to overcome the current experienced difficulties some developers³ working with this library offered assistance. With their help, an example application that enabled us to perform an operation test was created. This showed a good page rendering, easily supporting the execution of Javascript within the page. However, it is not possible to integrate it graphically into a Java application interface, as it must reside on its own separate window.

2.3.4 Apple Webkit

The Apple Webkit is an open source rendering engine mostly created by Apple developers. It is well known for being used in Apple's browser "Safari", which comes along with their operating system, being advertised as the fastest browser in the market. This engine appears to be a good solution because of its open source code, therefore portable to any operating system. But it hasn't yet been officially put into operation in a Java environment. There is an unofficial integration of this render for Java called "WebKit & Java integration" [30], however it requires to be specifically executed in the operating system "Mac OS X", thus making the advantage of being portable to multiple operating systems invalid. Given its so hard integration, it was not possible to test this rendering engine.

3 We take this opportunity to thank the kindness and willingness of Mr. António Cruz and Mr. João Carreira from ITDS.

2.3.5 JDIC (JDesktop Integration Components)

The JDIC [31] is a JAVA library with a set of components for desktop, like a “Browser”, a “FileManager”, and others. This library was specifically built to work in several different operating systems. In this particular case it was only necessary to test the browser component. The use of this library is fairly easy, as well as its integration into a Java application, since it uses the rendering engines already on the operating system, like the Internet Explorer engine on Windows or Firefox engine on Linux, making it very fast in the webpages rendering. This component doesn't provide an API giving direct access to the document object model of the web page being rendered, despite of the various rendering engines that it uses permitting it. However it allows Javascript code to be executed on the page that is being viewed, and as Javascript has access to the internal document object model of the web page, it is possible to access the structure even though it is an indirect access. As additional information we can refer that this project is no longer in active development.

2.3.6 SWT Library Browser Component

The Java graphical library SWT, was created by IBM containing a component which can integrate a browser window within a Java graphical interface. The component is also prepared to work in all operating systems that Java supports, using the default rendering engine for the particular OS where the application is running. Being this a very good advantage for applications that just need to show to a user a rendered webpage, it can't be used (just as it is) for our purpose, since we need that the data extraction occurs in the same rendering engine which was used to create the extraction script. However, the component allows to be configured to only use a specific rendering engine, requiring just that it be previously installed. So, we opted to use "Gecko" the Mozilla Layout rendering engine, since it is one of the most secure, has a great performance, it is widely used and can be independently installed.

Although this component doesn't provide direct access to the rendered webpage object model, its API allows the execution of Javascript code within the webpage as well as returning its results, thus enabling an indirect access to the object model of the page. An example

application was created to test the usage of this component, which revealed some discrepancies between its operation and the published specification. These had to be firstly corrected, in order to use the rendering engine properly and be able to execute extractions with it. Once resolved, it worked flawlessly and may now be used in the extraction system.

2.3.7 Rendering Engines Conclusion

Several ways can be used, to display a webpage within a Java application as we have seen. However, in this thesis we needed more than just a simple display of the webpage, it was necessary to allow the user to interact with the webpage when constructing the extraction script as well as performing automatically the actual data extraction. In both cases the execution of specific Javascript code in the rendered webpage by the rendering engine is required. Given this requirement, not all of the above are suitable, and we also have to consider the rendering engine performance, its security and if it has been recently updated. Besides these shown here, it was also considered some others like the "Java-to-XPCom bridge" used on the "Lixto" internet data extraction platform presented in section 2.1.1. These weren't included due to a lack of documentation, which made difficult its usage and integration in this project.

After this study, we have chosen to use the SWT Library Browser Component, since it presents the best characteristics, like its simple integration, the possibility of executing Javascript and supporting the use of Mozilla rendering engine "Gecko" which is very secure, fast rendering, recent and also allows its update without the need of recompiling the whole system.

2.4 Conclusions

Concerning the several studied solutions of data extraction from the Web, it was found that these act directly on HTML files. This aspect is seen today as a serious limitation, since the evolution of the Web is gradually loosing the connection with the HTML source files, making the data extraction impossible on a growing number of webpages.

Generally all extractors have specific favourable and unfavourable features, however none of


the evaluated holds the necessary requirements, thus not meeting the current targets. In these solutions some interesting features were found such as the use of standard languages like XPath and XML in the "ANDES" system, described in section 2.1.2. This approach allows an easier maintenance and extension without the need for the use of technicians with specific training, unlike the remaining which use proprietary extraction languages such as the "Elog" in the company's project "Lixto" or the "HEL" from the project "W4F". Another noticeable feature was the existence of an interface for the construction of extractions in the projects "W4F" and "Lixto". These interfaces significantly simplify the process omitting most of the technical details thus enabling its use by users without experience in computer-science. It should also be noted the ability to validate the extracted data presented in the "ANDES" system by the "Data Checker" module as well as the capability of data transformation through an XSLT pipeline where the data has to move across. All solutions under analysis have a specific process for data delivery and not a generic solution adaptable to different situations, thus requiring that the subsequent procedures be adapted to the extraction system in use.

The analysis of these products was important regarding the contact with various approaches that brought a consolidation in terms of architecture for the construction of a more generic solution and adapted to the WEB developments.


Where the interaction models are concerned, it was found that there is no ideal model for building extractions. However, there is a consensus on the selection of a particular element within a webpage, as in the iPhone and HTML editors. This is only based on pointing at the interior of its graphical representation. Although this is a simple interaction, it may not be sufficiently specific as webpage elements are displayed graphically overlapping each other. Thus, it will be necessary to provide the user with a method to fine-tune the selection, before its confirmation. This adjustment is also used in the HTML editors.

Chapter 3 - SESS Extraction

Detailed Implementation



As we intended to extend the previous extraction project to the WEB, this chapter describes its most significant implementation details.



3.1	Sections.....	42
3.2	Fields.....	43
3.3	Transformation.....	46
3.4	Data Delivery.....	47
3.5	The FFD Editor.....	49
3.6	Conclusion.....	51

This chapter presents the detailed implementation of the previous extraction project developed by Ricardo Raminhos in his master thesis, which was extended in this one. It will mainly focus on the extraction language used, as well as on the most significant details for the implementation of this new extraction system.

Since it is a declarative system, an important point is the extraction language, which determines all of its implementation. This language has been called FFD (File Format Definition), and it is an XML based language, being essentially composed by the following elements, "Sections", "Fields", "Transformations", "DataDeliveries", that we are going to see in detail. This extraction system was developed to handle just semi-structured text files (Figure 12), and not HTML source files.

```

Metadados sobre o Ficheiro :Data_list: 20050430_ace_mag_1m.txt
                             :Created: 2005 May 01 0009 UT
                             # Prepared by the U.S. Dept. of Commerce, NOAA, Space Environment Center.
                             # Please send comments and suggestions to SEC.Welkmaster@noaa.gov
                             #
                             # Magnetometer values are in GSM coordinates.
                             #
                             # Units: Bx, By, Bz, Bt in nT
                             # Units: Latitude degrees +/- 90.0
                             # Units: Longitude degrees 0.0 - 360.0
                             # Status(S): 0 = nominal data, 1 to 8 = bad data record, 9 = no data
                             # Missing data values: -999.9
                             # Source: ACE Satellite - Magnetometer
                             #
                             #
                             #           1-minute averaged Real-time Interplanetary Magnetic Field Values
                             #
                             #
                             #           Modified Seconds
                             #           UT Date   Time   Julian   of the
                             #           YR MO DA   HHMM   Day     Day     S     Bx     By     Bz     Bt     Lat.   Long.
                             #           -----
                             #           2005 04 30 0000 53490   0     0     8.5  -3.1  -9.9  13.4  -47.5  339.9
                             #           2005 04 30 0001 53490   60    0     8.5  -1.8 -10.3  13.5  -50.0  347.9
                             #           2005 04 30 0002 53490  120    0     7.7  -0.9 -10.9  13.4  -54.4  353.2
                             #           2005 04 30 0003 53490  180    0     6.7   0.5 -11.6  13.4  -60.0   3.9
                             #           2005 04 30 0004 53490  240    0     7.6  -3.3  -9.4  12.6  -48.5  336.5
                             #           2005 04 30 0005 53490  300    0     7.4  -0.5 -10.3  12.7  -54.1  356.3
                             #           2005 04 30 0006 53490  360    0     7.7  -0.9 -10.1  12.7  -52.3  353.3
                             #           2005 04 30 0007 53490  420    0     8.0  -1.2  -9.7  12.6  -50.3  351.5
                             #           2005 04 30 0008 53490  480    0     8.1  -2.0  -9.4  12.6  -48.3  346.2
                             #           2005 04 30 0009 53490  540    0     8.5  -1.9  -9.0  12.6  -46.0  347.4
                             #           2005 04 30 0010 53490  600    0     8.7  -2.2  -8.9  12.6  -44.8  346.1

```

Metadados sobre os Dados

Dados

Figure 12: Sample of a Semi-Structured Text File

3.1 Sections

The first step for defining an Extraction process is classifying the sections in the input file, considering for that purpose two kinds of possible sections: contiguous and delimited. Every section is then composed by a set of lines that appear together in sequence in the input file. In the contiguous section type, that set of lines are defined as a set of adjacent lines that share a

common pattern condition, where that pattern is just represented through a regular expression. In the delimited section type (Figure 13), the set of lines is defined by a set of enclosed lines between a start and stop delimiter.

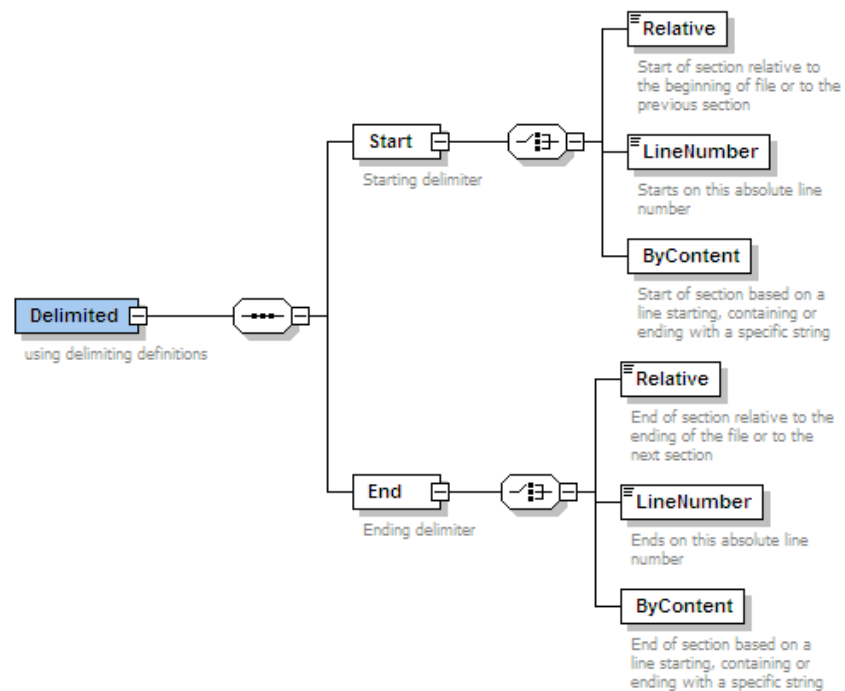


Figure 13: Section XMLSchema

3.2 Fields

Defining the extraction fields in each section will be the next step. All of these extraction fields are defined in a separated element from the sections called “Fields”, and linked to a section by an inner element called “SectionIndex”. The fields are also characterized within the system by a unique Name, and an offset called “starting line” that indicates the first line of the section text from where the data extraction is considered. These are divided in two types of extraction fields, the “SingleValue” field and the “Table” field.

The Single Value field extracts only one value data (e.g. the author’s name, a timestamp), and can be defined in two distinct ways:

by delimiters: Where the user specifies prefix and suffix strings, and the text in between is extracted.

by regular expression: Where the user specifies regular expression capturing the text of the single field.

The Figure 14 shows the XMLSchema for the definition of the Single Value.

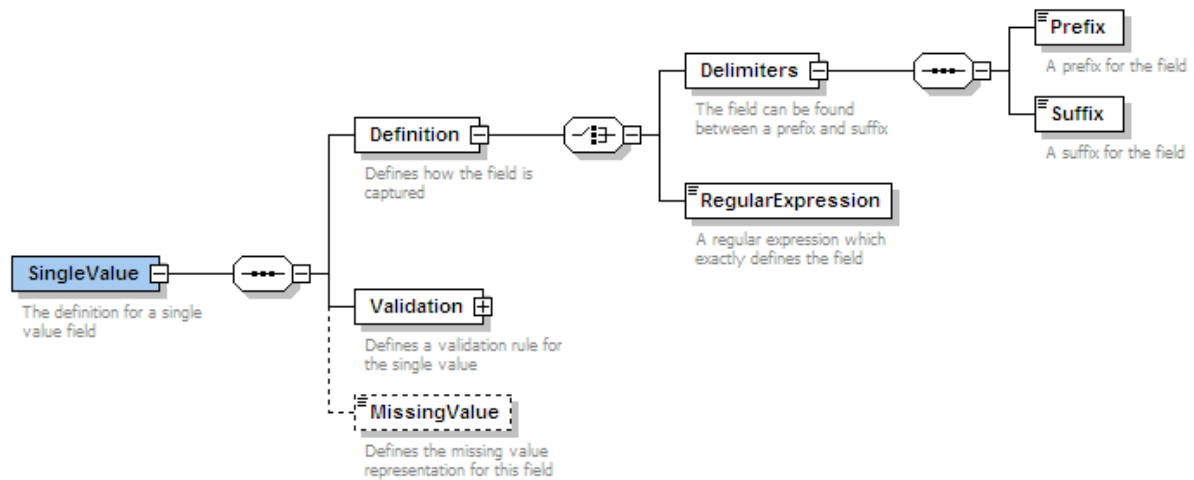


Figure 14: Single Value XMLSchema

The Table field, extracts a bi-dimensional matrix thus capturing multiple columns of data (e.g. the temperature for all European cities for all the days of a given month), and can be defined in three different ways:

by Fixed Width: Where the user specifies a set of column breaks (numeric positions in the text) that specify boundaries for each column.

by Char Delimiters: Where the user specifies a delimiter character that creates the boundaries for each column.

by Regular Expression: Where the user specifies a regular expression that captures all table columns. Each group in the regular expression is mapped into a table column.

The XMLSchema for the definition of the Table field is shown in figure 15.

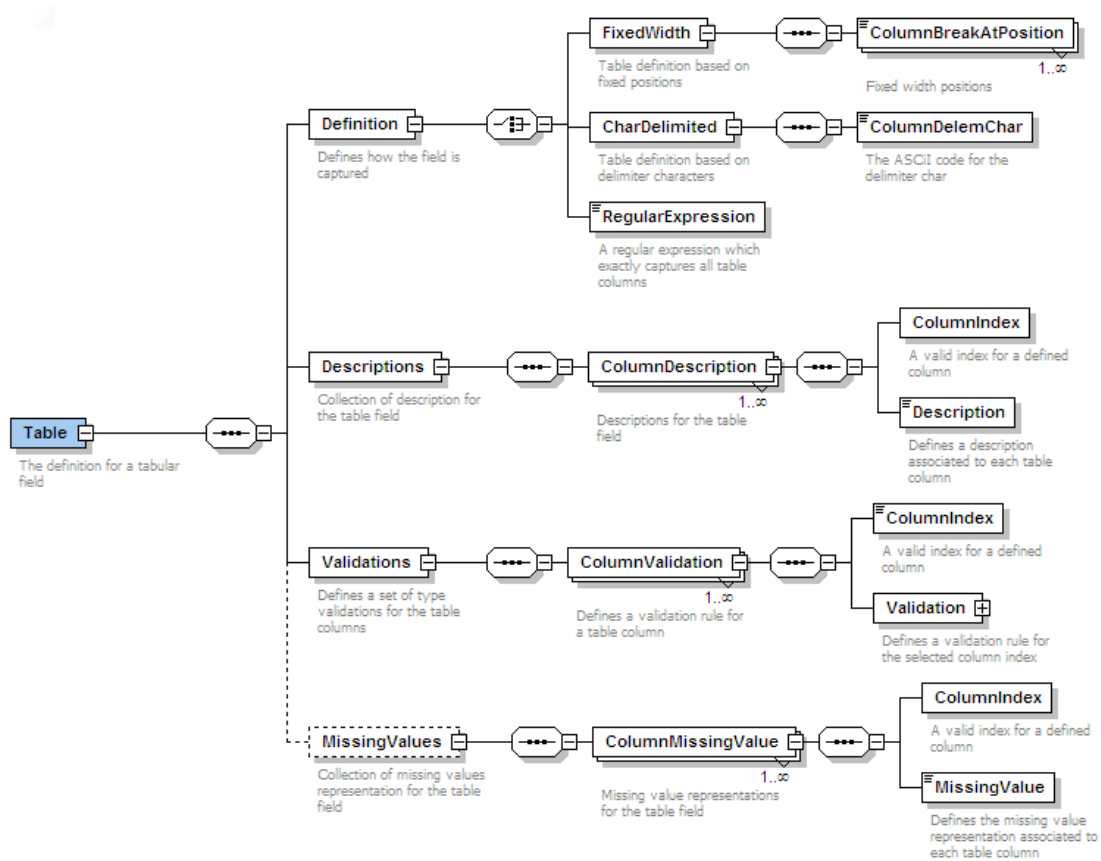


Figure 15: Table XMLSchema

The system also allows the user to control the way the missing values are processed during the extraction. In order to achieve that, each extraction field has an element called "MissingValues", where it stores the information relative to whether it allows absent values in the extracted data, and a custom string to replace the missing values text. (e.g. The empty string).

Both table and single value fields share the same data typing and validation scheme. In table field, the data typing and validations are done separately for every column defined. There are four data types available and each one has its own set of validation rules. The Figure 16 illustrates the four data types with their corresponding validation rules.

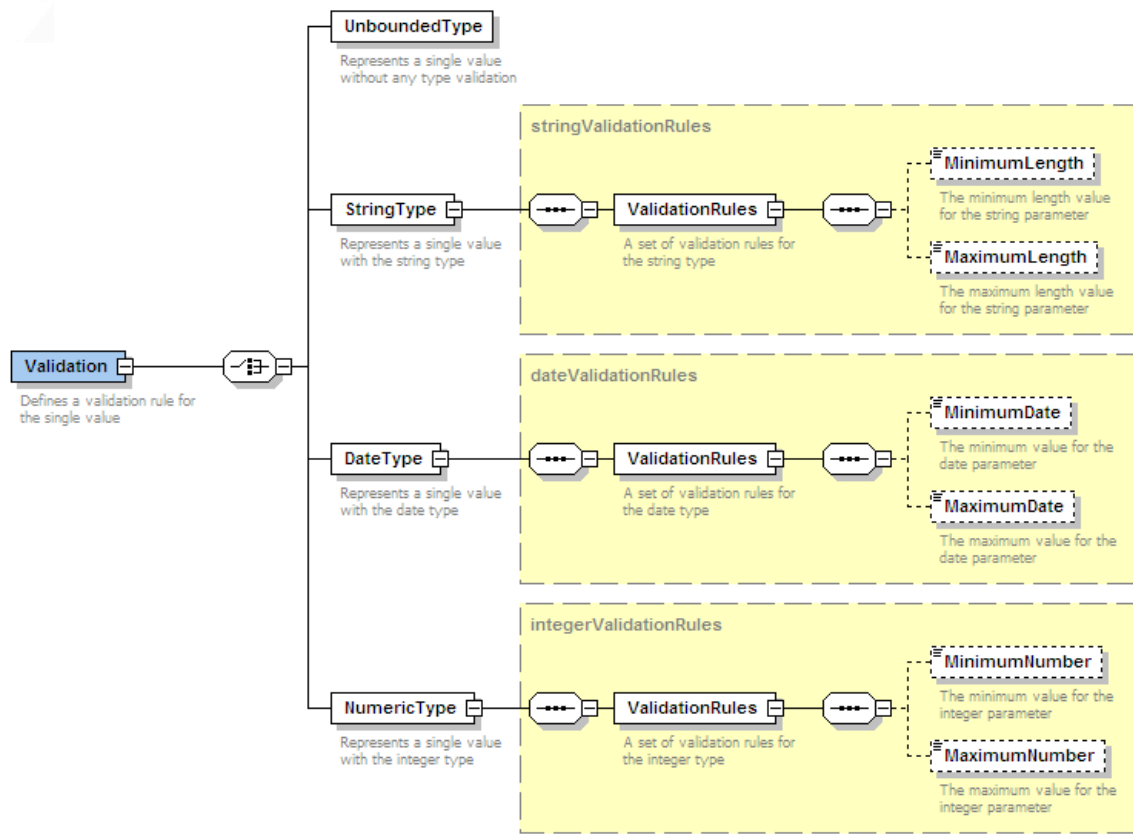


Figure 16: Data Typing and Validation Rules XMLSchema

3.3 Transformation

Once concluded the field definition step, the “Extraction” activity should be considered complete and the field outputs could then be used in the subsequent “Transformation” processes.

This step prepares the data to be delivered upon transformation since it may not always be ready for a direct data delivery. To accomplish that, the user can specify multiple transformations that will be applied to the data, through the creation of a pipeline of transformations where the outputs of one are fed into the inputs of the next. The transformations were implemented as plug-ins, so that they could be easily added or removed thus making their code independent from the system, and allowing a more flexible and adjustable way to fit the data nature and current transformation needs.

Once each transformation is defined, the system only needs to keep its logical part consisting

on the type of transformation, an identifier, input and output specifications, and the specific parameters of the transformation type according to the presented schema in the figure 17.

