Hugo Henrique Amorim Viana

Automatic Information Retrieval through Text-Mining

The dissertation presented for obtaining the Master's Degree in Electrical Engineering and Computer Science, at Universidade Nova de Lisboa, Faculdade de Ciências e Tecnologia.

Advisors : José António Barata,Professor Auxiliar FCT-UNL

Júri:

President : Luís Rosa Jardim Gonçalves

Vogals : José António Barata de Oliveira

Vogals : Luís Domingos Ferreira Ribeiro

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2013

Automatic Information Retrieval through Text-Mining

"Jamais haverá ano novo se continuar a copiar os erros dos anos velhos.". Luis Vaz de Camões

Agradecimentos

Os agradecimentos são a única parte da dissertação feita com a utilização da primeira pessoa do singular. Por tal, sinto-me com total liberdade para expressar o que sinto, e em português.

Em primeiro lugar quero dar um especial agradecimento ao meu orientador, o Professor José Barata, pela confiança que depositou em mim. Um muito obrigado pela oportunidade concebida.

Quero dar um enorme agradecimento à Ana Correia, pelo estímulo, amizade, carinho, críticas, sugestões no decorrer nesta minha breve estadia em Bremen. Obrigado por teres estado sempre comigo nos momentos onde me senti mais desanimado. Muito Obrigado.

Quero agradecer a Luis Neto, Rui Lopes e João Santos que se mantiveram sempre em contacto comigo, dando-me sempre apoio.

Um especial parágrafo para o meu ilustre companheiro e amigo para todas a horas "Mohamed" Ali, por toda a ajuda que me deu durante a fase final da escrita da tese, mas também por o privilegio de o ter conhecido no meu percurso académico. Um Muito Obrigado Maricas !

Um enorme obrigado para a 'Fada do Lar' Rita Faroppa que me ajudou imenso na escrita da tese, muito obrigado. Tu és enorme .

Este parágrafo é dedicado a Telmo Ferraria e Diogo Rego por todos os momentos partilhados em toda a minha jornada acamadémica, estando sempre presente para me ajudar em diversas ocasiões, bem como em momentos pessoais no qual me encontrei em baixo. Muito Obrigado por tudo.

Um especial obrigado a todos os colegas que me acompanharam durante todo o curso e no qual formei grandes laços de amizade. Independentemente de tudo sei que posso

contar com eles assim como podem contar comigo, Gonçalo Domingues, Nuno "Comandos" Rodrigues, Ricardo "Sargento" Rodrigues, Ruben Santinhos, Flávio Páscoa, Carlota Maçaneiro, José Vieira, Fábio de Passos, Filipe Viegas, João Pecorelli e Francisco Ganhão.

Aos meus amigos fora do curso, que acreditaram em mim e me foram sempre dando toda a força, Tiago Camarão, Renato Pereira, Sandro Rocha e Tiago Rodrigues.

Um especial obrigado Márcio Augusto, por toda ajuda e apoio prestado, estiveste sempre disponível. Obrigado.

À minha família, Pai, Mãe, Irmã, Tias, Tios e Primos por tudo. Foram sempre a minha fonte de inspiração e os meus guias em momentos mais escuros, onde tudo parecia correr mal. Mãe por acreditares sempre em mim e me ajudares sempre em tudo. Tia Alzira por estar sempre ao meu lado dando-me forças. Pai, por fazeres sempre tudo para me ajudar. Um enorme obrigado.

I would like to say thanks at all my colleagues at ATB, in particular Sebastian Scholze and Oliver Kotte for the support.

Sumário

Hoje em dia, uma enorme quantidade de firmas/empresas na União Europeia podem ser catalogadas de PMEs, empregando uma grande parcela dos trabalhadores a nível Europeu. No entanto, as PMEs não têm suporte para estarem constantemente a implementar métodos novos, nem tão pouco ferramentas como parte dos seus hábitos empresariais através de processos de inovação, que como é sabido é o motor para manter um empresa competitiva neste ambiente globalizado no qual as PMEs estão sujeitas, em particular na sociedade de hoje. Esta tese fornece uma plataforma que juntamente com o projecto *ExtremeFactories* (EF), proporciona uma ajuda as PMEs com o intuito de as tornar mais competitivas por meios de uma ferramenta que permite agendar uma tarefa de pesquisas.