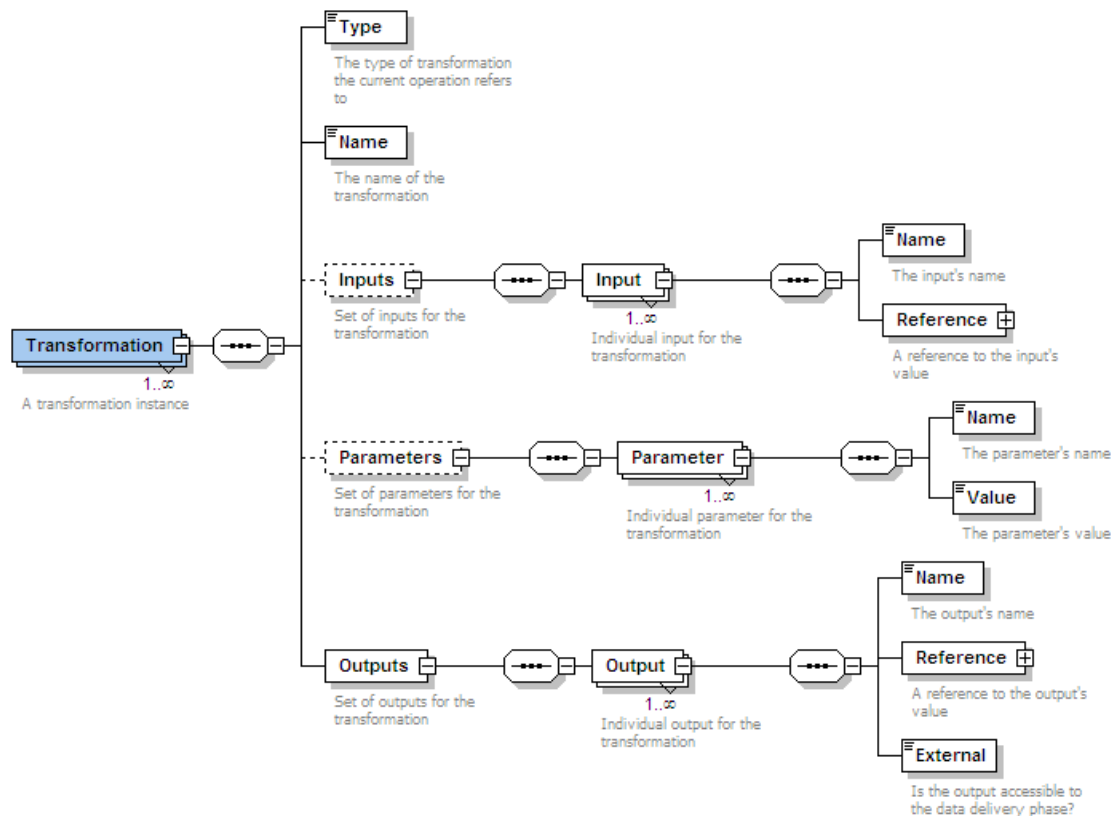


Figure 17: Transformation XMLSchema

3.4 Data Delivery

The extraction process ends with the delivery of the extracted and transformed data. It is in this step that the structure of the delivered data is defined along with the identification of relevant information for the delivery. The system allows the definition of several data deliveries per extraction process containing each one of them, an Identifier, a template and the actual processed data as shown in the figure 18.

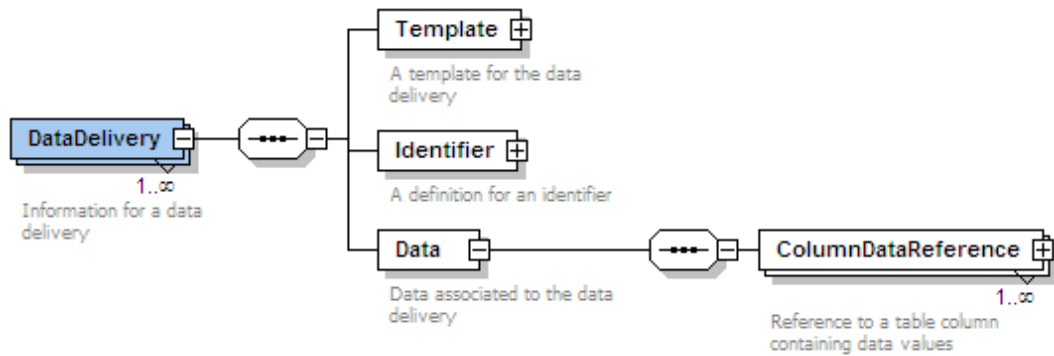


Figure 18: Data Delivery XMLSchema

The referred template contains the structural model in terms of how the data should be formatted, and how the external application is expected to receive it. These templates have been implemented in development time for the SESS project, and are intrinsically linked to the data domain through specific elements in the XMLSchema language that is used for defining templates in the project. The specifications include the template name, its group and delivery columns with name and type, as per the figure 19.

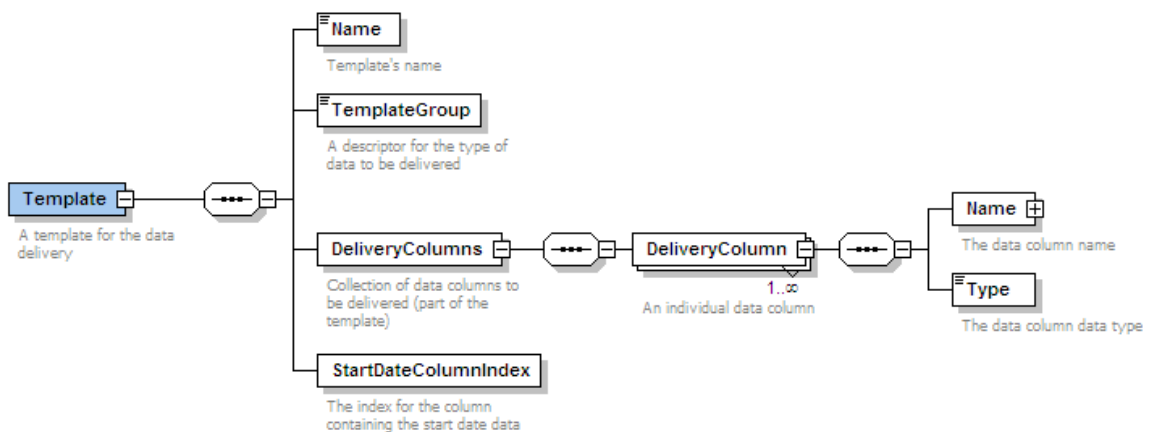


Figure 19: Template XMLSchema

The identifier is a value that is applied on each row of the data delivery, establishing in this way a relation between the present value in the record and the global identifier for that

parameter. This identifier is extremely relevant for the SESS project, having been specifically implemented in this extraction system.

The delivery data is defined by a group of references to column names (being the extracted fields or the transformation outputs), there for forming a row oriented data delivery at process time.

3.5 The FFD Editor

The representation of the extraction process on a standard XML Language is not within the reach of every domain users, so this has motivated the need for a graphical interface enabling the construction and edition of FFD metadata files, which led to the creation of the FFD Editor. This editor doesn't work directly over the FFD declaration but instead over a semi-structured sample text file, representative of a class of files. Through the live edition over data, based on a set of templates, wizards and graphical gestures performed by the user, the FFD specification is created. The editor interface is divided in three tabs, one for each step of the whole extraction process (extraction, transformation and data delivery), as shown in figure 20.

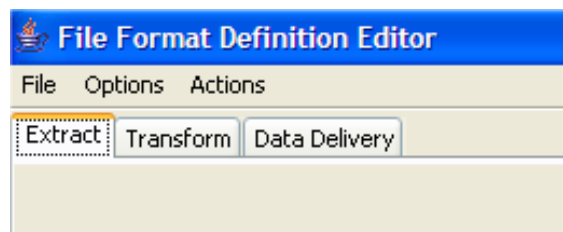


Figure 20: The FFD Editor Tabs

These three steps are correlated and work as a pipeline, where in the “Transformation” step can be used “Extraction” outputs, and in the “Data Delivery” step can be used the “Extraction” as well as the “Transformation” outputs.

3.5.1 FFD Extract Tab

It is in the extraction tab (Figure 21) where the user selects which values he wants to extract.

There are three buttons from where the operation type can be selected (to create a new section, a new single value or a new table), and all the other gestures and selections are made upon a sample text file. The extraction definition starts by creating some sections, which is done by graphically selecting the lines that will compose the new section. The editor also allows that the section be defined by a set of lines starting with a specific group of characters, or that the section be started relatively to where the other section ends or even ending at the end of the file.

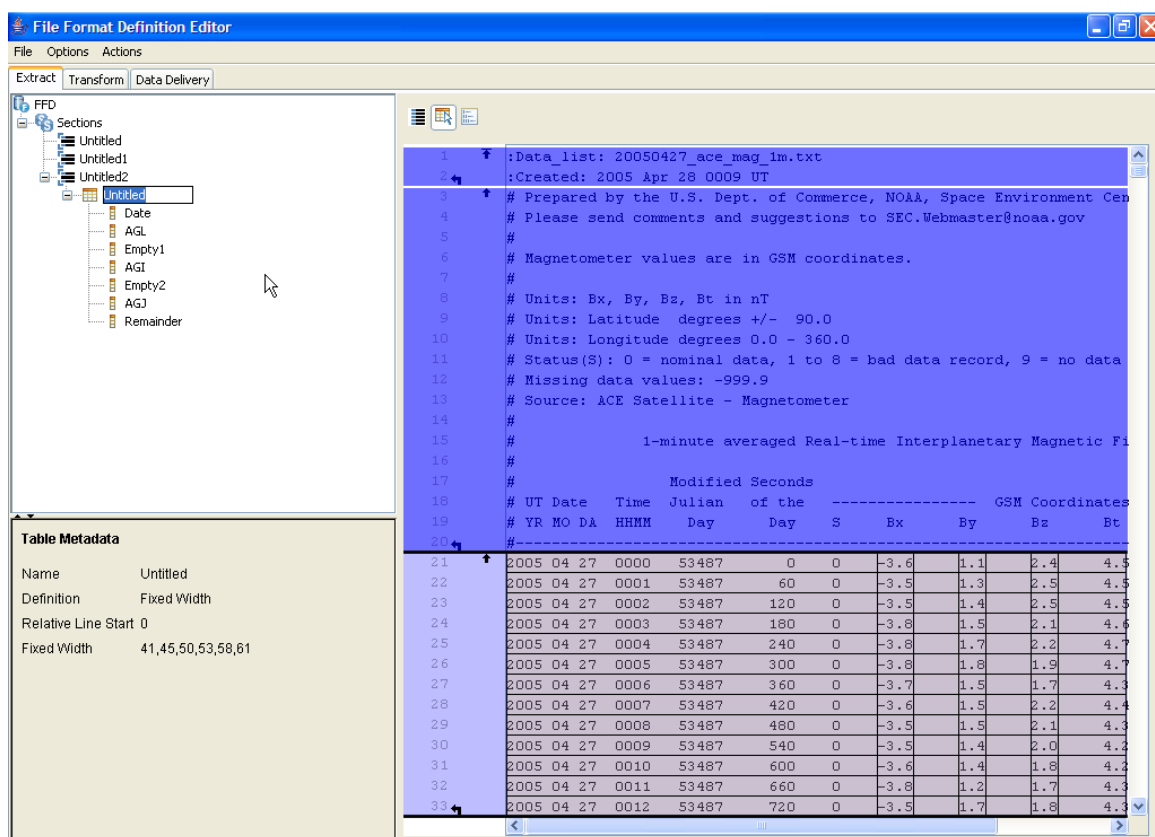


Figure 21: Extraction Tab ScreenShot

After creating the sections it is possible to create the extraction elements, like the single value or the table. The creation of these elements is started by selecting the line in case of the single value, or lines in case of the table, where the information is, and then answering a wizard. Here is where the element can be defined, how it will be extracted, in the case of the table how many columns does it have and how the values are separated, selected the data type with

the corresponding rules of validation for each column and how the missing values are processed.

The editor also shows a graphical representation of the sections, single values and tables on the sample text file, making it easy for the user to view the defined elements to extract.

3.5.2 FFD Transform Tab

In the transformation tab the user can choose which transformations the data has to pass so that it can be ready for delivery. This is done by creating a flow chart with pipelines of transformations, from the values outputted by the extraction to the values outputted from the transformations that are marked as external. The possible transformations the user can choose in this editor are the same that the extraction system has, so if a new transformation is added to the system it can also be added to the editor.

3.5.3 FFD Data Delivery Tab

The Data Delivery tab is where the user defines the several delivery instructions needed for that particular extraction. For each delivery it is necessary to choose a template from a list of previously created templates. The referred templates, as explained above, describe the delivery format including the metadata and the data columns names and their types. With a chosen template, the editor allows the user to map the values outputted by the extraction and transformation to the metadata and data columns specified by the template. However the delivery templates can't be defined through this system nor through the editor, so they have to be manually created outside.

3.6 Conclusion

This system was well succeeded and in result been included in SESS project as the main extraction system, since in this project the input data is in the form of semi-structured text files gathered from the measurements of the satellites sensors. This project was developed in order to replace the previous system, which had revealed poor performance when dealing


with the input files, as well as reduced flexibility and capabilities. This extraction system has a great performance importing and processing the input files, making it possible to process much more satellite data, more quickly, and with fewer computer resources. It is also very flexible, having a very simple graphical interface to define extraction processes, making it also possible to create extractions from more data sources and update the ones where the format has changed.

However as this system only supports one type of data source (semi-structured text files), it is not suitable to extract information from web pages. This is the main reason why in the majority of the cases the extraction system based on lines doesn't work directly on webpages, as they are organized by tags instead of lines, as well as being its rendering different from its source visualization.


Since this project was implemented specifically for the SESS Project, it has some particular features imposed by the data nature (which was going to be processed by the system), as well as by external systems receiving this system exported data. These particular features, makes it a non-generic system unable to be used in other different situations or projects. Some examples of these are:

- The values that must accompany every extraction determining its type ("Ad-Hoc", "Summary", "Realtime"), which are irrelevant to the extraction process being only used by external programs.
- The delivery templates intrinsically linked to the extracted data nature.
- The data delivery identifier, and global identifiers.

Chapter 4 - Architecture



This chapter presents the architecture of the WEB Extraction System, explaining the base language that supports the Extraction and Data Delivery systems



4.1	Extraction System Overview.....	54
4.2	Extraction Module.....	56
4.3	Data Delivery Module.....	66

This chapter is focused on the architecture, explaining the several processes and the specifically developed language that supports the webpage data extractor as well as the available options and the decisions based thereon.

4.1 Extraction System Overview

To develop this web extraction system, we started by constructing the system extraction core. In the core is everything needed to automatically perform the entire extraction, being supported by an XML based language called FFD (File Format Definition). Furthermore, the core was divided into modules so it will be easier to maintain when the system is in production, or to extend it in order to add new features and capabilities to the system. It also makes the system easier to upgrade, since it will only be necessary to change a part of the system instead of the whole system. Developing new features around the core system or integrating it into other systems is also simpler. Basically the system's core is composed by a set of three modules (Figure 22) that in conjunction perform every extraction procedure.

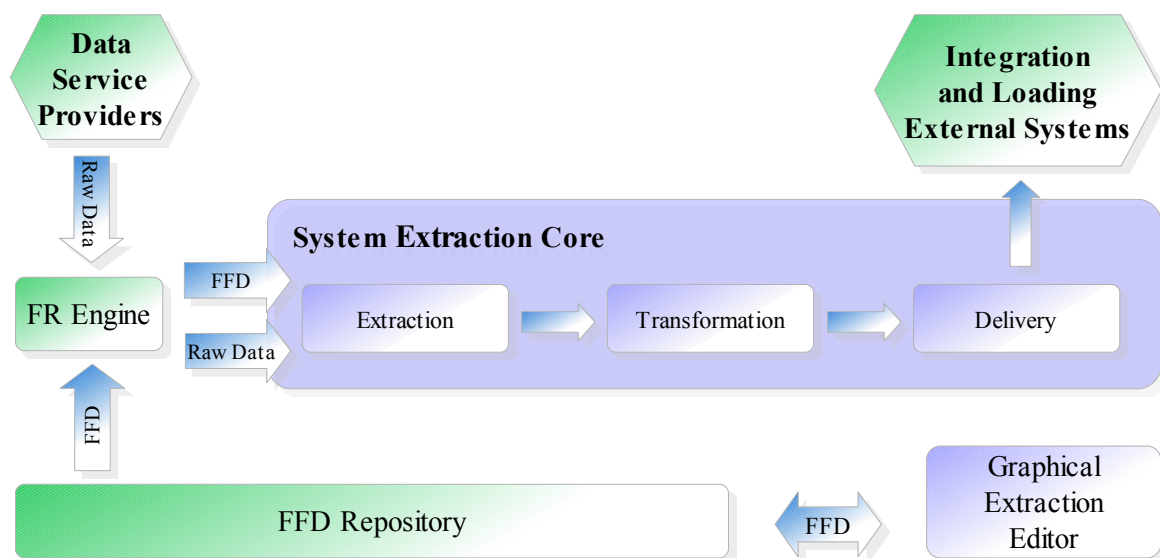


Figure 22: Architecture Basic Structure

The "Data Service Providers" and the "Integration and Loading External System" represent the external elements. The "FR Engine" is the module present in the previous system, with the ability to schedule and automatically initiate extractions, however it lacks the ability to navigate through the WEB, allowing only to redirect to a webpage.

The first module is the "Extraction Module". This module is responsible for extracting the information from the data source. It supports two different types of sources, semi-structured text files and webpages. This module also validates the extracted data to guarantee its quality. It will stop the extraction job if the data isn't correct, thus saving time and processing capacity. The extraction module and the XML based language that it supports is explained in detail in the section 4.2.

The second module is the "Transformation Module", which is responsible for the extracted data transformation. For this module it wasn't necessary to create a new language, since the existing one is already generic, not being dependent on the data source type and allowing the use of multiple transformations including customized ones. Hence it was decided to use it instead of creating a new one.

The third module is the "Data Delivery Module". This module is responsible for delivering the extracted and transformed information to external systems. In order to achieve it, this module was designed to be as generic as possible. Therefore it was divided in sub-modules (being each one responsible for a particular type of delivery), making it easy to develop new specific deliveries and include it in the extraction system. This makes the system more adaptable to any situation and use. More details about this module as well as the language that support it are explained in section 4.3.

In order for this modules interchange data, the core system stores the output data of each module in data columns, which are then put together in an hash-table. When a module data processing is started the hash-table is transferred.

There was a great concern about the conception of the extraction system core in terms of allowing the system to be as generic as possible as well as extensible and scalable.

An Editor was also created, to be possible to easily create FFD files. This editor was developed to be very intuitive and easy to manipulate, making its usage possible by any person even if they aren't computer experts. In order to be easy to use, the interactions between the user and the webpage are all "point and click" or filling wizards which allow for the definition of more complex extractions expressions. This way it is easy for any user, and the advanced users also can improve the automatic generated expression, making better usage of the extraction language.

4.2 Extraction Module

In order to extend the system to extract data from a different source type besides semi-structured text files, it was necessary to create a generic mechanism capable of supporting different source types including the pre-existing one inside it. However each different source type has its own particularities requiring a specific set of extraction expressions and rules, hence the mechanism has to support as many different extraction languages as the diversity of source types.

To create such mechanism, a new element called "Extraction" was added which would specify the type of extraction that will be applied holding all the expressions and rules of that extraction language, not allowing the mixture of elements of different languages, also guaranteeing the correctness of the extraction language by simply validating through the XML Schema.

Then the extraction elements needed to process the semi-structured text files have been joined in a single element called "TextFileExtraction", and all the elements needed to process web pages in an element called "WebPageExtraction", allowing the choice of one of these in the "Extraction" making then possible to select only one source type for each extraction, as shown in figure 23.

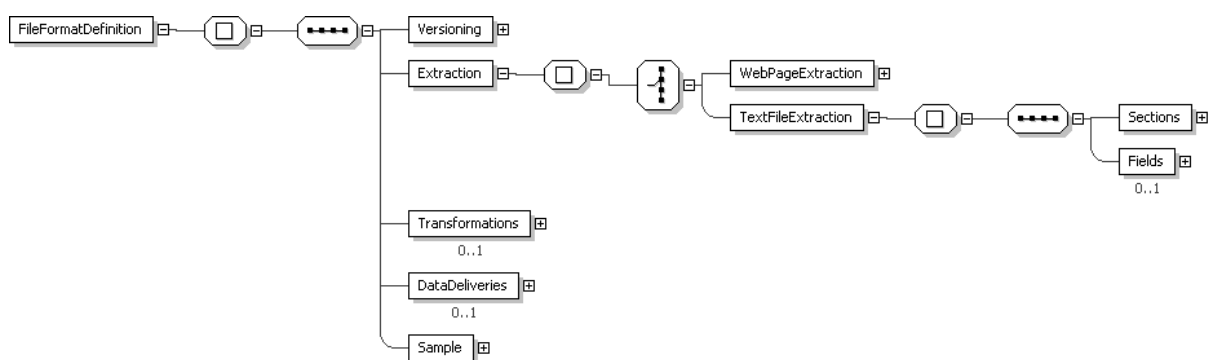


Figure 23: FFD XMLSchema with the extraction languages separated

In case of the webpage source type, it was necessary to create an all new language due to the webpages particularities which made the use of the pre-existent extraction language impossible, as its organization is tag oriented instead of text line oriented. Being hierarchically organized by tags it offers another advantage; it can be used in XPath

expressions to navigate through the document structure, something that the pre-existing language does not support. So a completely new language was created specifically for this source type. This new language for the webpages is divided into sections where each section contains extraction fields, which can then be of three different types, as we will see next in detail.

4.2.1 Sections

In order to decide if sections should be included in this language, it was necessary to examine and evaluate both options, with or without sections, in order to choose the best option.

If this language were created without sections, it would also allow the extraction of every value from a webpage and would be simpler to implement. However, it wouldn't be simpler for the user as it could lead him in error because of the inexistence of content separation on the webpage.

Therefore, including sections in the language, it will make it much easier for a domain user to create an extraction, as he may separate the contents of the page before choosing the extraction values within a section. Thus simplifying the creation of the extraction script, as well as reducing the likelihood of a user committing an error in the script. There are also other advantages in using sections, as they allow simpler and smaller extraction expressions, which are more rapidly evaluated, making the extracting process quicker and a lot more efficient since it uses fewer resources.

So, the taken option was the use of sections in the webpage extraction language, consequently creating the need to define them. To achieve that aim, two possible approaches had to be considered, as the Section being only one DOM Element, or a list of equal DOM Elements all siblings. Being the last a more complex solution both in terms of implementation as well as regarding the interaction with user, thus forcing him to have a more advanced computer training in order to use a quite more complex script creation interface that would then be less intuitive. It would also affect the performance of extraction, since it would be necessary to evaluate the extraction expressions multiple times for each section. Therefore, the chosen option was to use a single DOM Element per Section, as it is simpler to implement, faster when in execution and easier to interact with the domain user. Thus, it was defined in the

extraction language, that each section be represented by an element called "Section" (Figure 24), having subsequently 3 child elements as follows:

Name: An unique name identifying the section Name.

Definition: A XPath expression defining the DOM element that represents the section.

Fields: A set of extraction fields containing all the information needed to perform the data extraction.

Validation: A set of validations rules that determine the extracted data correctness.

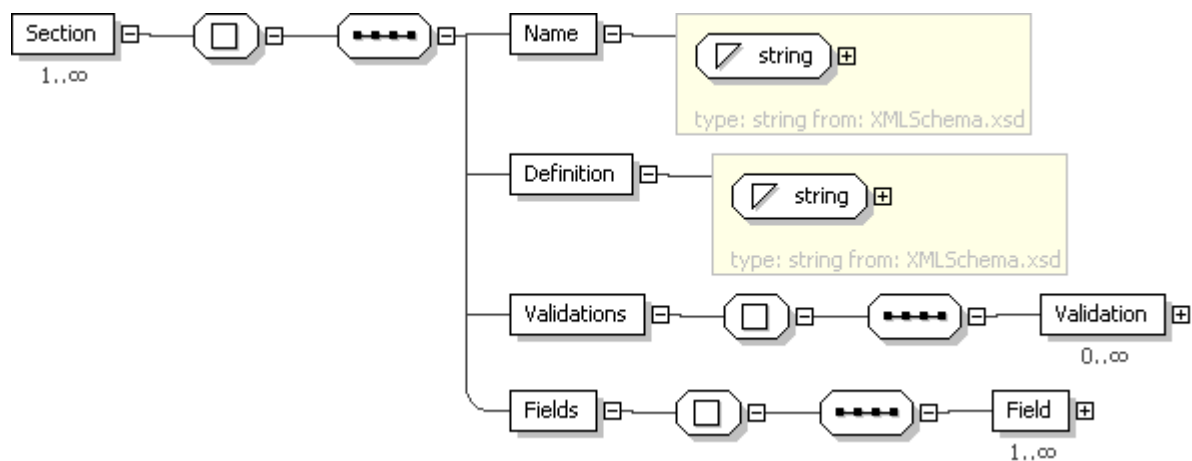


Figure 24: Section XMLSchema

To further improve the extraction language ensuring the correctness and quality of the extracted data, a section validation mechanism was created. This mechanism allows the user to specify several rules that the extracted data in that section has to comply with, in order to enable the system to know if the data is valid and if it can be processed as well as exported, otherwise the extraction is halted.

The rules in this mechanism are supported in the data extracted by the section fields, allowing the comparison of those values to each other, therefore enhancing the possibilities of data validation. The data validation of each individual extraction field is done using a different set of validation rules directly included in it, as it will be explained in the section 4.2.2. The Section Validation Rules were developed to enable the comparison of two data columns, being the results of the extraction fields one or more data columns. The Section Validation

Rule definition (Figure 25) is composed by the "Name" element with the rule's name, the "FirstExtractionField" and "SecondExtractionField" elements corresponding to the data columns being validated, and an element called "Operation" specifying the comparison operator. It is also possible to validate the number of values present in the data columns instead of the actual values, being only necessary to add the optional element called "CountLines" to one or both of the fields in the rule definition.

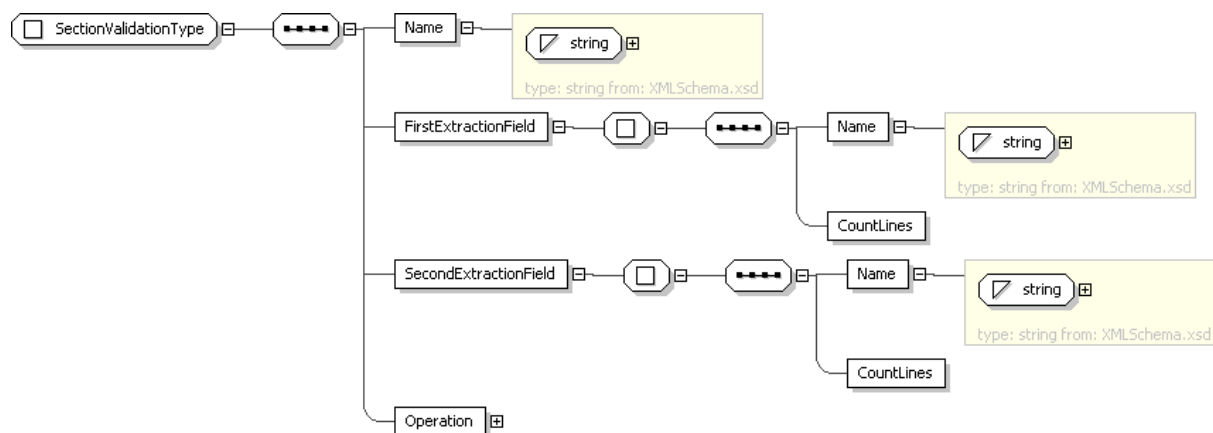


Figure 25: Section Validation Rule definition XMLSchema

There are five possible operators that can be applied:

Equal: Assuring both data columns have all their values equal in equivalent position.

Grater Than: Assuring all values in the first data column are grater than all values in second data column.

Grater or Equal Than: Assuring all values in the first data column are grater or equal than all values in second data column.

Less Than: Assuring all values in the first data column are less than all values in second data column.

Less or Equal Than: Assuring all values in the first data column are less or equal than all values in second data column.

They also rely on the column data type when performing the comparison, for example if compared columns are typed as dates in the comparison process they are compared as dates and not alphabetically. This is a very generic and powerful mechanism, making it flexible and

adaptable to any data.

4.2.2 Fields

Every section is composed by many extraction fields. These can be of three different types, each one with the appropriated characteristics for specific situations, the "Single Field", the "Special Field" and the "Table Field". The Figure 26 presents the basic structure of the field XMLSchema.

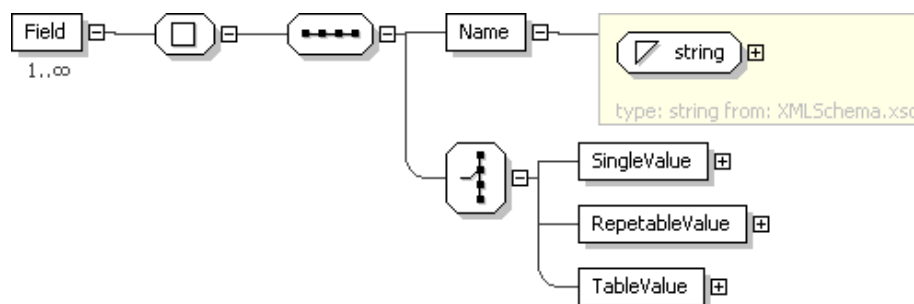


Figure 26: Field Element XMLSchema

For every field it is also possible to define value types, validation rules and missing values, as we will see further on.

4.2.2.1 Extraction Expressions

Where the webpage source type is concerned, as like explained above, there is an advantage to use XPath expressions to navigate through the document, however we wanted to have a yet more powerful extraction expression than XPaths, nevertheless maintaining its simple usage. Then, a new expression has been created from joining together regular expressions with XPath expressions. Hence the proposed execution of the expression, is evaluating first the XPath expression, being the root element always the element that defines the section. Then the result of the XPath (being it a text value, a DOM object or a group of DOM objects), is converted into text. The conversion is processed according to a boolean value, which was included to assist the usage of this expression, and determines whether the tags and their attributes are also present in the text, or if it is just the textual contents of those elements. Next the Regular expression is applied upon the previous results. Notice that the single field only uses the first result of the XPath evaluation. Figure 27 shows the expression

XMLSchema definition.

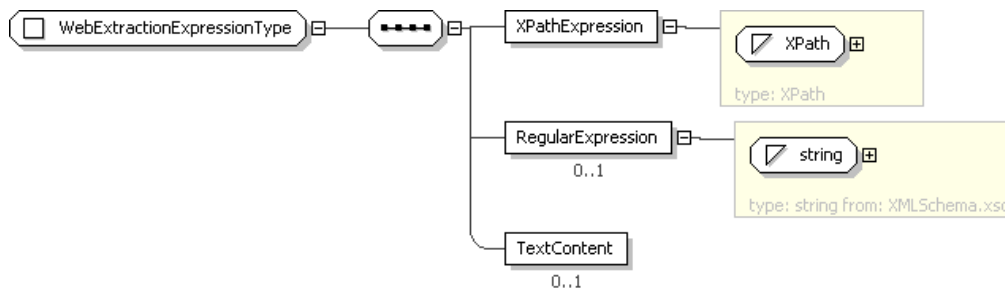


Figure 27: Webpage Extraction Expression XMLSchema

This extraction expression is used in every extraction field being explained in the next sections.

4.2.2.2 Single Field

This field is intended for the extraction of only one data value, for instance a time stamp, the document's author, or the number of items present in the document.

It is the specific element that determines this field type, which contains:

Definition: An extraction expression used in the extraction process.

Validation: A set of rules that determines the extracted data validation.

Missing Values: The Missing Values information.

Description: A simple description of the field to simplify the extension or maintenance of the extraction script.

The Figure 28 presents the XMLSchema details of the Single Field definition.

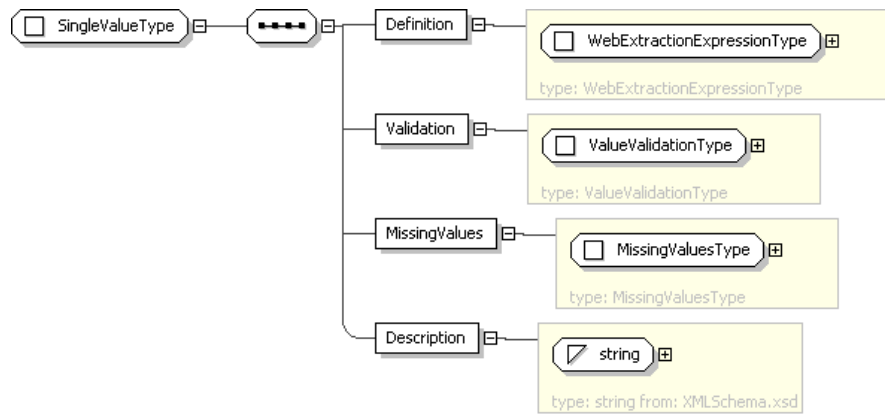


Figure 28: Single Value XMLSchema

4.2.2.3 Special Field

This field is intended for the extraction of unstructured repeated values, or repeated values that are not in the same position every time. For instance, all the links present in a section of the document or every telephone number. This type of field is defined as the Single field, being only on its execution that the extraction data is treated differently. The XPath expression is evaluated without any of the resulting values being eliminated. Next the Regular expression is applied to each one of them, resulting in the list of values the Special Field extracts. The Figure 29 shows the definition of the Special Field in XMLSchema.

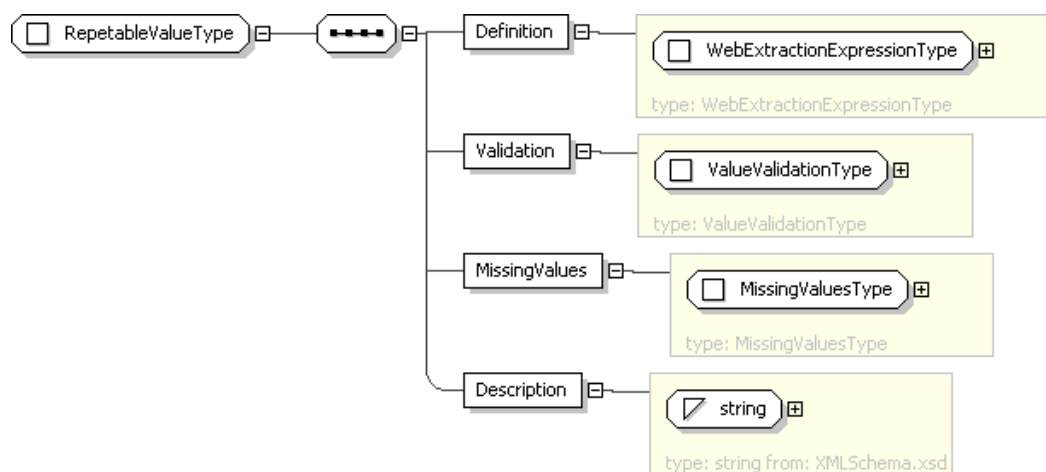


Figure 29: Special Value Field XMLSchema

To make it easy, this field can have a library of predefined Extractions Expressions, which

the user can then select and apply in his extraction. The library is in the graphical editor configurations and can be edited extended and adapted to a particular use. For instance, when creating extractions where obtaining telephone numbers is what is required, an extraction field for this purpose can be created and included in the library, becoming available to use on the next extraction to be created.

4.2.2.4 Table Field

This field is intended for the extraction of structured repeated values, like those present in one or several columns forming a table. It is composed of two elements, the element "Line" which contains the XPath expression (that when applied returns all the table rows), and the element "Columns" which contains a set of elements called "Column" representing the table columns. Each column is defined using similar fields like those used in the Special Field definition to which it was added the element "Name" to it, in order to be able to pinpoint the resulting Data column.

The Figure 30 shows the definition of the Table Field in XMLSchema.

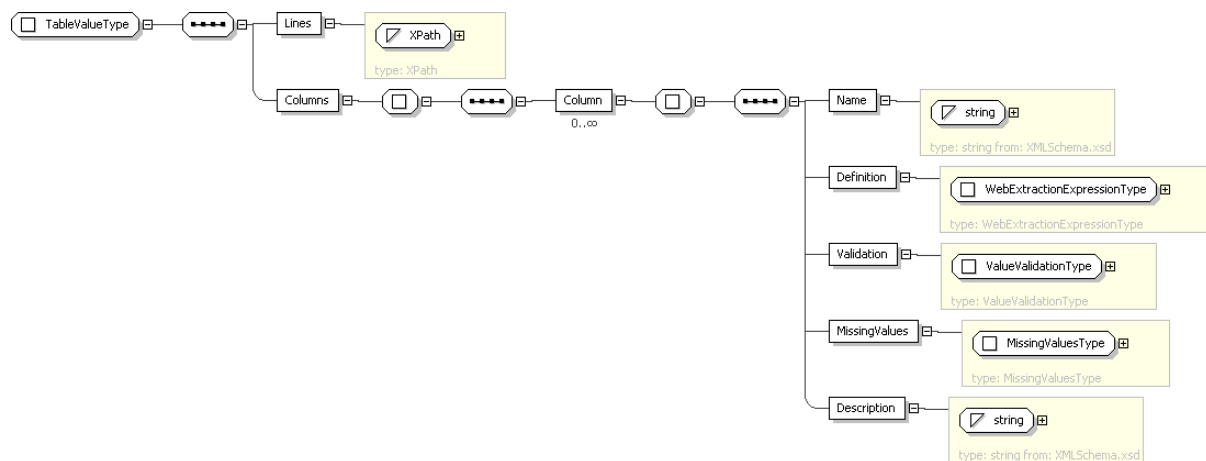


Figure 30: Table Value XMLSchema

4.2.2.5 Extraction Field Validation

Given the importance of ensuring the correctness of the extracted data, it is also necessary to have a data validation mechanism, where a type of value for each extraction field and a set of validation rules can be defined. To accomplish this, a new validation mechanism has been

created allowing the use of a set of four built-in basic value types namely the unbounded, the string, the date, the numeric and several other possible user-defined types, as shown in figure 31.

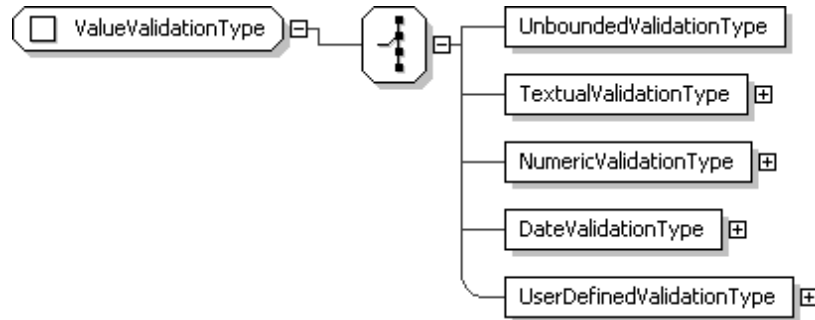


Figure 31: Value Validation Mechanism XMLSchema

The unbounded type it is a free format allowing to extract data unconditionally, being only defined by an element called "UnboundedValidationType" without validation rules.

The string type can validate any textual data, allowing to restrict the minimum and maximum length of the textual data, as well as matching it against a regular expression. It is defined (Figure 32) by an element called "TextualValidationType" and within it several elements representing each one a validation rule, the minimum and maximum length of the string, and a regular expression the extracted data has to match.

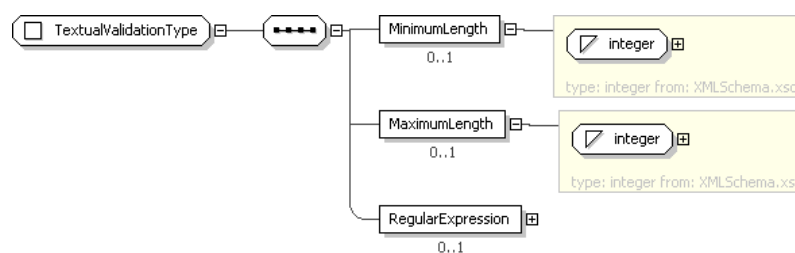


Figure 32: Textual Validation Type XMLSchema

The date and numeric data types can validate the extracted data, confirming they are numeric and a valid date respectively, and restrict their value by using the maximum and minimum value rules represented by the optional elements "MaximumValue" and "MinimumValue", within the main type elements called "NumericValidationType" and "DateValidationType",

as shown in figure 33.

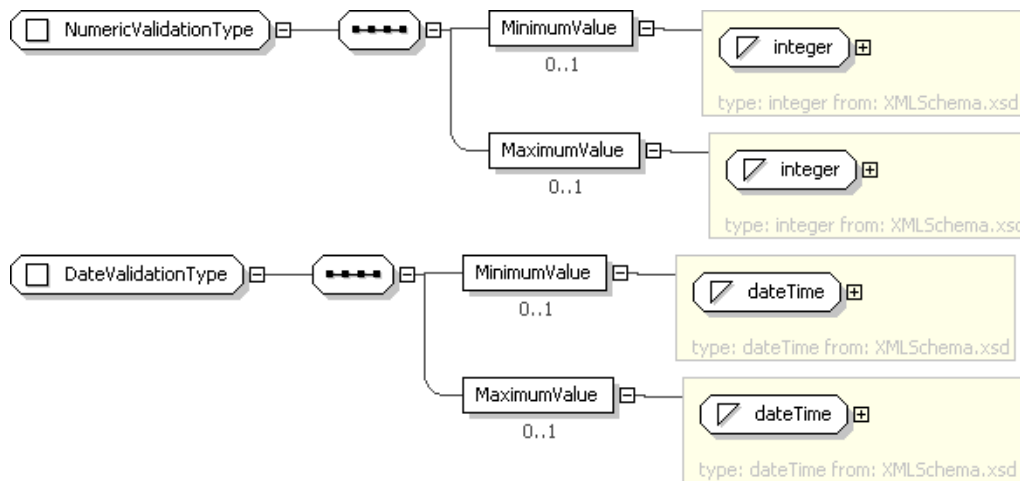


Figure 33: Numeric and Date Validation Type XMLSchema

The user defined type is a special type, developed to encapsulate a non-native type defined by a Regular Expression, to which the extracted data has to match when validating. This special type allows the user to define new validation types without needing to re-compile the whole system. Thus the addition of new types requires only the edition of the configuration file. This will also facilitate the user, since it doesn't force him to know a complex or specific regular expression, allowing him instead to select one of the previously inserted in the system, for instance an Integer, a Postal Code, an URL or an E-Mail. The user defined type is represented by an initial element called "UserDefinedValidationType" with two child elements, one for the type name and another for the regular expression, as shown in Figure 34.

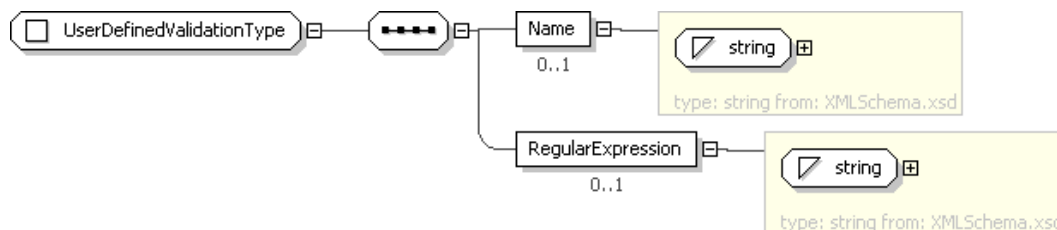


Figure 34: User Define Validation Type XMLSchema

Since these types must be available to the user, they are kept in the editor's configuration file,

with which the user can create the FFD. To simplify their implementation and also the management by the user, they are saved in the same exact way they are defined in the validation element of a field, shown in Figure 34. This also has another advantage, since the extraction system doesn't need to synchronize the user types present in the editor, or even to know which ones exist because the regular expression that defines the value type will be included in the FFD.

4.2.2.6 Missing Values

For this language a more advanced system to handle missing values was created. This is composed by a string called the "MissingValueReplacement" (which is going to replace the extracted data qualified as missing values thereby obtaining a better standardization of the data), and by a list of strings called "MissingValuesTexts" (representing the possible missing values present in the document). The Figure 35 shows the Missing Values XMLSchema.

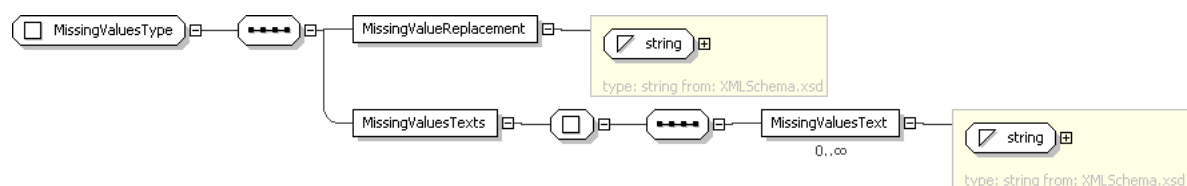


Figure 35: Field Missing Values XMLSchema

4.3 Data Delivery Module

To develop a generic extraction system, it is necessary to bear in mind that its Data Delivery must be flexible and adaptable to any situation and use. From this perspective the existing component for data delivery specifically developed and only suitable for the SESS project has become unusable in a different context, hence the need for a new data delivery system that can be used in any situation. In order to accomplish that, the new system must allow multiple deliveries per extraction process, several different ways to save or export the data files, custom metadata values, and an inner transformation process for delivering the data. To achieve this, a new language has been created to support the data delivery system and all its

new features.

First an FFD must allow multiple data deliveries, since an extraction may extract different kinds of data that should be delivered in several systems or be delivered separately. To accommodate several data deliveries, each delivery definition stays within a sub-element called "DataDelivery", being all of these grouped together in the "DataDeliveries" element. The delivery stage will process them separately generating an output for each one.

As shown in Figure 36, the Delivery definition is composed by four parts corresponding each to an inner element, containing specifications about where to physically export the file (in an element called "SaveTo"), additional information in the metadata of the delivery (in an element called "MetaData"), the data columns that will be in the delivery (in an element called "DataColumns"), the validation information used to verify the data before delivery (in an element called "Validations"), and an optional field where it can pass an XSLT to be applied to the delivery before its exporting (in an element called "XSLTProcess").