Nesta tese é apresentada uma plataforma de text-mining que tem a habilidade de agendar uma recolha de informação sobre palavras chave. De maneira a desenvolver esta plataforma foram tomadas várias decisões, nesse sentido uma delas que deve ser salientada, *Apache Lucene Core*[1] que garante uma eficiente framework para desenvolver aplicações para text-mining, que é altamente usada nesta tese

Keywords: Text-Mining, Stemmer,Clustering, Lucene.

[1] lucene.apache.org/core

Abstract

Nowadays, around a huge amount of firms in the European Union catalogued as Small and Medium Enterprises (SMEs), employ almost a great portion of the active workforce in Europe. Nonetheless, SMEs cannot afford implementing neither methods nor tools to systematically adapt innovation as a part of their business process. Innovation is the engine to be competitive in the globalized environment, especially in the current socio-economic situation. This thesis provides a platform that when integrated with *ExtremeFactories* (EF) project, aids SMEs to become more competitive by means of monitoring schedule functionality.

In this thesis a text-mining platform that possesses the ability to schedule a gathering information through keywords is presented. In order to develop the platform, several choices concerning the implementation have been made, in the sense that one of them requires particular emphasis is the framework, *Apache Lucene Core* [2] by supplying an efficient text-mining tool and it is highly used for the purpose of the thesis.

Keywords: Text-Mining, Stemmer, Clustering, Lucene.

[2] lucene.apache.org/core

Acronyms

CL *Computational Linguistics*

CSS *Cascade Style Sheet*

DR *Dimensionality Reduction*

EU-FP7 *European Commission - Seventh Framework Programme*

EF *Extreme Factories*

EM *Expectation Maximization*

HPIEW *High Precision Information Extraction Work Bench*

IF *Intermediate Form*

IS *Information Science*

KD *Knowledge Discovery*

KDD *Knowledge Discovery from Databases*

KDT *Knowledge Discovery from Text*

ML *Machine Learning*

NTBFs *New Technology Based Firms*

LSI *Latent semantic indexing*

PCA *Principal component analysis*

POS *Part of Speech*

SOA *Service Oriented Architecture*

SIGKDD *Special Interest Group on Knowledge Discovery in Databases*

SMEs *Small and medium enterprises*

RTD *Research and Technical Development*

TF-IDF *Term Frequency Inverse Document Frequency*

UML *Unified Modeling Languag*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Nowadays, nobody dispute the crucial roles the Small and Medium Enterprises (SMEs) play in the development and growth of our economy. Without exaggeration around a huge amount of firms in the European Union can be catalogued as SMEs (including hight-tech and industrial companies, professional service providers, etc.) employing almost a great portion of the active workforce in Europe, this quote attains a slightly idea about what is our economic society , "More than 21 million small and medium sized enterprises represent 99 per cent of all companies in the EU, demonstrating their massive importance to the economic recovery, writes European Commission vice-president"(Tajani 2013).

One of the main characteristics of these organizations is the lack of resources which is widely used to excuse the absence of systematic innovation or Research and Technical Development (RTD) processes within the organizations where these processes are normally triggered by exogenous factors such as the irruption of new more competitive products and actors in the market or the imposition of new standards and regulations. In general the term SMEs cannot afford implementing neither methods nor tools to systematically adapt innovation as part of their business processes, though innovation is the engine to be competitive in the globalized environment, especially in the current socio-economic situation. Such an assertion can be equally extrapolated to New Technology Based Firms (NTBFs) even though they might be organized as a result of an innovation

or RTD process.

However, many steps have been taken to improve competition between SMEs and Extreme Factories (EF) is one of these. EF is an European Commission - Seventh Framework Program (EU-FP7) funded project founded in 2011 and it involves a high number of SMEs (8), all of them being very representative of the industrial web of the participating countries. There is a WebSite where the entire project can be followed *http://www.extremefactories.eu*.