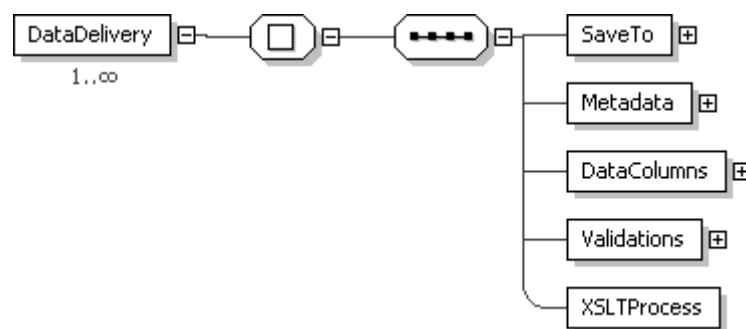


Figure 36: DataDelivery XMLSchema

4.3.1 Metadata

The "Metadata" inside the delivered output is the data that provides information about that particular delivery. This system allows the user to define customized Metadata fields, where each field is composed by a name and a value. The value can be one of two different types, a constant value or a referential value addressing an extracted field. Conceptually the time factor has been intended as a very important issue and commonly used in data extraction, so a special "holder" has been created to also allow the definition of initial and ending time where the data is related to. As in the case of a data set from a temperature sensor obtained within a specific time range, which should reflect its instant measurement. Furthermore, the system

automatically apply a time stamp in an element called "DeliveryDate", which is added to the correspondent Metadata field in the output, defining its delivery time.

4.3.2 Data Columns

The "Data Columns" element is where the list of (user specified) string type references to the extracted data for further use during the delivery process is kept.

4.3.3 Validation

In order to enable data validation after extraction and transformation, assuring the quality and consistency of the data to be delivered, the system needed a data validation mechanism. Therefore, the validation mechanism used in the extraction phase was adapted to validate the data in this phase, since the mechanism is powerful and flexible. To group all the validation rules used to compare two data columns, they are kept within the element "Validation", and for validating a single data column by giving it a type and validation rules an element also called "Validation" inside each Data Column is used .

4.3.4 XSLT Process

The "XSLT Process" element is where the user can store the XSL Transformations that will be applied to the delivery result converting it into the final formatted output. Allowing therefore, a customized inner process on the delivered data, without any external intervention, whenever a different output format is needed.

4.3.5 Save To

To provide a good adaptive capability so it can be generic enough to be used in any scenario, an export system has been created which besides already permitting quite different (and commonly used) export methods, it may well in the future be extended to support new ones.

Those that are already integrated are:

File: Contains an attribute with the file path, and simply writes the output generated by the delivery to a file in the specified path. It can also add a time stamp to the name

of the file to clearly show the order of the outputted files.

E-Mail: Contains an attribute with the e-mail address to where the output generated by the delivery should be dispatched as well as the needed attributes for the SMTP protocol like host, username and password.

External Program: Contains an attribute with the file path, where the output generated by the delivery will be written, as well as an attribute with the program file path and parameters that will be executed right after the delivery data has been saved. This not only gives the ability to externally convert the delivery format, but it also allows the execution of subsequent procedures permitting the data processing chain continuity.

Web Service: Contains an attribute with the host, where the Web Service is located, as well as an attribute with the path to the Web Service. With this information, it will transfer the generated delivery XML to that Web Service. This delivery was not implemented due to time restrictions.

4.3.6 Output Delivery

To understand the delivery process, a brief description of the operation is presented.

The process starts by using the Metadata and Data Columns definition, where an output is generated in a specific XML-based language using the basic structure, shown in Figure 37.

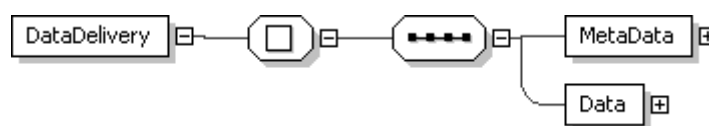


Figure 37: Delivery output basic structure

Considering the "DataDelivery" as the root element, we can focus on its two child elements:

MetaData: This element is populated based on the Delivery "Metadata" definition, which will contain the resulting elements named in the "Custom Metadata Fields" (from the definition), with their respective values.

Data: This element is filled in based on the Delivery "Data Columns" definition, which will contain a list of elements called "Column" with an attribute

"columnName", reflecting the name of the data to be exported in each column. Each column will then contain the corresponding extracted values in the referenced extraction field. The Figure 38 shows the XMLSchema corresponding to this element.

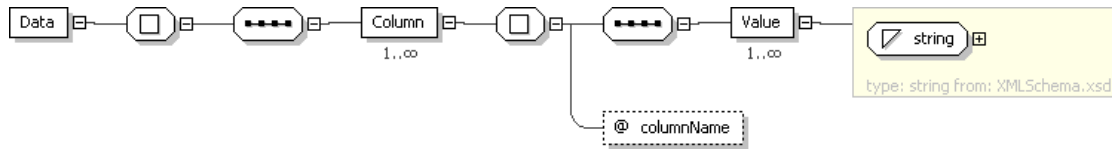


Figure 38: Delivery output Data element XMLSchema

If there is any XSLT Template rule within the "XSLTProcess" element, then the previous delivery result will be converted through the referred XSLT (Figure 39). Otherwise the system will skip directly to the "SaveTo" process.

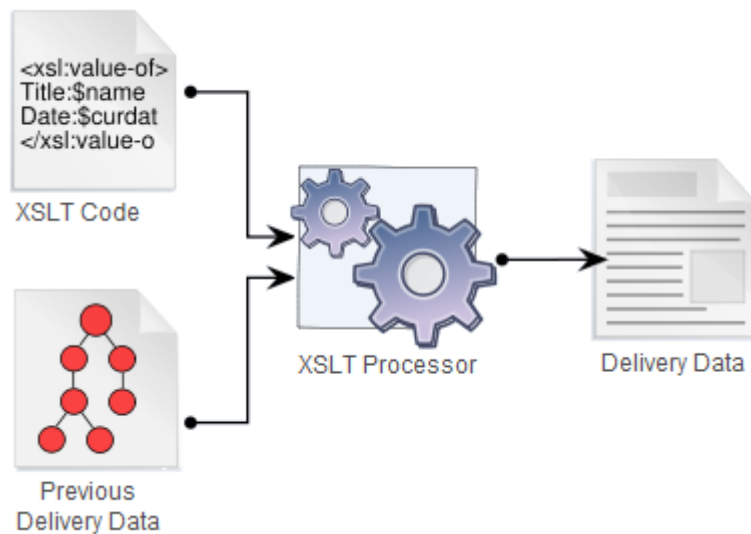




Figure 39: XSLT Process

The delivery process is completed by the "SaveTo" action, which will save the delivery result to its final destination in accordance with the existing definition.

Chapter 5 - Implementation



This chapter presents the implementation details of the WEB Data Extraction and Data Delivery systems, as well as the graphical editor supporting them.



5.1 Supporting Multiple Source Types.....	72
5.2 Webpage Data Extractor.....	73
5.3 New Data Delivery	74
5.4 The Graphical Editor.....	75
5.5 Validation.....	87

This chapter is focused on the implementation of the system described in the thesis. It explains how the extraction system most important functionalities were implemented. The implementation of the extraction editor is also explained in detail, as well as its usage.

5.1 Supporting Multiple Source Types

Since the beginning, the idea was to develop a generic webpage data extraction system capable of being used on any webpage, by any user for any extraction purpose. However, considering the existence of a similar system directed to semi-structured text files, it was decided to make use of the previous one to create a new more advanced extraction system capable of supporting multiple data source types on the assumption of maintaining the initial requirements. So, to take advantage of the existing components in the previous system implemented in Java and because the language has powerful features like its portability, security, easy usage, rich libraries and also being object oriented, it was decided that the implementation of the project be in Java. Therefore, the data processing core (which contains the data extraction and transformation procedures joined) has been rearranged in order to split both parts, the extraction from the transformation. This way, it was then possible to create a generic mechanism that can distinctly execute different extractions in accordance with the source type. This was achieved through the creation of an interface enforcing any class implementing it to contain every public method needed by the system to communicate with that extraction procedure, ensuring therefore the ability of new classes inclusion in order to process different data source types. Then a change was made to the semi-structured text file extractor code in order to adapt it to this new mechanism. This approach ensures that the previous extraction process remains working with all of its functionalities, however allowing the addition of new data source types to the extraction process.

At this stage all the conditions have been established to allow the creation of the new extraction class called "WebPageExtraction", which specifically supports the extraction of data from the webpage source type.

5.2 Webpage Data Extractor

As previously explained in this thesis, the object model of the rendered webpage can be different from the object model of the original page source file, due to simple scripts execution inside the page rendering. Also, the execution of more complex scripts that may even start HTTP requests, incorporating the dynamically received information, from different sources composing the rendered webpage, thus resulting in a complete loss of the source file concept. This led to the need of a more sophisticated extractor that could be used from the simplest and static webpages to the most complex and dynamic ones already existing and that increasingly will further populate the web, becoming the most common.

To accomplish this, it was decided to develop an extractor based on rendered webpages instead on their source files. This new extractor needs then to incorporate a rendering engine in its implementation in order to be able to perform the extraction from the rendered webpage.

As already seen in section 2.3, a study has been made on several existing engines from where the "Browser" component of the SWT Library has revealed good characteristics. It contains the necessary features for this specific implementation as well as seeming fairly simple to integrate within a Java application, Therefore being the selected one to be used in the new system.

To incorporate the rendering engine in the new extraction mechanism, a class has been created, called "BackgroundBrowser", which includes all methods needed by the system to reach the data within the rendered webpage, therefore avoiding to hardcode the system to a specific rendering engine. To make it possible for this class to supply the required methods, a JavaScript layer of code has been created specifically to be executed under the rendering engine with the aim of retrieving the data within the rendered webpage. This layer supports the concepts of Section, Single field, Special field and Table field as prototypes, providing the adequate extraction methods to each value type. It is also in this layer where the XPath expressions (that are part of the extraction language and in particularly the extraction expressions) are evaluated according to the rendered webpage document object model (DOM), which is maintained within the rendering engine.

It is to the "BackgroundBrowser" class that is affected the task of initiating the execution of

the JavaScript layer, using the component browser method "boolean org.eclipse.swt.browser.Browser.execute(String arg0)", passing the generated code with the XPath expressions. In return it expects the script execution result, passed through an intermediate data container that simultaneously the JavaScript can modify and where the Java "BackgroundBrowser" class, alerted of any modification, can retrieve the data. After this class implementation that supports the webpage rendering and the retrieving of data through the evaluation of XPath expressions, the necessary classes have been modularly created to support the extraction language. Then each module has a strict function inside the extraction, making up the extraction language core. All of these modules are supported by specific components of the extraction language detailed in the section 4.2. Here in the Java classes is where the regular expressions are evaluated.

Briefly the referred modules are as follows: "WebPageExtraction", "WebPageSection", "WebPageExtractionExpression", "WebPageMissingValues", "WebPageSingleField", "WebPageSpecialField", "WebPageTableField", "WebPageTableColumn".

Having already implemented all the necessary classes to the data extraction on the webpage source type, this extraction component can now be integrated into the newly revised system, thus obtaining a generic and powerful extraction system capable of achieving the initially proposed goal of performing extractions from webpages and semi-structured text files, as well as maintaining its ability for further expansions to new data source types.

5.3 New Data Delivery

As referred in the section 4.3, a new data delivery system became necessary, then a new language was fully developed to support it as detailed in the same section.

To implement the "Data Delivery" system supported in the newly created language, several classes have been developed. The system is also modular where each class has a specific role corresponding to a language component forming the whole "Data Delivery" process.

The modules that implement it are as follows:

"DataDeliveries", "DataDelivery", "DataDeliveryCustomMetadata",
"DataDeliveryCustomMetadataConstant", "DataDeliveryCustomMetadataReference",

"DataDeliveryMetadata", "DataDeliveryMetadataTime", "DataDeliveryMetadataMaxTime", "DataDeliveryMetadataMinTime", "DataDeliveryColumn", "DataDeliveryXSLTProcess", "DataDeliverySaveTo", "DataDeliverySaveToExternalProgram", "DataDeliverySaveToFile", "DataDeliverySaveToEMail".

The idea that the system should be generic as well as extensible has always been present. In this implementation with that purpose in mind a generic mechanism to save the exported data was created, thus making it possible to use the already developed methods (Export to: File, E-Mail and External program) that are quite generic and commonly used as well as integrating new ones. This was accomplished through the creation of an interface called "DataDeliverySaveTo", that all present and future classes supporting an export method have to implement. Provided that when adding new classes the system be re-compiled.

The new implemented system assures that all the previous functionalities remain possible, like the GID for instance, that are data specific identifiers (present in the delivery metadata) and mandatory in the previous system. They can be defined within the new system through the custom metadata fields getting the same final result.

5.4 The Graphical Editor

For the graphical editor some objectives have been outlined, as the interface user-friendliness, its intuitiveness and similar use with the previous extractor editor, thus providing a better integration and adaptability for the user. Furthermore, it has been considered as a major issue the hiding of the inner language and its technical details from the user, so then it may be used by domain users with less computer training.

Two new editor components have been implemented in accordance with the new language components, namely the Webpage Extraction and the Data Delivery System. There was no need for the creation of a new transformation editor because the corresponding language component has been kept unchanged. This new editor was implemented with the Java Graphical User Interface Toolkit "SWT" (Standard Widget Toolkit - from Eclipse Foundation), as it is the most suitable graphic library to be used with the selected webpage rendering engine. This caused the adopted solution for the editor not to be integrated with the

previous one, as it uses a different graphical library. Then, if a webpage extraction needs data transformation before delivery, it will be necessary to use the transformation component within the former editor. The graphical editor is then composed of two components (Webpage extraction and Data delivery), which will be introduced in sections 5.4.1 and 5.4.3.

5.4.1 The Extraction Editor

The webpage extraction component interface is divided into four distinct areas, as shown in Figure 40.

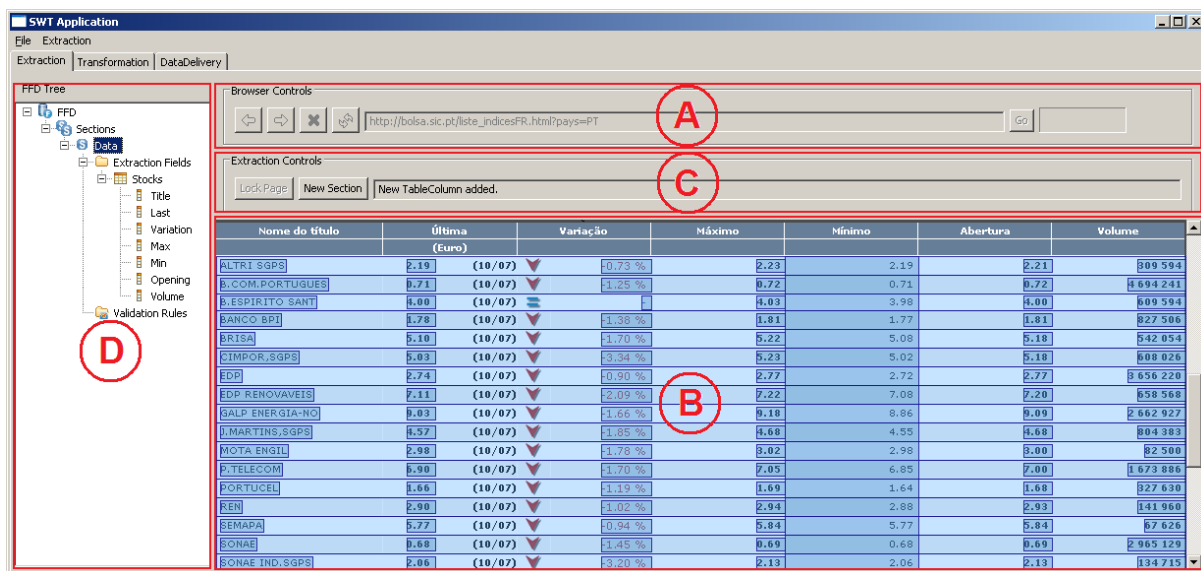


Figure 40: Webpage extraction component interface

Browser Controls (A) - It groups the basic browser controls allowing the navigation as well as moving back, forward, stopping or refreshing the webpage. It also has a gauge bar showing the webpage loading progress.

Browser (B) - Is the area where it is shown the rendered webpage. It is also used to navigate to the target webpage and during the extraction script creation it is where the user can select the extraction elements directly on the rendered webpage and monitor the state of the current extraction.

Extraction Controls (C) - It contains the controls for the creation of the extraction script, allowing the lock of the webpage making it the extraction script sample page, from where the user may start the process of adding new sections, new single value fields, new table value fields and new table columns. All of these actions become visible on the browser as well as in the FFD Tree. In this area there is also a tip box to assist the user during the script creation to point to the next possible actions or the result of one of them.

FFD Tree (D) - Is where the sections and extraction fields that compose the current extraction are represented. It also permits editing or removing any of these extraction elements by selecting them through the popup menu (mouse right click). The adding actions of extraction elements are reflected in the Tree giving an hierarchical view of them. The Tree icons are shaped in accordance with the element type, making it simpler and intuitive to use. The FFD Tree also gives the possibility of showing graphically on the rendered webpage the extraction element through its selection.

5.4.1.1 Script Creation

The creation of an extraction script is a quite simple task. The user starts by navigating to the target webpage through the "Browser controls" or directly in the "Browser", then locks the webpage (in the Extraction controls) making it the extraction script sample page. The next action will be the identification of the data areas on the webpage to create the sections. The process is initiated for each one with the addition of a new section, and then pointing the mouse inside it (Figure 41.A). The editor will let the user adjust the section by choosing the element according to the graphical pretended size, which is shown outlined in the browser (Figure 41.B), and finally naming it in the wizard (Figure 41.C). In the wizard the several possible extractions expression which can be used to create this section are also shown by order of importance. The editor has already selected the recommended one, so only advanced users should alter this selection or create a new expression. In section 5.4.1.4 is described in more detail how the expressions are generated, how they are ordered by relevance and how the recommendation is suggested.

Nome do título	Última (Euro)	Variação	Máximo	Mínimo	Abertura
ALTRI SGPS	1.77 (13:37)	-3.86 %	1.84	1.71	1.84
B.COM.PORTUGUES	0.73 (13:54)	-6.19 %	0.77	0.71	0.77
B.ESPIRITO SANT	5.22 (13:51)	-1.71 %	5.31	5.17	5.31
BANCO BPI	1.52 (13:51)	-2.63 %	1.55	1.50	1.55
BRISA	5.26 (13:55)	-0.60 %	5.35	5.25	5.25
CIMPOR,SGPS	3.70 (13:45)	-2.40 %	3.80	3.69	3.78
EDP	2.53 (13:55)	-1.06 %	2.55	2.51	2.54
EDP RENOVAVEIS	5.72 (13:54)	+0.81 %	5.74	5.65	5.67
GALP ENERGIA-NO	8.63 (13:54)	-3.47 %	8.97	8.61	8.94
J.MARTINS,SGPS	3.15 (13:55)	-8.16 %	3.42	3.10	3.40
MOTA ENGL	2.21 (13:55)	-3.49 %	2.28	2.21	2.28
P.TELECOM	6.48 (13:55)	-0.28 %	6.59	6.40	6.45
PORTUCEL	1.41 (13:47)	-1.40 %	1.43	1.41	1.43
REN	3.85 (13:51)	-0.97 %	3.88	2.96	3.87
SEMAPA	5.74 (13:55)	-2.12 %	5.85	5.70	5.85
SONAE	0.47 (13:45)	-1.46 %	0.48	0.47	0.48
SONAE IND.SGPS	1.55 (13:20)	-3.19 %	1.58	1.53	1.58

(A)

Nome do título	Adjust Sections	Move-To Bottom	Volume		
ALTRI SGPS	0.77 (13/02) ▲ +1.05 %	0.78	0.75	0.76	7.4
B.COM.PORTUGUES	5.30 (13/02) ▲ -	5.38	5.25	5.30	7
B.ESPIRITO SANT	1.58 (13/02) ▲ +0.13 %	1.60	1.57	1.58	4
BANCO BPI	5.35 (13/02) ▲ +0.94 %	5.38	5.30	5.31	5
BRISA	3.75 (13/02) ▲ +2.88 %	3.76	3.70	3.70	5
CIMPOR,SGPS	2.62 (13/02) ▼ -1.88 %	2.78	2.62	2.78	5.1
EDP	5.76 (13/02) ▲ +1.41 %	5.76	5.63	5.71	5
EDP RENOVAVEIS	9.15 (13/02) ▲ +0.52 %	9.28	8.98	9.11	16
GALP ENERGIA-NO	3.54 (13/02) ▲ +3.03 %	3.57	3.49	3.58	13
J.MARTINS,SGPS	2.32 (13/02) ▼ -0.85 %	2.35	2.30	2.35	9
MOTA ENGL	6.64 (13/02) ▲ +3.09 %	6.65	6.45	6.45	27
P.TELECOM	1.46 (13/02) ▲ +0.27 %	1.46	1.45	1.46	1
PORTUCEL	3.15 (13/02) ▲ +0.64 %	3.19	3.13	3.13	4
REN	5.94 (13/02) ▼ -1.98 %	6.13	5.94	6.03	12
SEMAPA	0.48 (13/02) ▲ +0.83 %	0.49	0.48	0.48	5.7
SONAE	1.61 (13/02) ▲ +1.19 %	1.62	1.59	1.59	
SONAE IND.SGPS	1.16 (13/02) ▲ +2.11 %	1.17	1.14	1.14	1
SONAE COM.SGPS	0.48 (13/02) -	0.48	0.48	0.48	2
TEIXEIRA QUARTE	4.89 (13/02) ▼ -0.85 %	4.14	4.09	4.13	2
ZON MULTIMEDIA					

(B)

Section Name: Data1

Definition XPath:

- /html/BODY/TABLE[@id="Mido"]/TBODY/TR/TD/FORM/TABLE/TBODY/TR[4]/TD/TABLE/TBODY
- /html/BODY/TABLE/TBODY/TR/TD/FORM/TABLE/TBODY/TR[4]/TD/TABLE/TBODY
- /html[1]/BODY[1]/TABLE[1]/TBODY[1]/TR[1]/TD[1]/FORM[1]/TABLE[1]/TBODY[1]/TR[4]/TD[1]/TABLE[1]/TBODY[1]
- /*[1]/*[2]/*[2]/*[1]/*[1]/*[1]/*[1]/*[2]/*[1]/*[4]/*[1]/*[1]/*[1]
- /

New Section Cancel

(C)

Figure 41: Addition of a new Section

Having at least one section created, the user may now select the extraction elements.

To add a Single Field he must first have the section selected. Then initiates the addition of a new single field, and pointing the mouse on the target data selects the desired element to extract(Figure 42.A). Finally through the wizard, it is possible to re-adjust or re-define the extraction details (Figure 42.B), define the value type and create validation rules, missing values and give it a name. In the extraction details several options are available and ordered

by relevance, being the suggested one already selected. Only advance users should alter the selection or create a new. More details on how these expressions are generated and suggested are describe in section 5.4.1.4.

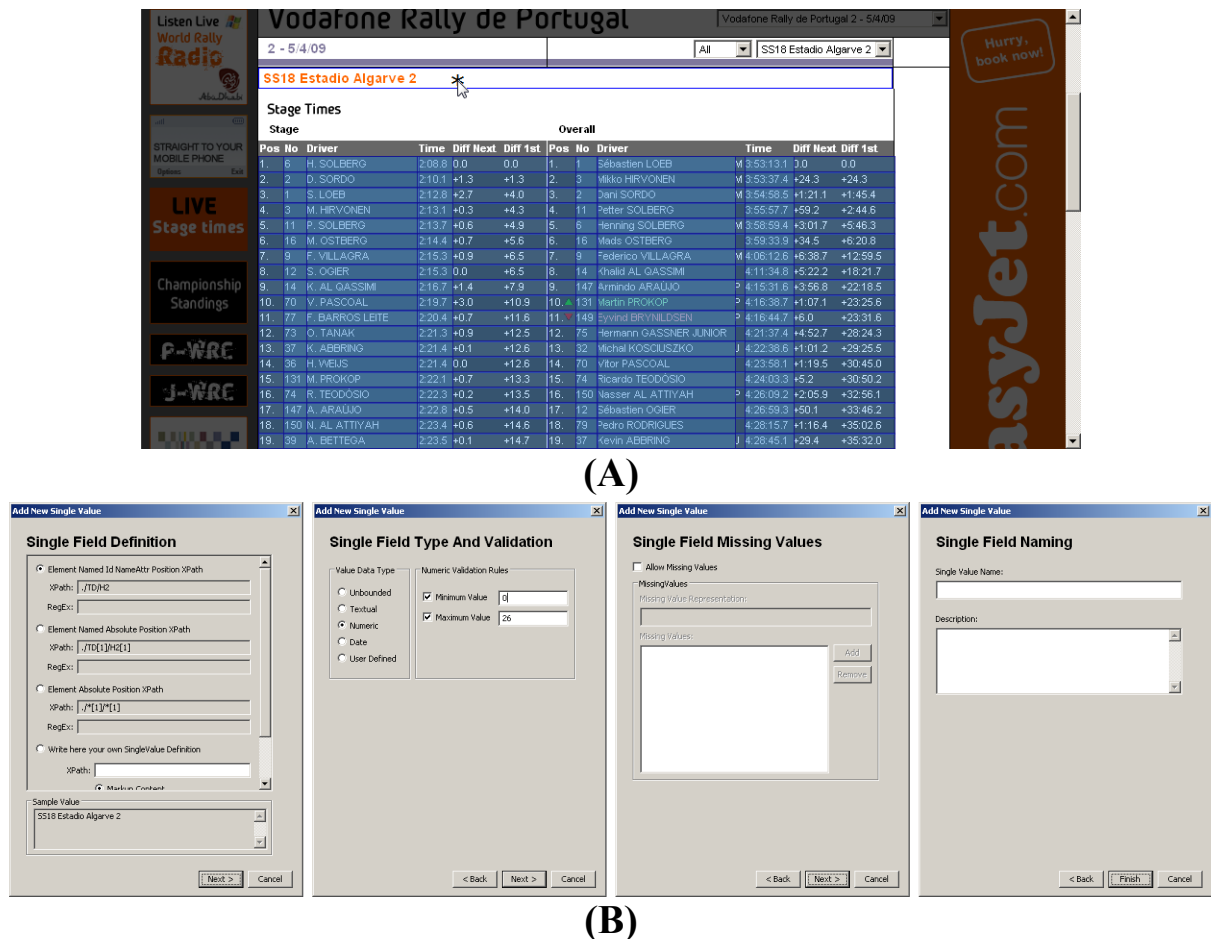


Figure 42: Addition of a new Single Field

To add a Table Field the user must have the section previously selected and initiate the new Table Field process. Then identify the lines that compose the table by pointing the mouse on any table cell (Figure 43.A), and choosing the adequate option that properly splits the lines which compose the table in accordance to the graphical line dividers shown on the rendered webpage (Figure 43.B), and then finally filling in the name in the wizard (Figure 43.C). This wizard has also the option of excluding a specified number of lines, eliminating for instance the headers of the table. There are also several options of XPath expressions for the chosen line, these are displayed in accordance with their relevance, and the suggested one is already selected. Only advanced users should alter this option or create a new one. The section

5.4.1.4 describes in more detail how the expressions are generated, classified and the best one suggested.

Vodafone Rally de Portugal

SS18 Estadio Algarve 2

Stage Times						Overall							
Pos	No	Driver	Time	Diff	Next	Diff 1st	Pos	No	Driver	Time	Diff	Next	Diff 1st
1	6	H. SOLBERG	2:08.8	0.0	0.0	0.0	1	1	Sebastien LOEB	M 3:53:13.1	0.0	0.0	0.0
2	2	D. SORDO	2:10.1	+1.3	+1.3	+1.3	2	3	Mikko HRVONEN	M 3:53:37.4	+24.3	+24.3	+24.3
3	1	S. LOEB	2:12.8	+2.7	+4.0	+4.0	3	2	Dani SORDO	M 3:54:59.0	+1:21.1	+1:45.4	+1:45.4
4	3	M. HRVONEN	2:13.1	+0.3	+4.3	+4.3	4	11	Peter SOLBERG	M 3:55:57.7	+52.2	+2:44.6	+2:44.6
5	11	P. SOLBERG	2:13.7	+0.6	+4.9	+4.9	5	6	Herrning SOLBERG	M 3:58:59.4	+3:01.7	+6:46.3	+6:46.3
6	18	M. OSTBERG	2:14.4	+0.7	+5.6	+5.6	6	16	Mads OSTBERG	M 3:59:33.9	+3:45.5	+8:20.8	+8:20.8
7	9	F. VILLAGRA	2:15.3	+0.9	+6.5	+6.5	7	9	Federico VILLAGRA	M 4:06:12.6	+6:38.9	+12:59.5	+12:59.5
8	12	S. OGIER	2:15.3	0.0	+6.5	+6.5	8	14	Khalid AL QASSBI	P 4:11:34.9	+5:22.2	+11:21.7	+11:21.7
9	14	K. AL QASSBI	2:16.7	+1.4	+7.9	+7.9	9	147	Armando ARAUJO	P 4:15:31.6	+3:56.8	+22:18.5	+22:18.5
10	70	V. PASCOAL	2:19.7	+3.0	+10.9	+10.9	10	131	Martin PROKOP	P 4:16:38.7	+1:07.1	+23:25.6	+23:25.6
11	77	F. BARROS LEITE	2:20.4	+0.7	+11.6	+11.6	11	149	Eyvind BRYNLOSEN	P 4:18:44.7	+5.0	+23:31.6	+23:31.6
12	73	O. TANAK	2:21.3	+0.9	+12.5	+12.5	12	75	Hermann GASSNER JUNIOR	J 4:21:37.4	+4:52.7	+26:24.3	+26:24.3
13	37	K. ABERING	2:21.4	+0.1	+12.8	+12.8	13	32	Michal KOSULSIZKO	J 4:22:38.6	+1:01.2	+29:25.5	+29:25.5
14	38	H. VIEUS	2:21.4	0.0	+12.8	+12.8	14	70	Vitor PASCOAL	J 4:23:58.1	+1:19.5	+30:45.0	+30:45.0
15	131	M. PROKOP	2:22.1	+0.7	+13.3	+13.3	15	74	Ricardo TEODOSIO	P 4:26:03.2	+5.2	+30:50.2	+30:50.2
16	74	R. TEODOSIO	2:22.3	+0.2	+13.5	+13.5	16	150	Nasser AL ATTIAH	P 4:28:09.2	+2:05.9	+32:56.1	+32:56.1
17	147	A. ARAUJO	2:22.8	+0.5	+14.0	+14.0	17	12	Sebastien OGIER	J 4:28:59.3	+50.1	+33:48.2	+33:48.2
18	150	N. AL ATTIAH	2:23.4	+0.6	+14.6	+14.6	18	79	Pedro RODRIGUES	J 4:28:15.7	+1:16.4	+33:08.0	+33:08.0
19	39	A. BETTEGA	2:23.5	+0.1	+14.7	+14.7	19	37	Kevin ABERING	J 4:28:45.1	+29.4	+36:32.0	+36:32.0

(A)

Adjust Line element:

SS18 Estadio Algarve 2

Stage Times						Overall							
Pos	No	Driver	Time	Diff	Next	Diff 1st	Pos	No	Driver	Time	Diff	Next	Diff 1st
1	6	H. SOLBERG	2:08.8	0.0	0.0	0.0	1	1	Sebastien LOEB	M 3:53:13.1	0.0	0.0	0.0
2	2	D. SORDO	2:10.1	+1.3	+1.3	+1.3	2	3	Mikko HRVONEN	M 3:53:37.4	+24.3	+24.3	+24.3
3	1	S. LOEB	2:12.8	+2.7	+4.0	+4.0	3	2	Dani SORDO	M 3:54:58.5	+1:21.1	+1:45.4	+1:45.4
4	3	M. HRVONEN	2:13.1	+0.3	+4.3	+4.3	4	11	Peter SOLBERG	M 3:55:57.7	+52.2	+2:44.6	+2:44.6
5	11	P. SOLBERG	2:13.7	+0.6	+4.9	+4.9	5	6	Herrning SOLBERG	M 3:58:59.4	+3:01.7	+6:46.3	+6:46.3
6	18	M. OSTBERG	2:14.4	+0.7	+5.6	+5.6	6	16	Mads OSTBERG	M 3:59:33.9	+3:45.5	+8:20.8	+8:20.8
7	9	F. VILLAGRA	2:15.3	+0.9	+6.5	+6.5	7	9	Federico VILLAGRA	M 4:06:12.6	+6:38.9	+12:59.5	+12:59.5
8	12	S. OGIER	2:15.3	0.0	+6.5	+6.5	8	14	Khalid AL QASSBI	P 4:11:34.9	+5:22.2	+11:21.7	+11:21.7
9	14	K. AL QASSBI	2:16.7	+1.4	+7.9	+7.9	9	147	Armando ARAUJO	P 4:15:31.6	+3:56.8	+22:18.5	+22:18.5
10	70	V. PASCOAL	2:19.7	+3.0	+10.9	+10.9	10	131	Martin PROKOP	P 4:16:38.7	+1:07.1	+23:25.6	+23:25.6
11	77	F. BARROS LEITE	2:20.4	+0.7	+11.6	+11.6	11	149	Eyvind BRYNLOSEN	P 4:18:44.7	+5.0	+23:31.6	+23:31.6
12	73	O. TANAK	2:21.3	+0.9	+12.5	+12.5	12	75	Hermann GASSNER JUNIOR	J 4:21:37.4	+4:52.7	+26:24.3	+26:24.3
13	37	K. ABERING	2:21.4	+0.1	+12.8	+12.8	13	32	Michal KOSULSIZKO	J 4:22:38.6	+1:01.2	+29:25.5	+29:25.5
14	38	H. VIEUS	2:21.4	0.0	+12.8	+12.8	14	70	Vitor PASCOAL	J 4:23:58.1	+1:19.5	+30:45.0	+30:45.0
15	131	M. PROKOP	2:22.1	+0.7	+13.3	+13.3	15	74	Ricardo TEODOSIO	P 4:26:03.2	+5.2	+30:50.2	+30:50.2
16	74	R. TEODOSIO	2:22.3	+0.2	+13.5	+13.5	16	150	Nasser AL ATTIAH	P 4:28:09.2	+2:05.9	+32:56.1	+32:56.1
17	147	A. ARAUJO	2:22.8	+0.5	+14.0	+14.0	17	12	Sebastien OGIER	J 4:28:59.3	+50.1	+33:48.2	+33:48.2
18	150	N. AL ATTIAH	2:23.4	+0.6	+14.6	+14.6	18	79	Pedro RODRIGUES	J 4:28:15.7	+1:16.4	+33:08.0	+33:08.0
19	39	A. BETTEGA	2:23.5	+0.1	+14.7	+14.7	19	37	Kevin ABERING	J 4:28:45.1	+29.4	+36:32.0	+36:32.0

(B)

Add New Table Field

Table Field

Table Name:
Stage

Lines XPath:
 Element Named Id NameAttr Position XPath
 ./tbody/tr[3]/td[1]/div[1]/table/tbody/

 Element Named Absolute Position XPath
 ./tbody[1]/tr[3]/td[1]/div[1]/table[1]/tr

 Element Absolute Position XPath
 ./*[1]/*[3]/*[1]/*[2]/*[2]/*[1]/tr

 Write here your own SingleValue Definition

Ignore first 2 Lines

New Table Cancel

(C)

Figure 43: Addition of a new Table Field

Having the table lines already defined, with the FFD Tree table element selected the user may now delineate the columns by initiating the addition. The system responds shadowing everything around a table line, allowing the selection of the data by pointing to it (Figure 44.A). To finalize, the user may fill in the subsequent wizard with the extraction details

(Figure 44.B), the column data type and validation rules, the missing values and its name.

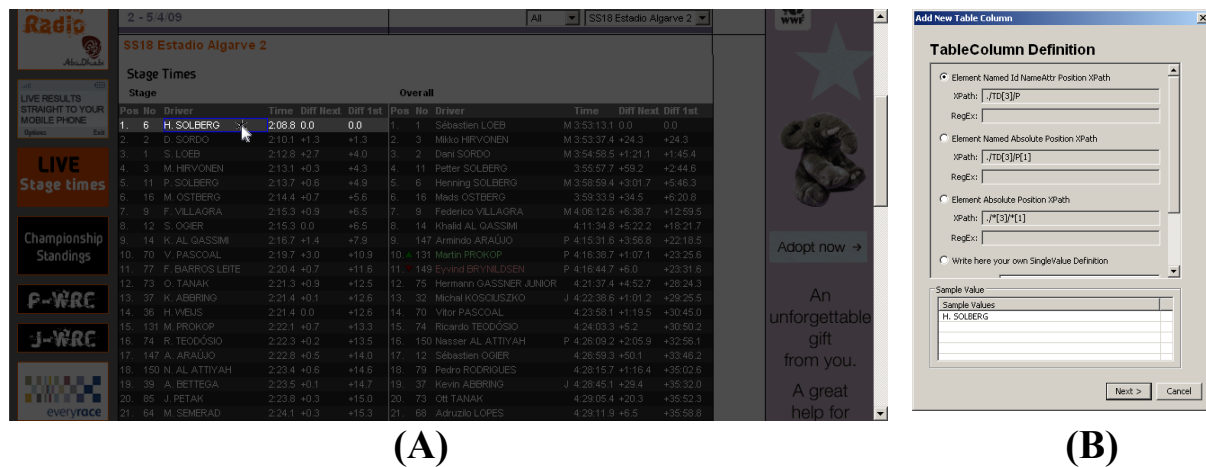


Figure 44: Addition of a new Table Column

To add a Special Field, the user must have the section previously selected and initiate the new Special Field wizard (Figure 45). In the wizard the user can choose from predefined extractions or customise a new one. The system is also prepared so new predefined extractions can be added, which are stored in the configuration file. Then like in the table column and single field the user can define the value type, create validation rules, missing values and give it a name.

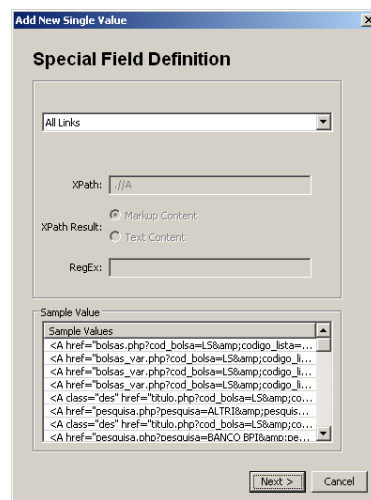


Figure 45: Special Field Wizard

5.4.1.2 GUI Implementation Details - Java Interface

This GUI (Graphical User Interface) was developed using two languages (Java and JavaScript) that seamlessly interact between them, not revealing any disruption to the user.

The Java interface is the main component that controls the whole extraction script creation process, as well as the graphical environment containing the previously explained Browser Controls, Browser, Extraction Controls and FFD Tree. It allows the user to manage the extraction script by adding, editing and removing elements (as sections, single values, table values and table columns) in a simple and intuitive way. This interface uses the core library⁴ to implement the extraction elements, maintaining their consistency and compliance with the whole system. Additionally, a specific set of user interactions has been developed in order to assist the script creation, allowing a user with low computer expertise to select the data from the rendered webpage. This was implemented separately in a JavaScript layer that this interface controls. As previously referred on the webpage extractor description, the JavaScript layer is controlled through the call of the "execute" method from the browser component, which passes the controlling generated code, expecting in return the result of that controlling code execution sent back through an intermediate data container to the Java interface. Upon receiving the webpage interactions result from the JavaScript layer, the Java interface shows the specific extraction element wizard presenting the results to the user for further decision, also allowing him to include other specific details like the data type, validation rules and name.

5.4.1.3 GUI Implementation Details - JavaScript Layer

The JavaScript layer was created specifically to perform the interaction between the user and the webpage and also to graphically display the extraction elements on the rendered page. This layer is executed inside the rendering engine, being added and started when the user locks the page, thus remaining active during all the script creation or edition processes. The allowed user interactions performed by this layer include all the webpage data selections through element pointing actions as well as the selection adjusting through an options frame. This layer operates within the rendering engine having access to the rendered webpage object model, which is created by the referred engine, being also responsible for the XPath expression construction. For this purpose it has been designed an heuristic modular

4 The core library is where all the components that model the extraction language are included, in order to simplify the implementation, maintaining code consistency as well as the integration of all the system components, as the current editor and the extractor among others.

mechanism, allowing for the same selected element the creation of several valid extraction expressions, which are then passed to the Java interface.

5.4.1.4 Automatic Extraction Expressions Generator Mechanism

In order to generate the extraction expressions to present to the user in a graphical editor's wizard, the "Extraction expressions generator control mechanism" was constructed (Figure 46). It starts its operation by receiving the selected element node, from the Javascript functions (which interacts with the user). Then passing it to the several "Expression generator modules", where each one with its specific heuristic generates an extraction expression composed by an XPath and a regular expression. Each of the "Expression generator modules" uses information present in the webpage structural elements as well as their position and hierarchy to generate the extraction expression. The generated expressions are then returned back to the control mechanism. Here they are grouped together and sent within a list with each specific heuristic name to the Java module "Extraction expressions classification mechanism", where they are ordered by its specific classification and then presented in a wizard for the user to choose from, with the best proposal selected. This is also a generic mechanism, since it allows the addition of more "Expression generator modules" with their specific heuristics, as well as the modification of the "Extraction expressions classification mechanism" resulting in a different proposal order.

Five different "Expression generator modules" were implemented in the system. The first one is called "ElementAbsolutePositionXPath" and it generates the XPath expression based only on the position of the specific element in relations to its element father. This is the most simple XPath expression that can be created, it is not a robust expression, since it doesn't support any change on the webpage, however it can be interpreted and processed faster than the others. The second one is called "ElementNamedAbsolutePositionXPath" and generates the XPath expression based on the position of element as well as the element name. The third is called "ElementNamedPositionXPath" and generates the XPath expression based mainly on the element's name, and their position if it is needed. The fourth expression generator module is called "ElementNamedIdPositionXPath" and creates the XPath based on the elements name, using also the "id" attribute if this exists and if it is unique, in last resort it

can also uses the position of the element. The fifth and most complex expression generator module is called "ElementNamedIdNameAttrPositionXPath", and it generates the XPath like the previous one, except it can also use the "name" attribute instead of the "id" using the same rules for this attribute. The XPath expression resulting from this generator is the more robust one.

As it is described the generator functions starts with the simplest one and evolves to a more complex one, that with enough information of the elements in the webpage, it can generate a very robust expression which even admits small changes in the webpage.

The "Extraction expressions classification mechanism" modules are located in the wizards of the several extraction fields, to allow a different order for each one. The currently implementation of this module is equal for every field and always suggests the XPath expression called "ElementNamedIdNameAttrPositionXPath", since it the most robust.

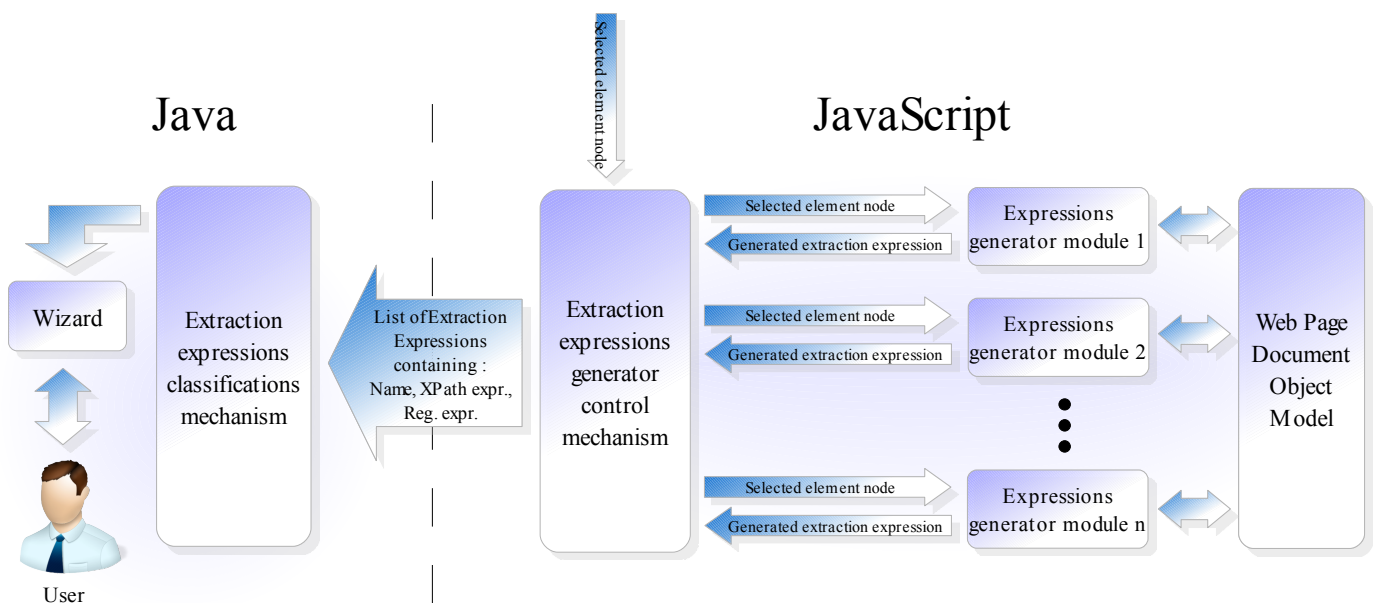


Figure 46: Automatic extraction expressions generator mechanism Diagram

5.4.2 The Webpage Extraction Tester

The Webpage Extraction Tester interface (Figure 47), was created to access the correctness of the extraction currently in development, allowing the user to see any errors, the values being extracted and compare them to original ones present in the webpage.