The EF consortium proposes the conception of a collaborative platform with semantic capabilities (by means of ontology modeling) that implements a radically new methodology for the adoption of a systematic innovation process in globally acting networked SMEs. Whereas, the concept of this thesis has been made as a complementary block of EF to aid it by providing a schedule monitoring capability. Nevertheless, a question emerges, how will be possible to improve the platform by means of this new feature ? The following examples will help explaining the envisaged approach and show how monitoring can be utilized for fostering innovation. For instance, mobile phones, are known for being an ever growing market and at any given moment a new mobile could appear with not only new features but also new prices (i.e. low prices), in that sense, this could be an attempt to monopolize the current market. In fact, rival SMEs, which dispute the same market, are getting fewer less profits than before. Another example could be a SMEs discovery of a new technological advantage, and due to improve its own process of building a car (for example), hence, it could produce more cars with less costs, consequently it improves its profits, and again the rival SMEs get even fewer less profits than before.As a consequence, there is a need to keep maintaining a healthy market, and the *monitoring functionality* is an effort to solve a few problems. According to this purpose *monitoring functionality* might be useful, on the one hand to monitor updated prices on the other hand to enhance technological advancements. At last but not least, this helps monitoring the stage of an innovation process, in the sense that it could guarantee better results for the final product.

1.2 Objectives

EF provides the possibility to add campaigns[1] as well as ideas[2] and therefore a *monitoring platform* that allows finding relevant information from external sources (websites, publications, etc) that need to be monitored for the platform as other potential source of innovation for creation of campaigns and inception of ideas. Hence, it has motivated a development of a new platform component to handle this situation.

The component platform proposed for this thesis is organized in two important parts . The first one has the potential to organize the resources (i.e PDF or Wepgage) in clusters, and then match them to specific text pattern, and the second is assuring that the keyword(s) are unique, in other words, the platform has to assess whether several words with the same meaning do or do not exist . For instance, Like and Likes have the same meaning ,therefore the platform must detect this similarity . This platform was developed to give a support to the Innovation Process in SMEs. However, why EF requires a new platform ? it is not trivial to understand why such a tool is helpful, but for a SMEs (company), you want to be always aware of the prices of competitors, as well as new regulations, new technology , and so on.

In summary, this platform has the objective to support the gathering new information about a particular keyword(s) on a specific resource[3], as explained before the platform also provides a feedback about this keyword (i.e. if there is already a keyword with the same meaning or not). A comprehensive way to describe is as follows: a user of the EF platform provides keyword(s), a resource to be monitored (for example a web page or another document source, like PDF files) and a schedule for the monitoring itself. Incorporating these inputs, the platform then takes care of gathering information from the given resources that match the given keyword(s) and notify the user about those matches consequently.

[1] Represents an innovation or a problem, which should be resolved through innovation

[2] Any idea or suggestion to develop or improve design,production,marketing or management of a product service

[3] Resource is defined as any kind of document source

1.3 Outline of the Thesis

This thesis is organized in five chapters. The first chapter gives the outline of the scope of the thesis. The second chapter supplies a valid state of the art about Text-Mining. The third chapter describes and explains all the choices that have been chosen for the best implementation. The fourth chapter provides an overview about the platform, for instance, the webpage views are explained in that chapter. Finally, chapter five gives the overall conclusions.

Chapter 2

State of the Art - Text Mining

Throughout the years a constantly growing world of technology, where hardware becomes cheaper and better through the days, new opportunities arise, that were previously unimaginable or simply not feasible, have been seen by society. For that reason, owners could now store more information than previously, as a consequence, it was becoming hard to process and understand this amount of information. Certainly, this situation requires somehow a new approach to make it easier, and *TextMining* arises. Researchers from a variety of communities have developed new methods to mine text. As a matter of fact three of them has been contributed largely for the development of those methods and need to be underlined : Knowledge Discovery from Databases (KDD), Information Science (IS), and Computational Linguistics (CL).

In addition, is important to realize and look deeper why such topic (textmining) is so important for today on. In fact, with the increasing possibility of storing information at a reasonable price whenever and wherever it is produced, companies now have the means to collect and track nearly every conceivable event that takes place, such as a weather phenomenon, an earthquake or a heart attack. Even more important it can store, track and search any decision taken, such as a parliamentary vote, the sale of stock options or the purchase of a product, such as milk or a vehicle. As a consequence to this a gap between creating the information or data and actually understanding it is created. Furthermore, there is an increasing shortage of capacity to process and interpret all this information. As a consequence of what has been previously said, there is a need for an

automatic or at least a semiautomatic processing and interpretation of data, to ultimately attain new information. Regarding the gathering of information through a resource, a problem related with how to find a pattern in a huge source arises (like database ,documents, etc), and now is not that easy to search for a pattern in a source and obtain good results.

According with the previous introduction the *TextMining* choice is not clear, a reasonable explanation for that is that textmining is a huge tool to calculate similarity of texts and for that reason it is vital to know textmining approaches. Later on in this chapter it will be explained not only text mining in a theoretical way but also a few concepts of text mining that will be needed to follow the chapter 4.

2.1 A step further into Text Mining

To begin , a straightforward definition of text mining, also known as text data mining (Hearst 1997) or knowledge discovery from textual databases (Feldman 1995a), refers generally to the process of extracting interesting and nontrivial patterns or knowledge from unstructured text documents. It can be viewed as an extension of data mining or knowledge discovery (KD) from (structured) databases (Feldman 1996b), (Simoudis 1996).

Further definitions include, for instance, (Feldman 1996b) defined text mining as "Determining the information of interest to any individual while minimizing the amount of search through irrelevant information". Feldman's definition aligns more closely to information retrieval (one of the first steps when are dealing with textual data, is the extraction of relevant and irrelevant documents from a collection) than text mining (where the objective is to uncover patterns that are novel, interesting, and understandable). Other researchers suggest that text mining is done in two steps: the first provides structure to the initial corpora and the second enables knowledge discovery (Nahm 2000a).

Hearst, along of years of survey has achieved many definitions about textmining , as a result of the constant growing of stored data. For instance a first approach Hearst argues that text data mining is characterized as finding novel words from textual data. In a subsequent work, the same author argues that "text mining is the discovery by computer

of new, previously unknown information, by automatically extracting information from different written resources"(Hearst 2013a) and that "a key element is the linking together of the extracted information together to form new facts or new hypotheses to be explored further by more conventional means of experimentation" (Hearst 2013b) .

Accordingly with the previously quoted definitions it is really clear that the text mining have got a new shape, in an effort to follow up a growth of the data stored.Meanwhile, a recent survey provides an accurate definition for the purpose of the thesis "text mining is the process of identifying novel, relevant and understandable patterns from a collection of texts. The subclass of information extraction (such as systems that identify people, places, and organizations from a given set of documents) is not text mining because the researcher knows in advance which facts the system should extract and thus the activity does not satisfy the novelty criterion." (Blake 2013). Therefore, is clear now that text-mining can be done text mining without information extraction , despite the fact of being unusual, for this reason for developing the monitoring platform both will be needed.

Although, it is tempting to align "text mining" with "data mining" , actually data mining is just one of the steps within the knowledge discovery process. In contrast, text mining is more akin to the entire knowledge discovery process that comprises selection, preprocessing, transformation, data mining, interpretation, and evaluation steps(Nahm 2000a) . Unlike the numeric, ordinal, and categorical data types that initially motivated the (KDD) community, text has inherent complexities such as synonymy and polysemy and cultural nuances, which means that the preprocessing and transformation steps have even more impact on the quality of the patterns produced using text than using nontext information resources.