The screenshot shows the 'Webpage Extraction Tester' window. At the top, the 'Test Url' is 'http://bolsa.sic.pt/liste_indicesFR.html?pais=PT'. Below this is a table of stock data with columns: Nome do título, Última (Euro), Variação, Máximo, Mínimo, Abertura, and Volume. The table lists various stocks like ALTRI SGPS, B.COM.PORTUGUES, etc. Below the table is a 'Resulting Data Columns' section with a message 'FFD applied successfully.' and a table of extracted data columns: Stocks.Last, Stocks.Min, Stocks.Variation, Stocks.Title, Stocks.Volume, Stocks.Opening, and Stocks.Max.

Nome do título	Última (Euro)	Variação	Máximo	Mínimo	Abertura	Volume
ALTRI SGPS	2.19	-0.73 %	2.23	2.19	2.21	309 594
B.COM.PORTUGUES	0.71	-1.25 %	0.72	0.71	0.72	4 694 241
B.ESPIRITO SANT	4.00	-	4.03	3.98	4.00	609 594
BANCO BPI	1.78	-1.38 %	1.81	1.77	1.81	827 506
BRISA	5.10	-1.70 %	5.22	5.08	5.18	542 054
CIMPOR,SGPS	5.03	-3.34 %	5.23	5.02	5.18	608 026
EDP	2.74	-0.90 %	2.77	2.72	2.77	3 656 220
EDP RENOVAVEIS	7.11	-2.09 %	7.22	7.08	7.20	658 568
GALP ENERGIA-NO	9.03	-1.66 %	9.18	8.86	9.09	2 662 927
J.MARTINS,SGPS	4.57	-1.85 %	4.68	4.55	4.68	804 383
MOTA ENGIL	2.98	-1.78 %	3.02	2.98	3.00	82 500
P.TELECOM	6.90	-1.70 %	7.05	6.85	7.00	1 673 886
PORTUCEL	1.66	-1.19 %	1.69	1.64	1.68	327 630
REN	2.00	-1.00 %	2.04	2.00	2.03	141 060

Stocks.Last	Stocks.Min	Stocks.Variation	Stocks.Title	Stocks.Volume	Stocks.Opening	Stocks.Max
2.19	2.19	-0.73 %	ALTRI SGPS	309 594	2.21	2.23
0.71	0.71	-1.25 %	B.COM.PORTUGUES	4 694 241	0.72	0.72
4.00	3.98	-	B.ESPIRITO SANT	609 594	4.00	4.03
1.78	1.77	-1.38 %	BANCO BPI	827 506	1.81	1.81
5.10	5.08	-1.70 %	BRISA	542 054	5.18	5.22
5.03	5.02	-3.34 %	CIMPOR,SGPS	608 026	5.18	5.23
2.74	2.72	-0.90 %	EDP	3 656 220	2.77	2.77

Figure 47: Webpage Extraction Tester interface

Its purpose is to test the extraction (during creation time), with the sample page or other similar pages to show if the extraction has errors or if it will extract all the data the user requires. If there's an error in the extraction it will alert the user and show the error information so the user will be able to correct it. Otherwise, if the extraction is applied successfully, it will show the extracted data in their columns, allowing the user to know exactly the values that are being extracted as well as their graphical position on the rendered page.

For instance in figure 47, the user tested an extraction on a sample page obtaining the resulting columns. Then when he selects the "Stock.Variations" data column, causes the interface to highlight (in dark blue) the graphical position of the extracted values on the rendered page.

5.4.3 The Data Delivery Editor

The data delivery editor interface is divided into three distinct areas, as the figure 48 shows.

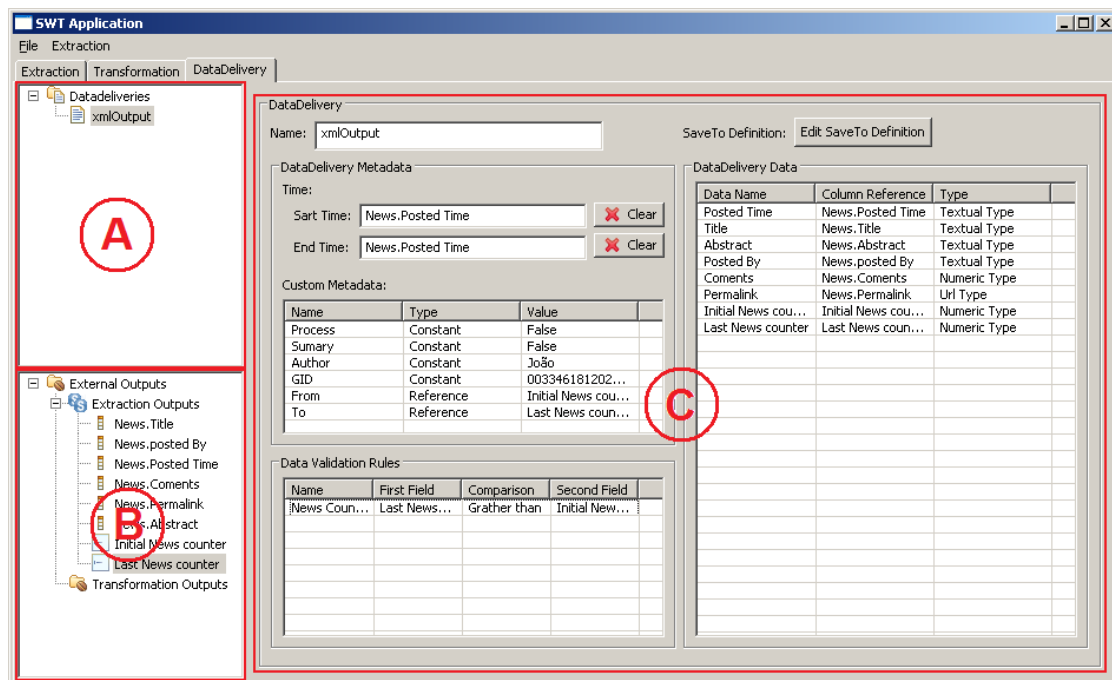


Figure 48: Data delivery editor interface

The **"Data deliveries" list (A)** - is where all the delivery processes for the current extraction are represented. It is in this area that new ones can be added and remove any existent or select one in order to watch or edit its details.

The **"External Outputs" Tree (B)** - is where the extraction and transformation outputs are shown. These outputs can be dragged and dropped to the "Data Delivery Data" and "Data Delivery Metadata" available on the third area, thus allowing its addition to the delivery.

Delivery Details (C) - is where the specific details of the data delivery selected in (A) are presented. It contains its "Name", the "Save To" definitions, the "Metadata" (containing the time metadata special holder and the "Custom Metadata" table), the "Data Validation Rules" table (which contains validation rules between the data delivery columns) and the "Delivery Data" table (which has all the data columns that will be exported, along with their type).

This editor allows defining the data delivery elements that are going to be used on the Delivery process already described, in order to export the extracted data to external systems for further processing or storage. As this stage uses the same validation mechanism as the extraction phase, its interface is identical making it easy and intuitive to use.

5.5 Validation

This section presents validation tests that were performed to assess if the extraction system complied with the requirements previously defined. The web pages used in these tests were chosen due to represent diverse areas where it would be useful to extract information, and the validations to represent real extractions scenarios. The web pages in these validations contain information about the Stock Market, Sports, Electronic Stores, News Web sites and Educational Information.

5.5.1 Stock Market

There is much interest in information about the stock market. Common people want to know if their stocks are going up, and financial experts want to calculate statistics to know what stocks to buy. However before calculating statistics they first need to extract the stocks information. We choose to extract this information from the site "bolsa.sic.pt" (Figure 49) as it lists every stock in the Portuguese Stock Market, is frequently updated and contains a lot of information about each stock that is displayed in a simple tabular form.



Figure 49: Site "bolsa.sic.pt"

To create an extraction on this webpage, a section surrounding the table listing the stocks was first created (Figure 50.A), using the auto-generated extraction expression. In that section a table was created by defining its lines (Figure 50.B) and then the columns (Figure 50.C).

Nome do título	Última	Variação	Máximo	Mínimo	Abertura	Volume
ALTRI SGPS	2.23 (16/09)	-0.58 %	2.25	2.21	2.21	37 485
B.COM.PORTUGUES	6.74 (16/09)	+0.68 %	6.74	6.73	6.73	5 075 126
B.ESPIRITO SANT	4.37 (16/09)	-0.57 %	4.45	4.34	4.39	1 614 993
BANCO BPI	1.90 (16/09)	+0.53 %	1.91	1.89	1.90	329 293
BRISA	5.30 (16/09)	+0.95 %	5.33	5.26	5.30	541 787
CIMPOR,SGPS	5.18 (16/09)	+0.45 %	5.20	5.10	5.10	273 968
EDP	2.78 (16/09)	-0.14 %	2.83	2.78	2.81	6 171 424
EDP RENOVAVEIS	7.02 (16/09)	-1.18 %	7.10	7.02	7.05	884 683
GALP ENERGIA-NO	9.47 (16/09)	+1.26 %	9.49	9.37	9.47	869 368
J.MARTINS,SGPS	4.65 (16/09)	-0.36 %	4.70	4.62	4.70	1 364 382
MOTA ENGL	3.00 (16/09)	-0.66 %	3.02	3.00	3.02	127 348
P.TELECOM	7.19 (16/09)	+1.27 %	7.20	7.07	7.15	1 987 578
PORTUCEL	1.72 (16/09)	+0.41 %	1.72	1.70	1.72	118 254
REN	2.89 (16/09)	-0.86 %	2.93	2.89	2.91	120 260
SOMPA	6.00 (16/09)	+0.50 %	6.01	5.93	5.99	81 699
SONAE	0.70 (16/09)	-	0.70	0.70	0.70	2 715 184
SONAE IND,SGPS	2.15 (16/09)	+0.94 %	2.15	2.10	2.13	228 793
SONAE COM,SGPS	1.69 (16/09)	+0.60 %	1.70	1.68	1.70	37 151
TEIXEIRA DUARTE	0.94 (16/09)	-	0.94	0.93	0.93	169 479
ZON MULTIMEDIA	3.80 (16/09)	+0.05 %	3.81	3.77	3.80	292 684

(A) Creating the Section

(B) Defining the Table Field

(C) Defining the Table Columns

Figure 50: Creating the Stock Market Extraction

The columns created are a direct match to the columns provided in the webpage, being the values extracted by the columns the Stock Name, the Last Price, the Variation, the Maximum, the Minimum, the Opening Price and the transacted Volume. Every expression used in this extraction was automatically generated by the editor. The data type validation was also defined to guarantee that all stock prices (Last, Opening, Maximum, Minimum) are numeric.

Figure 51 presents the results of applying this extraction to the sample page containing the PSI20 stocks and another random page of stocks demonstrating the extraction will work for every stocks listing from the "bolsa.sic.pt" webpage.

Resulting Data Columns

FFD applied successfully.

Stocks.Last	Stocks.Min	Stocks.Variation	Stocks.Title	Stocks.Volume	Stocks.Opening	Stocks.Max
2.23	2.21	-0.58 %	ALTRI SGPS	37 405	2.21	2.25
0.74	0.73	+0.68 %	B.COM.PORTUGUES	5 075 120	0.73	0.74
4.37	4.34	-0.57 %	B.ESPIRITO SANT	1 614 993	4.39	4.45
1.90	1.89	+0.53 %	BANCO BPI	329 283	1.90	1.91
5.30	5.26	+0.95 %	BRISA	541 787	5.30	5.33
5.18	5.10	+0.45 %	CIMPOR,SGPS	273 968	5.10	5.20
2.78	2.78	-0.14 %	EDP	6 171 424	2.81	2.83
7.02	7.02	-1.18 %	EDP RENOVAVEIS	884 083	7.05	7.10
9.47	9.37	+1.26 %	GALP ENERGIA-NO	869 368	9.47	9.49
4.65	4.62	-0.36 %	J.MARTINS,SGPS	1 364 382	4.70	4.70
3.00	3.00	-0.66 %	MOTA ENGIL	127 348	3.02	3.02
7.19	7.07	+1.27 %	P,TELECOM	1 987 578	7.15	7.20
1.72	1.70	+0.41 %	PORTUCEL	118 254	1.72	1.72
2.89	2.89	-0.86 %	REN	120 260	2.91	2.93
6.00	5.93	+0.50 %	SEMAPA	81 849	5.99	6.01
0.70	0.70	-	SONAE	2 715 184	0.70	0.70
2.15	2.10	+0.94 %	SONAE IND.SGPS	228 793	2.13	2.15
1.69	1.68	+0.60 %	SONAECOM,SGPS	37 151	1.70	1.70
0.94	0.93	-	TEIXEIRA DUARTE	169 479	0.93	0.94
3.80	3.77	+0.05 %	ZON MULTIMEDIA	292 684	3.80	3.81

(A) Sample Webpage Results

Resulting Data Columns

FFD applied successfully.

Stocks.Last	Stocks.Min	Stocks.Variation	Stocks.Title	Stocks.Volume	Stocks.Opening	Stocks.Max
0.75	0.72	+1.35 %	F.RAMA	5 125	0.72	0.75
100.00	100.00	+1.98 %	FC PORTO 6% 060	5 080	100.00	100.00
0.11	0.11	-	FENALU	150	0.11	0.11
1.61	1.61	-2.42 %	FINIBANCO,SGPS	5 500	1.63	1.63
0.14	0.14	-	FISIPE	14 285	0.14	0.14
0.11	0.11	+10.00 %	FITOR	1 988	0.11	0.11
100.00	100.00	-	FNB VARFIX TV08	0	100.00	100.00
0.02	0.02	-	FORTS 11C 0908Z	1 800	0.02	0.02
0.01	0.01	-99.05 %	FORTS 12C 0908Z	3 250	0.01	0.01
0.23	0.23	+4.55 %	FORTS 13P 0608Z	10 000	0.23	0.23
0.01	0.01	-99.56 %	FORTS 14C 0608Z	30 000	0.01	0.01
0.17	0.17	+21.43 %	FORTS 16C 0608Z	3 200	0.17	0.17
0.01	0.01	-99.63 %	FORTS 16P 0908Z	15 000	0.01	0.01
0.48	0.48	+77.78 %	FORTS 17P 0608Z	4 050	0.48	0.48
0.09	0.09	+50.00 %	FORTS 18C 0608Z	1 000	0.09	0.09
0.03	0.03	-57.14 %	FORTS 18C 0908Z	700	0.03	0.03
0.01	0.01	-50.00 %	FORTS 20C 0608Z	5 000	0.01	0.01
0.01	0.01	-	FORTS 22C 1208Z	10 000	0.01	0.01
1.60	1.39	-	FORTS 9P 0908Z	1 200	1.39	1.60
0.03	0.03	-25.00 %	FRESE 25P 0909C	1 000	0.03	0.03

(B) Randomly chosen Webpage

Figure 51: Stock Market Extraction Results

This extraction can also go a step further, making it resilient to small changes. To accomplish this, we can set the Section with the XPath Expression:

```
"html/body/table[@id="Miolo"]//table/tbody[./tr[1]/td[1]//.[text()="Nome do título"]]"
```

This way, the section is now pointing to the table that has the text "Nome do título" in the first cell of the first line, making it always possible to find the listing of stocks even if they change its position on the page. To confirm the extraction would resist to such change, a copy of the sample webpage was altered by eliminating the advertising and all tables between the page header and the stock listing table, and tested the extraction on it. Figure 52 shows the manipulated page used in the test. An alteration like this would be a consequence of a server error not creating the page headers, or a different Advertising Manager, or simply a rearrange of the webpage style. The results of the test were equal to the non-adulterated webpage, showing the extraction can resist to some changes on the target webpage.

The reason why this extraction expression had to be done by hand, was because the little or non-existent information present on the web page structural elements. As explained in section 5.4.1.4 the "Expression generator modules" rely on that information to generate robust expressions.



Nome do título	Última (Euro)	Variação	Máximo	Mínimo	Abertura	Volume
ALTRI,SGPS	2.23 (17/07)	-0.58 %	2.25	2.21	2.21	37 405
B.COM.PORTUGUES	0.74 (17/07)	+0.68 %	0.74	0.73	0.73	5 075 120
B.ESPIRITO SANT	4.37 (17/07)	-0.57 %	4.45	4.34	4.39	1 614 993
BANCO BPI	1.90 (17/07)	+0.53 %	1.91	1.89	1.90	329 283
BRISA	5.30 (17/07)	+0.95 %	5.33	5.26	5.30	541 787
CIMPOR,SGPS	5.18 (17/07)	+0.45 %	5.20	5.10	5.10	273 968
EDP	2.78 (17/07)	-0.14 %	2.83	2.78	2.81	6 171 424
EDP RENOVAVEIS	7.02 (17/07)	-1.18 %	7.10	7.02	7.05	884 083
GALP ENERGIA-NO	9.47 (17/07)	+1.26 %	9.49	9.37	9.47	869 368
J.MARTINS,SGPS	4.65 (17/07)	-0.36 %	4.70	4.62	4.70	1 364 382
MOTA ENGIL	3.00 (17/07)	-0.66 %	3.02	3.00	3.02	127 348

Figure 52: Manipulated webpage

5.5.2 Electronic Stores – Kuantokusta Delivery

The website "www.kuantokusta.pt" is a shopping guide which allows a user to compare the

price of a product on different on-line shops. Being a business it only shows products from their client's stores, requiring only a file containing the product list. The file has to respect a few strict rules in order to be correctly imported into the Kuantokusta system. Each product must come in its own line containing the product name, brand, price and link values separated by a semi-colon. The figure 53 shows a partial of a sample file ready to be imported by the system.

```

Portatil Acer Aspire 1642LMi;Acer;99.99;www.kuantokusta.net;1
Portatil Acer Aspire 1642WLMi;Acer;99.99;www.kuantokusta.net;1
Portatil Acer TravelMate C311XMi;Acer;99.99;www.kuantokusta.net;1
Caixa Aerocool Baydream Case Silver;Aerocool;99.99;www.kuantokusta.net;1
Motherboard Asus P5AD2-E;Asus;99.99;www.kuantokusta.net;1
Motherboard Asus P5GDC Pro;Asus;99.99;www.kuantokusta.net;1
Motherboard Asus P5WD2-E Premium;Asus;99.99;www.kuantokusta.net;1
Portatil ASUS A6VM-Q756H;Asus;99.99;www.kuantokusta.net;1
Multifunções Brother DCP-115 C;Brother;99.99;www.kuantokusta.net;1
ATI Radeon 9250 128MB DVI + TV-out Club 3D;Club 3D;99.99;www.kuantokusta.net;1
ATI Radeon X300SE 256MB HYPER MEMORY DDR PCI-E DVI Club 3D;Club 3D;99.99;www.kuantokusta.net;1
Placa Gráfica EVGA GeForce 6800 XT 128Mb PCI-E;EVGA;99.99;www.kuantokusta.net;1
Disco 160GB SATA II Hitachi;Hitachi;99.99;www.kuantokusta.net;1
Impressora Multifunções HP Office Jet P5C 1410;HP;99.99;www.kuantokusta.net;1
Processador Intel Pentium D 805 2.6Ghz Dual Core 533Mhz 775 2X1MB
Box;Intel;99.99;www.kuantokusta.net;1
Gravador DVD LG GSA-H10B Black Bulk;LG;99.99;www.kuantokusta.net;1
Gravador DVD Samsung SH-S162A Black Bulk;Samsung;99.99;www.kuantokusta.net;1
Portatil Sony Vaio AR11B - Intel Dual Core T2300;Sony;1111.99;www.kuantokusta.net;0
PEN DRIVE 512MB Twinmos;Twinmos;1111.99;www.kuantokusta.net;0
Twinmos Pen 2Gb USB 2.0;Twinmos;1111.99;www.kuantokusta.net;0
Disco 250GB (8 Mb Cache) western Digital;Western Digital;1111.99;www.kuantokusta.net;0
DDRII 512MB-533MHZ;OEM;1111.99;www.kuantokusta.net;0

```

Figure 53: Kuantokusta Sample File

However, not every store is capable of generating this file, and we can use this extraction system to do just that. So we started by creating a FFD to extract the products of the "www.nuno-candeias.pt" on-line computer store, which aren't clients of Kuantokusta. In this store the products are displayed as a list, as we can see in figure 54. Their products are divided into categories and they don't display more than ten products on each page, thus creating a lot of pages to extract data.

The screenshot shows a web page with a navigation menu on the left, a main product list, and a login/register form on the right. The product list is titled 'Foram encontrados 14 produtos' and displays four items:

Product Name	Price
Hitachi 750 GB SATA II 300 16MB Buffer 7200 rpm ESGOTADO	€ 69,00
Samsung 160GB SATA II 300 8MB Buffer 7200 rpm	€ 39,00
Samsung 320GB 7200RPM SATA II 16MB Buffer	€ 43,50
Samsung 500 GB SATA II 300 7200 rpm 16 MB Buffer	€ 52,50

The right side of the page features a 'CLIENTES' section with a login form (Login, Password, Register) and a 'COMPRAS' section with a search bar and a 'NOTÍCIAS' section with a link to 'Loja da Tapada Candeias... [+]'.

Figure 54: Nuno Candeias Sample Product List

As Table Extraction Field is generic enough to extract the data that is not in tabular form, it was used for extraction in this case. The extraction was created upon a random product list page, since the objective is to extract data on all pages. It has one section hiding the whole page except the product list, and in that section the table field to extract the products. The columns in the Table Field extract the Name of the product, the brand, the price, the link for the product page. The figure 55.A shows a graphical representation of the extraction and the figure 55.B shows the position of the extraction elements highlighted on the rendered page.

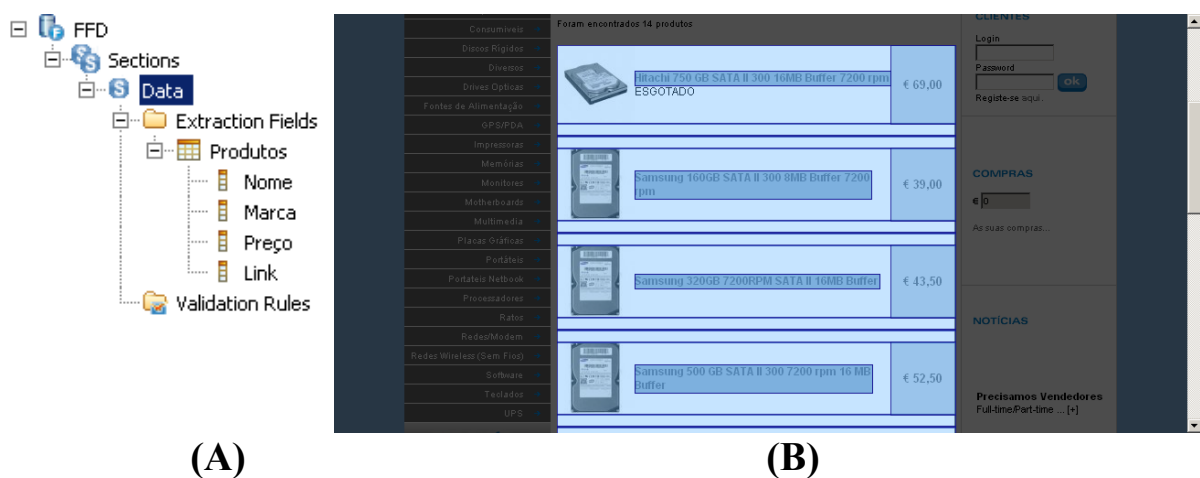


Figure 55: Visualizing the Extraction

In the definition of the extraction columns the validation with regular expressions of the price was also included. This will interrupt the extraction if the price isn't valid. The entire extraction process was created only with the suggestions of the editor, being all the extractions expressions automatically generated. The user didn't have to create, modify or correct any of them.

With the extraction created, a test of the extraction was performed in order to check if the values were valid and if there were no errors with the extraction. The results of this test are shown in figure 56.

Resulting Data Columns

FFD applied successfully.

Produtos.Marca	Produtos.Nome	Produtos.Preço	Produtos.Link
Hitachi	Hitachi 750 GB SATA II 300 16MB Buffer 7200 rpm	€ 69,00	detalhes_produto.php?id=2993
Samsung	Samsung 160GB SATA II 300 8MB Buffer 7200 rpm	€ 39,00	detalhes_produto.php?id=2190
Samsung	Samsung 320GB 7200RPM SATA II 16MB Buffer	€ 43,50	detalhes_produto.php?id=2684
Samsung	Samsung 500 GB SATA II 300 7200 rpm 16 MB Buffer	€ 52,50	detalhes_produto.php?id=2194
Samsung	Samsung 750 GB SATA II 300 7200 rpm 32 MB Buffer	€ 77,50	detalhes_produto.php?id=3053
Samsung	Samsung 1TB SATA II 300 7200 rpm 32 MB Buffer	€ 87,99	detalhes_produto.php?id=2196
Seagate	Seagate 160 GB SATAII 300 7200 rpm	€ 38,00	detalhes_produto.php?id=3063
Seagate	Seagate 320 GB SATAII 300 7200 rpm 16MB Buffer	€ 45,50	detalhes_produto.php?id=2210
Seagate	Seagate 500 GB SATAII 300 7200 rpm 16MB Buffer	€ 50,99	detalhes_produto.php?id=2192
Seagate	Seagate 500GB SATA II 32MB 7200RPM	€ 52,00	detalhes_produto.php?id=2173

Figure 56: Extraction First Test Results

After the first test, the extraction was also tested in other randomly chosen pages to ensure the extraction would not fail. The results are shown in Figure 57.

Resulting Data Columns

FFD applied successfully.

Produtos.Marca	Produtos.Nome	Produtos.Preço	Produtos.Link
Asrock	Asrock P4I65GV	€ 64,50	detalhes_produto.php?id=2247
Asus	Asus P5KPL-AM SE	€ 39,99	detalhes_produto.php?id=3401
Asus	Asus P5KPL-AM	€ 48,00	detalhes_produto.php?id=2720
Asus	Asus P5N73-AM	€ 47,50	detalhes_produto.php?id=3402
Asus	Asus P5N73-CM	€ 54,50	detalhes_produto.php?id=2291
Asus	Asus P5KPL 1600	€ 58,99	detalhes_produto.php?id=2673
Asus	Asus P5Q-VM	€ 99,00	detalhes_produto.php?id=2761
Asus	Asus P5Q-EM	€ 117,90	detalhes_produto.php?id=2936
Asus	Asus P5QPL-AM	€ 63,00	detalhes_produto.php?id=3260
Asus	Asus P5QL-CM	€ 77,99	detalhes_produto.php?id=2818

Resulting Data Columns

FFD applied successfully.

Produtos.Marca	Produtos.Nome	Produtos.Preço	Produtos.Link
Celeron	Celeron D E1400 2.0GHZ 775 FSB800 BOX	€ 47,00	detalhes_produto.php?id=3101
INTEL	INTEL DUALCORE E5200 2.5GHZ 775 FSB800 2MB BOX	€ 65,00	detalhes_produto.php?id=3028
INTEL	INTEL DUALCORE E5300 2.67GHZ 775 FSB800 2MB BOX	€ 77,00	detalhes_produto.php?id=2841
INTEL	INTEL DUALCORE E5400 2.7GHZ 775 FSB800 2MB BOX	€ 92,00	detalhes_produto.php?id=3435
INTEL	INTEL DUALCORE E6300 2.87GHZ 775 FSB1066 MHZ 2MB BOX	€ 94,00	detalhes_produto.php?id=3565
INTEL	INTEL C2D E7400 2.8GHZ 1066MHZ 3MB Cache BOX	€ 119,00	detalhes_produto.php?id=2402
NTEL	NTEL C2D E7500 2.93GHZ 1066MHZ 3MB Cache BOX	€ 140,00	detalhes_produto.php?id=2919
INTEL	INTEL Core2 QUAD CORE Q8200 1333MHZ 4MB BOX	€ 170,00	detalhes_produto.php?id=3301
INTEL	INTEL Core2 QUAD CORE Q8400 1333MHZ 4MB BOX	€ 177,00	detalhes_produto.php?id=3436
NTEL	NTEL Core2 QUAD CORE Q9400 1333 MHZ 6MB BOX	€ 215,00	detalhes_produto.php?id=3222

Figure 57: Extraction Test Results on other pages

With the extraction part created and checked we can define the data delivery part to end the extraction process. After creating a small program which receives an XML input with data columns and exports a text file in accordance with the KuntoKusta rules, we added a new data delivery with its SaveTo type defined to "External Program" and set the "External Program to execute" property to the path of the small program cited above (Figure 58).

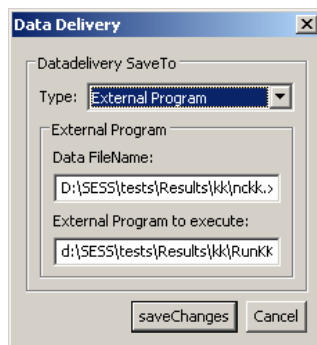


Figure 58: Delivery SaveTo type

This way the system will extract the products information out of the store webpage and delivers a text file which can be imported by the KuntoKusta system. Figure 59 shows a sample of the resulting file.

```
Hitachi 160GB SATA II 300 8MB Buffer 7200 rpm;Hitachi;€ 37,50;detalhes_produto.php?id=2174;1
Hitachi 500 GB SATA II 300 16MB Buffer 7200 rpm;Hitachi;€ 69,00;detalhes_produto.php?id=2993;1
Maxtor 160 GB SATAII 300 7200 rpm;Maxtor;€ 36,00;detalhes_produto.php?id=3063;1
Maxtor 250 GB SATAII 300 7200 rpm 8MB Buffer;Maxtor;€ 41,99;detalhes_produto.php?id=2199;1
Maxtor 500GB SATA II 32MB 7200RPM;Maxtor;€ 57,50;detalhes_produto.php?id=2173;1
Maxtor 1TB SATA II 32MB 7200RPM;Maxtor;€ 104,00;detalhes_produto.php?id=2717;1
Samsung 160GB SATA II 300 8MB Buffer 7200 rpm;Samsung;€ 36,50;detalhes_produto.php?id=2190;1
Samsung 320GB 7200RPM SATA II 16MB Buffer;Samsung;€ 46,00;detalhes_produto.php?id=2684;1
Samsung 500 GB SATA II 300 7200 rpm 16 MB Buffer;Samsung;€ 58,50;detalhes_produto.php?id=2194;1
Samsung 640 GB SATA II 300 7200 rpm 16 MB Buffer;Samsung;€ 69,99;detalhes_produto.php?id=3062;1
Hitachi 160GB SATA II 300 8MB Buffer 7200 rpm;Hitachi;€ 37,50;detalhes_produto.php?id=2174;1
Hitachi 500 GB SATA II 300 16MB Buffer 7200 rpm;Hitachi;€ 69,00;detalhes_produto.php?id=2993;1
Maxtor 160 GB SATAII 300 7200 rpm;Maxtor;€ 36,00;detalhes_produto.php?id=3063;1
Maxtor 250 GB SATAII 300 7200 rpm 8MB Buffer;Maxtor;€ 41,99;detalhes_produto.php?id=2199;1
Maxtor 500GB SATA II 32MB 7200RPM;Maxtor;€ 57,50;detalhes_produto.php?id=2173;1
Maxtor 1TB SATA II 32MB 7200RPM;Maxtor;€ 104,00;detalhes_produto.php?id=2717;1
Samsung 160GB SATA II 300 8MB Buffer 7200 rpm;Samsung;€ 36,50;detalhes_produto.php?id=2190;1
Samsung 320GB 7200RPM SATA II 16MB Buffer;Samsung;€ 46,00;detalhes_produto.php?id=2684;1
Samsung 500 GB SATA II 300 7200 rpm 16 MB Buffer;Samsung;€ 58,50;detalhes_produto.php?id=2194;1
Samsung 640 GB SATA II 300 7200 rpm 16 MB Buffer;Samsung;€ 69,99;detalhes_produto.php?id=3062;1
```

Figure 59: Sample of the Resulting File

The final delivery could also be made using the XSLT inner Process, thus eliminating the need for an external program.

5.5.3 News Websites

Another kind of popular and frequently visited websites are the news websites. Most of them don't present their information on a regular tabular form, thus making them a good subject for testing the extraction system. In this case the webpage chosen was the technological news site Engadget ("<http://www.engadget.com/>", Figure 60), which presents their news in a blog form. As in previous extractions, a section is created first surrounding the data that will be extracted. Although the data is arranged vertically without being in a tabular form it is

structured and a table field can be used to extract it. When defining the lines to the table, the entire area of a single news was selected for each line, this way it is possible to select a value to extract within it using a table column, which will be replicated for all the other news. The first table columns created are the most simple ones, the Title, the Abstract and the Permalink (the permanent URL to the news page), required only to point and click on them to have the system generate extraction expressions for them. The other columns are more complex since they require the use of regular expressions to extract them. The first is the "Posted By" column which we start by pointing and click on the "Posted By" text and letting the editor create the XPath expression for the element. Then in the Table Column Definition wizard with "your own Table Column Definition" selected and using the generated XPath we can select the "Text Content" in the XPath result option and add the regular expression "by (.*)", posted .*", which extracts the text between the "by" and the ", posted" getting the name of the poster. For the other more complex columns the strategy is the same, the only difference is the Regular Expression used. For instance, for the "Posted Time" column the regular expression is ".*, posted (.*)" and for the "Comments" it is "([0-9]*)".



Figure 60: Engadget Webpage

The figure 61.A shows the complete extraction tree, and the figure 61.B shows the extraction

graphical representation on the rendered page.

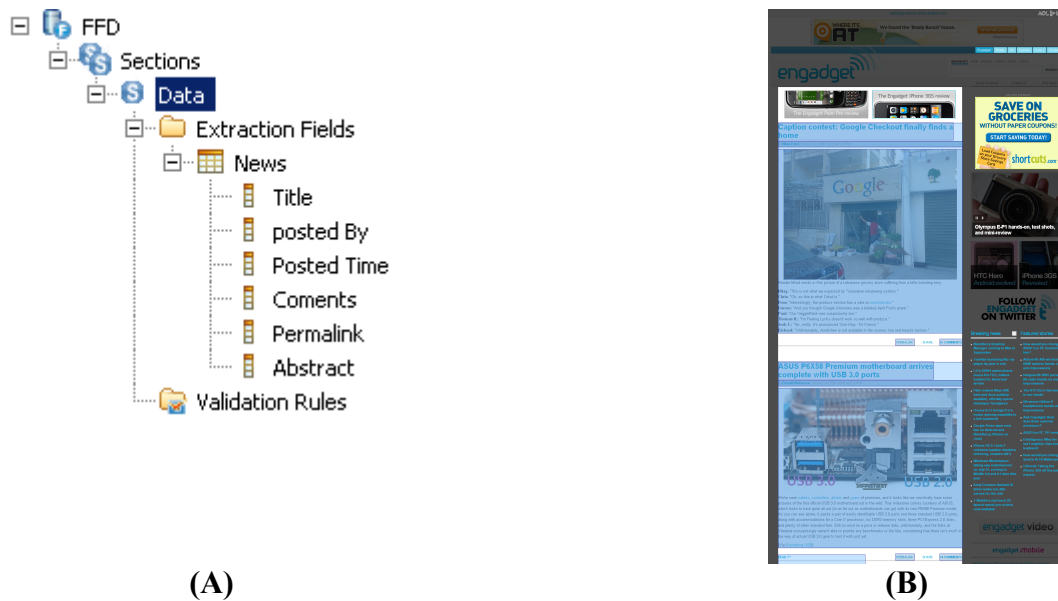


Figure 61: Visualizing the Extraction

In this extraction two single fields were also created to extract the minimum and maximum values of the displayed sub-set of news archive. These were used in a section validation which compared them in order to validate the number of news extracted.

With the extraction created we tested on the sample page, presenting the result in figure 62.

Resulting Data Columns

FFD applied successfully.

News.Abstract	News.Posted Time	News.Title	News.Permalink	News.Coments	News_posted By
While we're not too fond of the me...	Jul 21st 2009 at 8:23AM	Fuji F70 EXR compact superzoom spott...	http://www.engadget.com/20...	1	Tim Stevens
Just by looking at the Mobil ION 23...	Jul 21st 2009 at 7:56AM	Point of View Mobii netbook has Ion insi...	http://www.engadget.com/20...	4	Vladislav Savov
The noble goal of a \$100 laptop for...	Jul 21st 2009 at 7:14AM	Oddly humble Negroponce lists OLPC's F...	http://www.engadget.com/20...	10	Tim Stevens
Sure, you know Shure, the audio c...	Jul 21st 2009 at 6:42AM	Shure introduces three new sets of can...	http://www.engadget.com/20...	11	Tim Stevens
Usually when we freak out about t...	Jul 21st 2009 at 6:09AM	EATR robots claim to be vegetarian... s...	http://www.engadget.com/20...	10	Vladislav Savov
We don't know where, and we don't...	Jul 21st 2009 at 5:28AM	If Microsoft made a toaster...	http://www.engadget.com/20...	120	Thomas Ricker
Unless you're a member of the US...	Jul 21st 2009 at 4:31AM	Panasonic's Quick Power Dry hand drye...	http://www.engadget.com/20...	36	Thomas Ricker
When your netbook is too bulky to...	Jul 21st 2009 at 3:24AM	LG's Xnote Mini X120 Levi's Special Editi...	http://www.engadget.com/20...	16	Thomas Ricker
Now here's a smart marketing camp...	Jul 21st 2009 at 1:47AM	Toyota's giant solar flowers popping up...	http://www.engadget.com/20...	32	Ross Miller
It won't be quite the same as putt...	Jul 21st 2009 at 12:16AM	Nissan looking to give an inductive char...	http://www.engadget.com/20...	31	Ross Miller
#saleschart { border: 0px solid #3...	Jul 20th 2009 at 11:01PM	The Daily Roundup: here's what you mi...	http://www.engadget.com/20...	0	Ross Miller
What you are about to read is from...	Jul 20th 2009 at 9:59PM	Lawsuit alleges Apple conspired with M...	http://www.engadget.com/20...	92	Ross Miller
Heading to Comic-Con this year? IF...	Jul 20th 2009 at 8:49PM	Special edition Mad Catz Street Fighter...	http://www.engadget.com/20...	24	Ross Miller
Although it won't beat OnLive out t...	Jul 20th 2009 at 7:58PM	Israeli cable TV provider tries its hand a...	http://www.engadget.com/20...	34	Ross Miller
Looking to be in the envy of all you...	Jul 20th 2009 at 7:12PM	Xbox Live update preview program now...	http://www.engadget.com/20...	26	Ross Miller
Reader Milad sends in this picture o...	Jul 20th 2009 at 6:31PM	Cation contest: Google Checkout final...	http://www.engadget.com/20...	135	Nilav Patel

Figure 62: Engadget Extraction Results

This test shows that the extraction system is capable of extracting multiple information even though they are not displayed in a regular table form, being adaptable to different webpages structures and styles.

5.5.4 Sports Websites – World Rally Championship

The official website for the World Rally Championship ("www.wrc.com") has all the information about the championship, the teams, the drivers, the rallies, the stages, the stage's times and more. In this validation we created an extraction to obtain the driver's time on a particular stage on a rally. The figure 63 shows the webpage used to create the extraction.

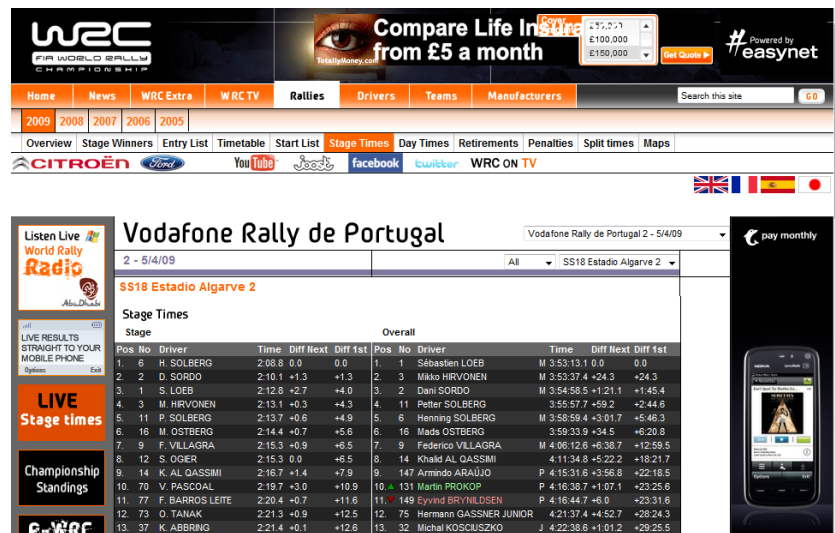


Figure 63: Stage Times

In this extraction we created two tables one for the Stage times and another for the Overall times. The figure 64.A shows the extraction tree and the figure 64.B shows the graphical representation of the extraction on the rendered page.



Figure 64: Visualizing the Extraction

The extraction was tested successfully and the results are presented in figure 65.

Resulting Data Columns

FFD applied successfully.

Overall.Time	Overall.Driver	Stage Number	Stage.No	Overall.No	Stage Name	Stage.Time	Stage.Driver
3:53:13.1	Sébastien LOEB	SS18	6	1	Estadio Algarve 2	2:08.8	H. SOLBERG
3:53:37.4	Mikko HIRVONEN		2	3		2:10.1	D. SORDO
3:54:58.5	Dani SORDO		1	2		2:12.8	S. LOEB
3:55:57.7	Petter SOLBERG		3	11		2:13.1	M. HIRVONEN
3:58:59.4	Henning SOLBERG		11	6		2:13.7	P. SOLBERG
3:59:33.9	Mads OSTBERG		16	16		2:14.4	M. OSTBERG
4:06:12.6	Federico VILLAGRA		9	9		2:15.3	F. VILLAGRA
4:11:34.8	Khalid AL QASSIMI		12	14		2:15.3	S. OGIER
4:15:31.6	Armindo ARAUJO		14	147		2:16.7	K. AL QASSIMI
4:16:38.7	Martin PROKOP		70	131		2:19.7	V. PASCOAL
4:16:44.7	Eyvind BRYNILDSEN		77	149		2:20.4	F. BARRIOS LEITE
4:21:37.4	Hermann GASSNER JUNIOR		73	75		2:21.3	O. TANAK
4:22:38.6	Michał KOŚCIEUSZKO		37	32		2:21.4	K. ABBRING
4:23:58.1	Vitor PASCOAL		36	70		2:21.4	H. WEIJS
4:24:03.3	Ricardo TEODÓSIO		131	74		2:22.1	M. PROKOP
4:26:09.2	Nasser AL ATTIAH		74	150		2:22.3	R. TEODÓSIO
4:26:59.3	Sébastien OGIER		147	12		2:22.8	A. ARAUJO
4:28:15.7	Pedro RODRIGUES		150	79		2:23.4	N. AL ATTIAH
4:28:45.1	Kevin ABBRING		39	37		2:23.5	A. BETTEGA
4:29:05.4	Ott TANAK		85	73		2:23.8	J. PETAK
4:29:11.9	Adrúzilo LOPES		64	68		2:24.1	M. SEMERAD
4:32:01.3	Hans WEIJS		68	36		2:24.5	A. LOPES
4:32:50.8	Ton WILLIAMS		148	62		2:24.6	G. SINGH GTII

Figure 65: Stage Times Extraction Results

Since this extraction, ends up extracting two different tables, the "Stage" table and the "Overall" table, when delivering the data it will perform two separate deliveries one for each table values.

5.5.5 Educational Information

The system was also used to extract educational information, in this case it was the curricular planning of the school subjects. A sample webpage(Figure 66) containing this information is located in: <http://www.di.fct.unl.pt/ficha/?ano=2009&lingua=pt&sem=1&codigo=8152>.

Interpretação e Compilação de Linguagens (2008/2009) - Departamento de Informática

Descrição

A disciplina tem como objectivo fornecer aos alunos um conhecimento sólido na área de concepção e implementação das linguagens de programação, efectuando um estudo sistemático dos conceitos sintácticos, semânticos e pragmáticos fundamentais subjacentes, que são centrais na ciência e na Engenharia Informática.

Os alunos adquirem a capacidade de analisar de forma objectiva as linguagens de programação existentes ou a existir, de compreender e saber utilizar as técnicas básicas de implementação de linguagens de programação, incluindo interpretadores, máquinas virtuais, e compiladores, e desenvolverão uma capacidade acrescida de aprender novas linguagens de programação de forma mais madura, assim como uma acrescida capacidade de concepção e desenvolvimento de software.

O estudo da semântica das linguagens será baseado essencialmente em técnicas operacionais, envolvendo o estudo de técnicas de interpretação e compilação dirigidas pela sintaxe, sendo cobertos os mecanismos encontrados na maior parte das linguagens funcionais, imperativas e centradas em objectos, incluindo os respectivos sistemas de tipos. Sempre que possível, ilustrar-se-ão os conceitos com exemplos retirados de linguagens de programação existentes (Pascal, Java, C, C++, ML, etc) ou "extintas" (Algol, Simula).

Objectivos

Saber

- Conhecer a arquitectura e as técnicas usadas na implementação de interpretadores e compiladores de linguagens de programação.
- Conhecer os elementos constituintes das linguagens de programação (valores, nomes, expressões, estado, abstracção funcional, encapsulamento) e a sua semântica operacional.
- Saber exprimir as construções das linguagens de programação modernas por composição dos elementos referidos no ponto anterior.
- Conhecer técnicas de especificação de sistemas de tipos e as propriedades por estes garantidas.
- Conhecer os limites da análise estática de programas, sabendo identificar problemas de análise indecidíveis.

Figure 66: Webpage Containing School Subjects information

For this one, several sections were created (one for each kind of data). The figure 67 shows the extraction tree.

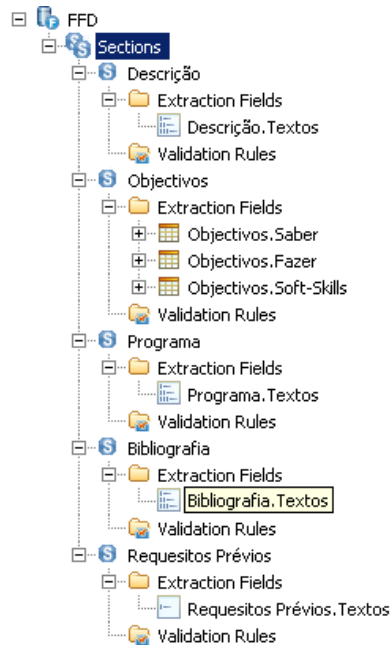


Figure 67: The Extraction Tree

Figure 68 shows the results of the extraction on the webpage containing the curricular information of the "Interpretação e Compilação de Linguagens" school subject.

Resulting Data Columns						
FFD applied successfully.						
Programa.Textos	Bibliografia.Textos	Requisitos Prévios.Textos	Descrição.Textos	Objectivos.Fazer.Textos	Objectivos.Soft-Skills.Textos	Objectivos.Saber.Textos
Princípios.	Notas de apoio à cadeira ...	Conhecimentos de especif...	A disciplina tem como obje...	Construir analisadores sin...	Participar num projecto de...	Conhecer a arquitectura e...
Expressividade das Língua...	"Concepts in Programming...		Os alunos adquirirão capa...	Representar e manipular ...	Raciocinar sobre sistemas ...	Conhecer os elementos co...
Interpretação de Programas	"The Study of Programmin...		O estudo da semântica da...	Descrever a semântica op...	Desenvolver a capacidade...	Saber exprimir as constru...
Valores e Expressões; Defi...	"Essentials of Programmin...			Conceber e implementar u...	Propôr e concretizar soluç...	Conhecer técnicas de esp...
Sistemas de Tipos	"Compiling for the .Net Co...			Conceber e implementar a...		Conhecer os limites da an...
Princípios, objectivos e limit...	"Modern Compiler Implem...					
Compilação de Programas						
Arquitectura de um compila...						

Figure 68: Extraction Results

5.5.6 Sports Websites – Soccer

An extraction using the site "www.futebol365.pt" containing news and information about soccer was also performed. In this particular case an extraction was created to gather players' information. The figure 69 shows the sample webpage chosen.



Figure 69: Soccer Player Details webpage

The figure 70.A shows the extraction tree with all the extraction fields used in the extraction and figure 70.B shows the graphical representation of the extraction on the rendered page.

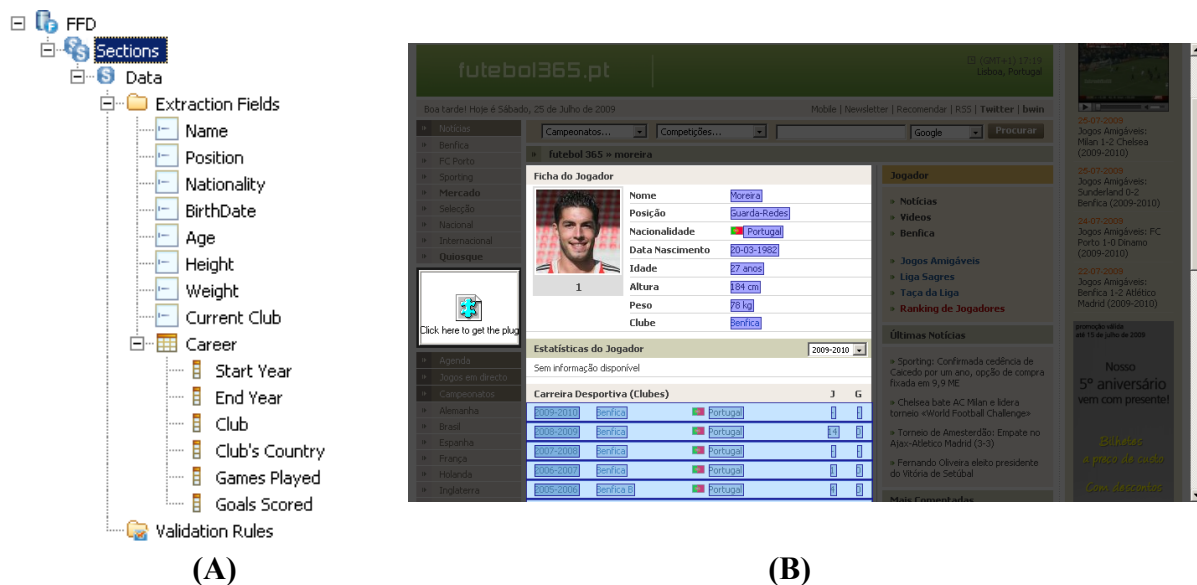


Figure 70: Visualizing the Extraction

After applying the extraction to the rendered webpage, we obtained the results presented in figure 71, showing the extraction correctness and ability to be used in all the player's details webpages.

Resulting Data Columns							
FFD applied successfully.							
BirthDate	Name	Position	Current Club	Weight	Age	Height	Nationality
20-03-1982	Moreira	Guarda-Redes	Benfica	78	27 anos	184	Portugal

Resulting Data Columns						
FFD applied successfully.						
Career.Games Played	Career.End Year	Career.Goals Scored	Career.Start Year	Career.Club	Career.Club's Country	
-	2010	-	2009	Benfica	Portugal	
14	2009	0	2008	Benfica	Portugal	
-	2008	-	2007	Benfica	Portugal	
1	2007	0	2006	Benfica	Portugal	
4	2006	0	2005	Benfica B	Portugal	
6	2006	0	2005	Benfica	Portugal	
15	2005	0	2004	Benfica	Portugal	
33	2004	0	2003	Benfica	Portugal	
31	2003	0	2002	Benfica	Portugal	
10	2002	0	2001	Benfica	Portugal	
-	2001	-	2000	Benfica	Portugal	
-	2000	-	1999	Benfica	Portugal	

Figure 71: Extraction Results

5.5.7 Electronic Stores – Amazon

An extraction to another electronic store was created to show the ability to extract data that isn't in regular tabular form. The webpage chosen was the Digital Cameras section on the popular website Amazon. Figure 72 shows the rendered webpage.

The screenshot shows the Amazon.co.uk website interface. At the top, there's a navigation bar with 'Hello, Sign in to get personalised recommendations', 'New Customer? Start here', and 'Great savings in our Summer Sale'. Below that, there's a search bar with 'Electronics' entered and a 'Basket' icon. The main content area is titled 'Electronics & Photo - Photography - Digital Cameras' and shows 'Showing 1 - 24 of 2,373 Results'. The first three items are:

- Panasonic Lumix TZ6 Digital Camera - Black (10.1MP, 12x Optical Zoom) 2.7" LCD**
Buy now: **£193.14**
12 Used & new from £193.14
In stock on July 27, 2009
Eligible for **FREE** Super Saver Delivery.
★★★★★ (33)
- Panasonic Lumix TZ7 Digital Camera - Black (10.1MP, 12x Optical Zoom) 3.0" LCD**
Buy now: **£264.99**
15 Used & new from £259.00
Get it by **Tuesday, Jul 28** if you order in the next **48 hours** and choose express delivery.
Eligible for **FREE** Super Saver Delivery.
★★★★★ (32)
- Panasonic DMCLZ10 Digital Camera - Silver (10.1MP, 5x Optical Zoom) 2.5 inch LCD**
Buy now: **£440.44** ~~£499.99~~
6 Used & new from £399.99
Get it by **Tuesday, Jul 28** if you order in the next **52 hours** and choose express delivery.
Eligible for **FREE** Super Saver Delivery.
★★★★★ (3)

Figure 72: Amazon Digital Cameras Section webpage

In this extraction the option was to only extract the model and price of the digital cameras. The figure 73 shows the graphical representation of the extraction on the rendered page.

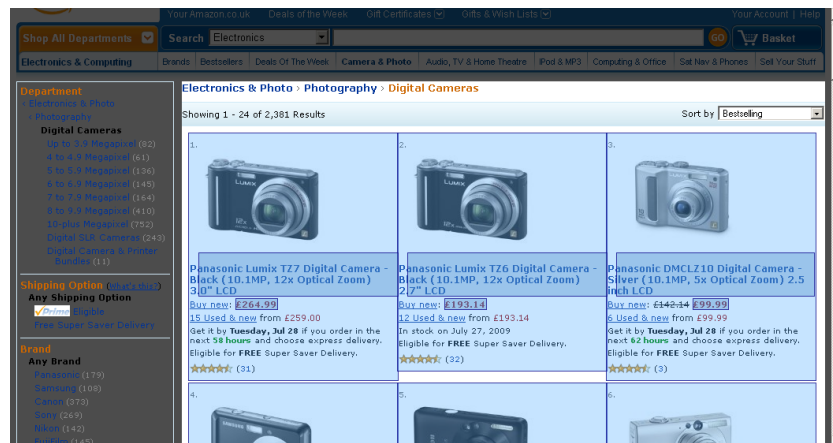


Figure 73: Visualizing the Extraction

The results obtained when applying the extraction on the sample webpage are shown in figure 74.

Resulting Data Columns

FFD applied successfully.

Products.Price	Products.Model1
£264.99	Panasonic Lumix TZ7 Digital Camera - Black (10.1MP, 12x Optical Zoom) 3.0" LCD
£193.14	Panasonic Lumix TZ6 Digital Camera - Black (10.1MP, 12x Optical Zoom) 2.7" LCD
£99.99	Panasonic DMCLZ10 Digital Camera - Silver (10.1MP, 5x Optical Zoom) 2.5 inch LCD
£68.99	Samsung ES15 Digital Camera - Black (10MP, 3x Optical Zoom) 2.5" LCD
£185.80	Canon Digital IXUS 100 IS Digital Camera - Black (12.1 MP, 3.0x Optical Zoom) 2.5" LCD
£249.99	Canon Digital IXUS 85 IS Compact Camera - Silver (10 MP, 3x Optical Zoom) 2.5" High Resolution PureColor LCD II
£284.50	Sony DSLR-A200K Digital SLR Camera + Zoom Lens Kit (18-70 mm F3.5-5.6)
£350.88	Canon PowerShot G10 14.7MP Digital Camera - 5x Optical Zoom, 3 inch PureColor LCD II Viewfinder - Black
£248.23	Canon PowerShot SX200 Digital Camera - Black (12.1 MP, 12x Optical Zoom) 3.0" LCD
£99.00	Sony Cyber-shot DSCS9505 - Silver (10.1 MP, 4x Optical Zoom) 2.7" LCD
£183.90	Canon Digital IXUS 100 IS Digital Camera - Silver (12.1 MP, 3.0x Optical Zoom) 2.5" LCD
£133.27	Panasonic Lumix FS15 Digital Camera - Black (12.1MP, 5x Optical Zoom) 2.7" LCD
£469.00	Nikon D40 with AF-S DX Zoom-Nikkor 18-55mm f/3.5-5.6G ED II
£139.50	Panasonic Lumix FS7 Digital Camera - Black (10.1MP, 4x Optical Zoom) 2.7 inch LCD
£79.98	Sony CyberShot S930 Digital Camera - Black (10.1 MP, 3x Optical Zoom) 2.4" LCD
£166.74	Sony Cyber-shot DSCW270B Digital Camera - Black (12.1 MP, 5x Optical Zoom) 2.7" LCD
£162.89	Canon Digital IXUS 95 IS Digital Camera - Silver (10 MP, 3.0x Optical Zoom) 2.5" LCD
£129.21	Sony Cyber-shot DSCS950B - Black (10.1 MP, 4x Optical Zoom) 2.7" LCD
£83.02	Samsung ES55 Digital Camera - Silver (10MP, 3x Optical Zoom) 2.5" LCD
£207.70	Fujifilm Finepix F200EXR Digital Camera - Black (12MP, 5x Optical Zoom) 3" LCD
£54.98	Polaroid i735 Pink Digital Camera 7MP 3xOptical Zoom +2GB +Hard Case Bundle
£632.48	Canon EOS 500D Digital SLR Camera (incl. EF 18-55 mm IS Lens Kit)
£377.90	Nikon D60 Digital SLR Camera - Black (AF-S 18-55 DX II VR Lens Kit)
£137.80	Canon PowerShot A1100 IS Digital Camera - Silver (12.1 MP, 4x Optical Zoom) 2.5" LCD

Figure 74: Amazon Extraction Results


5.5.8 Validations Conclusion

The presented validations show the ability of the extraction system to be used in a variety of different web pages. These validations tested all the features implemented in the system.


Regarding the performance of the extraction system, the tested validations revealed that the

actual data extraction is very satisfactory and doesn't appear to be as much affected by the web page size as it is by the complexity of the extraction. However, as the used render was built for web page visualization and not for background processing, when performing an extraction the render has to create all the graphical appearance of the web page, slowing significantly down the extraction process., as long as 3 to 5 sec.

Chapter 6 - Conclusions and Future Work



This chapter draws final conclusions on the design and implementation of this thesis as well as presents future work activities.



6.1 Conclusions.....	106
6.2 Future Work.....	107

This chapter draws final conclusions on the design and implementation of this thesis and presents future work.

6.1 Conclusions

The extraction system, whose design and implementation is presented in this thesis, was built to be a generic, extensible and modern data extractor capable of supporting multiple data source types, particularly webpages. Being designed with a modular architecture this system can be easily extended and maintained. It also makes it easier to create modules that support different data source types, increasing its usability in different and various tasks or situations.

The specific extraction module for webpages was developed considering the evolution of the Web, allowing a modern rendering engine to interpret and build the webpage to perform the extraction on it, instead of being limited to just extracting data from HTML source files. This way it is able to support simple static HTML webpages as well as those dynamically composed from several sources. An analysis was made about the rendering engines available for Java integration (section 2.3), and thereafter the SWT browser component using the Mozilla engine was chosen. This was indeed a good choice among the available ones, since it is modern, it is in constant development, it is frequently used in several browsers and other software and has a great performance. However the SWT browser component operation is not in accordance with its own documentation. This results in an extra effort to create a wrapper class, surrounding the component, to simulate a similar behaviour to the described in its documentation. Furthermore, this work-around with its error detection system, led to a marked decrease of performance when the system encounters an error.

A new specific extraction language was created for this module to suit this data source and to take advantage of it being structured. We also had to take into consideration that the language had to extract any kind of data, so it had to have a good generic ability to specify and identify data. With these two goals in mind, we created the webpage extraction expression, used in this system, which is composed by an XPath expression (to take advantage of webpage structure), a boolean value to indicate the format of the XPath expression (if it is the markup text or only the elements text content) and a Regular expression (which provides concise and flexible means for identifying strings of text). After the tests made which are described in section 5.5, this expression proved to be very expressive (allowing it to identify precisely

which data to extract) and easy to interpret or create (since it is based on two standard and popular expression types). This specific extraction language is only a part of an unified language which controls every feature of this extraction system. This language is divided into three main components (Extraction, Transformation and Data Delivery), in accordance with the modules of the core system. It includes all the different extraction languages allowing only one per extraction without mixing their rules. It also contains the new Data Delivery language created to support the new module.

Although this project has an unified language and the extractor has all the described modules working, the same could not be achieved with the new graphical editor, which only supports the creation of webpage extractions and data deliveries. To create the semi-structured text files extraction and transformations the user will have to resort to the previous graphical editor.

During the implementation of this project it was discovered that the web page render, has a particular way to deal with frames. As they are separated documents, the render creates several separate DOM trees, one for each document. This wasn't the expected behaviour, since only one web page with all the framed documents integrated is displayed for the user. Furthermore, there is always one main web page that contains the frame element which incorporates another document, suggesting that the DOM tree of the new document would be attached to the frame element of the main page DOM tree. Since it has a different behaviour than what it would be expected, the extraction system is not able to support web pages with frames, only allowing the process of one document per extraction.

In general, the whole extraction system including the graphical editor works as expected, allowing the creation of extractions in many different webpages containing a variety of data types, like those shown in section 5.5.

6.2 Future Work

This section presents future activities for the work described in this thesis, for the improvement and extension of the capabilities of the extraction system. Regarding these activities the following points are suggested:

- 1) Since the Webpage Extraction Module does not support extracting data inside of

Frames or iFrames, it could be useful to include this feature and make it transparent for the user. This could be done by implementing multiple web page extractions per Extraction Process, like the multiple Data Deliveries which the system already includes. This can be done by changing the FFD Definition Language, to contain a new element called "Extractions" which will group several extraction definitions. This way the editor could mask it creating a second extraction specifically for the webpage inside the Frame or iFrame, without user intervention.

Another feature where the system could be extended is the supported data source types. As for now, it already supports Webpage and Semi-Structured Text File Extractions, however it can be expanded to allow the extraction on others, like the Portable Document Format (PDF) or Open XML Paper Specification (OpenXPS). Both of these standard file formats are essentially containers for representing digital content in a paper-like fashion. To develop these modules new specific extraction languages need to be created as well as a graphical editor which would allow the user to interact with the rendered document to create an extraction without needing to know the language technical details.

- 2) Currently the graphical extraction editor is only able to create Webpage Extractions and Data Deliveries, forcing the user to create Semi-Structured Text File Extractions as well as the Transformations in another editor. It would be practical to consolidate all these actions in a single Graphical Editor.


To integrate even further the extraction system, the File Retriever Engine should be extended to enable navigation through the Web. This would also require a declarative language in order to record the links to follow, the data to fill in forms and links to click.

- 3) The improvements of the Expression Generator Modules as well as the Expressions Classification Mechanism used for calculating the best Extraction Expression to present to the user would be an interesting feature. New Expression Generator Modules for specific webpages or types of webpages could also be added, making the system more adapted to a particular usage.


Another useful improvement would be the data transfer system between modules. This system is used to transfer the resulting values from the Extraction Module to the Transformation Module and from it to the Data Delivery Module. Currently the system core groups the results in string arrays, thus restricting the extraction to Textual Values. The system could be improved by extending these transfers to other types of data such as images, videos, flash, binaries, and other multimedia content. This way it would allow the Extraction System to be able to extract, transfer and deliver those types of data.

A new feature that would improve the extraction system is an automatic creator of extractions. Currently, the user must explicitly define the sections and all extraction fields. The Graphical Editor should propose to the user a complete Extraction with its sections and extractions fields. This automatically complete generated extraction would be created based on the sample webpage data, using two distinct processes. The first, would be by searching for well structured tables, and the second by searching the data for specific formats given by regular expressions. The data type and validation rules could also be automatically inferred, using regular expressions.

Chapter 7 - References



This chapter comprises all the bibliographic contents referenced throughout the report.



-
- [1] Ricardo Fortuna Raminhos
Extraction and Transformation of data from semi-structured text files using a declarative approach, 2007, MSc Thesis, Departamento de Informática – Faculdade de Ciências e Tecnologia/Universidade Nova de Lisboa.
- [2] H. Lie, B. Bos, C. Lilley, W3C
The text/css Media Type, 1998
<http://tools.ietf.org/html/rfc2318>
- [3] The World Wide Web Consortium (W3C)
<http://www.w3.org/>
- [4] Extensible Markup Language (XML)
<http://www.w3.org/XML/>
- [5] XHTML
<http://www.w3.org/MarkUp/>
- [6] MathML
<http://www.w3.org/Math/>
- [7] Scalable Vector Graphics (SVG)
<http://www.w3.org/Graphics/SVG/>
- [8] Document Object Model (DOM)
<http://www.w3.org/DOM/>
- [9] Brendan Eich
INNOVATORS OF THE NET: BRENDAN EICH AND JAVASCRIPT
http://wp.netscape.com/comprod/columns/techvision/innovators_be.html

- [10] David Flanagan
JavaScript: The Definitive Guide, 5th Edition, 2006
Publisher: O'Reilly
Print ISBN-13: 978-0-59-610199-2
- [11] Michael K. Bergman
The Deep Web: Surfacing Hidden Value, 2001
<http://hdl.handle.net/2027/spo.3336451.0007.104>
- [12] Jill H. Ellsworth, Matthew V. Ellsworth
The Internet Business Book, 1994
publicado por John Wiley & Sons, Inc.
- [13] Gruchawka, Steve
How-To Guide to the Deep Web for IT Professionals
<http://techdeepweb.com/>
- [14] Tim O'Reilly
What Is Web 2.0. O'Reilly Network, 2006.
<http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html>
- [15] “The Web as Platform” - Web 2.0 conference
5 a 7 de Outubro de 2004 no Hotel Nikko
<http://www.web2summit.com/web2con/>
- [16] Robert Baumgartner, Michal Ceresna, Gerald Ledermüller
DeepWeb Navigation in Web Data Extraction
<http://ieeexplore.ieee.org/iel5/10869/34212/01631550.pdf>
<http://www.springerlink.com/content/y2u7dwwru3apg9lg/>

-
- [17] Jussi Myllymaki
Effective Web data extraction with standard XML technologies, 2002
<http://www10.org/cdrom/papers/102/index.html>
- [18] Arnaud Sahuguet e Fabien Azavant
WysiWyg Web Wrapper Factory (W4F)
<http://db.cis.upenn.edu/DL/WWW8/www8.ps.gz>
- [19] Jonathan Crabtree, Scott Harker, e Val Tannen
The Information Integration System K2
<http://db.cis.upenn.edu/K2/K2.doc>
- [20] Ficstar
<http://www.ficstar.com/>
- [21] KnowleSys Software
<http://www.knowledsys.com/>
- [22] Fetch Technologies
<http://www.fetch.com/>
- [23] Link Web Extractor
http://www.linkws.com/br/marketing/extractor_apres.htm
- [24] LinkWS
<http://www.linkws.com/>
- [25] HTML Extractor
<http://www.freewr.com/freeware.php?download=html-extractor&lid=2799>

- [26] Iconico Data Extractor
<http://www.seoconsultants.com/tools/iconico/data-extractor/>

- [27] Peter Sharpe, Vidur Apparao, Lauren Wood
Document Object Model Range, 2000
<http://www.w3.org/TR/DOM-Level-2-Traversal-Range/ranges.html>

- [28] C.N.Medappa
JREx - The Java Browser Component
<http://jrex.mozdev.org/>

- [29] The Lobo Project
Lobo: Java Web Browser
<http://lobobrowser.org/index.jsp>

- [30] Dmitry Markman
WebKit & Java integration
<http://www.concord.org/~dmarkman/jws/webkit/>

- [31] JDesktop Integration Components
<https://jdic.dev.java.net/>