As the most natural form of storing information is text, text mining is believed to have a commercial potential higher than data mining. Additionally, most of companies information is contained in text documents. Text mining, however, is also a much more complex task (than data mining) as it involves dealing with text data that is inherently unstructured and fuzzy. Text mining is a multidisciplinary field, involving information retrieval, text analysis, information extraction, clustering, categorization, visualization, database technology, machine learning (ML), and data mining, thus , some of multidisci-

plinary field will be review later throughout this chapter.(Srivastava and M. 2009)

Consequently, as explained before , there is two paradigms that is important to understand when handling searching by patterns, which are , textmining and information retrieval. The first one handles to uncover patterns in a understandable collection, and the second one aims to differentiate between relevant and irrelevant documents. Besides, this combination have got good results, in spite of introduce others. As consequence of that now providers no longer assume that a user can clearly define his or her information need or that there will be only a small set of relevant information sources. Instead, a text mining system is data driven in that the starting point is the collection of information and not the users specific searching need. Rather than provide the user with a ranked list of articles, a text mining system will allow a user to explore and interact with patterns of information from within a collection of potentially relevant documents. The goal of a text mining system is to aid a user identify meaningful patterns and learn about the information space related to the information need.(Blake 2013)

A slightly way to describe the process of textmining is that, systems first convert text into a collection of words in the preprocessing and transformation steps of the (KDD) process and then uses standard data mining techniques such as clustering, classification, and association rules in the datamining step. However, further on this chapter will review deeper this process .

Last but not least, it is very important to underline the importance of Dimensionality Reduction Techniques because handles with stemming of a word, indeed, it is highly used for the indexer as saving space for the files indexed. This is important to retain but it will be used in the application. For instance, if you were to create a matrix, where the cell (i,j) reflected the number of times that word i appeared in document j, then most of the cells would contain zero. Strategically reducing the dimensionality of the original sparse matrix can reduce noise and thus help subsequent algorithms to avoid overfitting the model to nuances in the original text collection (Law 2006). Dimensionality reduction techniques commonly used with text collections are described in the section Dimensionality Reduction Techniques.

2.2 Information Extraction Strategies

Information extraction systems generally take either a Machine Learning (ML) or a knowledgebased approach. Meanwhile, a question arises , what is a ML ? The name suggests something like artificial intelligence, and actually it is a branch of it, which is about the construction and study of systems that can learn from data. Robust Automated Production of Information Extraction Rules (RAPPER) [1] system, which is an example of ML. It learns rules from the complete information extraction task and works on semistructured text (Eikvil 1999).

One difficulty with learning approaches is that they require training examples. To overcome this challenge, researchers have developed interactive systems that semiautomate the training process where a user first annotates a subset of the resources; the system then proposes an extraction pattern and the information that will be extracted using the current pattern; and lastly the user validates the pattern. (Caruana 2000) has developed a semiautomatic system called High Precision Information Extraction WorkBench(HPIEW) to extracts several difficult fields form text remarks fields. In addition, it provides a user the ability to specify range checks between 0.1 and 1.0, which means that the matched value must be between these numbers. The rules produced were regular expressions and evaluated by randomly selecting a record within the rule set and comparing it with the Protein Data Bank (PDB)[2]. The system took an afternoon to extract information from a 5.200 PDB sample and revealed approximately no errors. (Day 1997) have developed a new set of integrated tools collectively called the Alembic Workbench, which aims to reduce the overhead associated with the corpus development. The system enables users to constrain the part of speech (such as noun and verb). It (which generates LISPlike rules) took between 1 and 3 hours to extract information from a 25,000 training sample.

(Riloff 1996) also recognized the effort involved in manually labeling textual resources and reduced the effort needed by requiring that the user provide only a relevance judgment rather than annotate individual words. Results showed no significant difference between correct, correct and duplicate, and missing extractions between systems with

[1] RAPIER is a bottomup inductive learning system for learning information extract rules

[2] The Protein Data Bank (PDB) is a repository for the 3D structural data of large biological molecules (Berman HM 2000)

different levels of annotation. Moreover, the less specific labels produced significantly fewer spurious extractions. Knowledgebased approaches are also able to extract information from a textual resource, such as keywords in *ExtremeFactories*.

2.3 Dimensionality Reduction Techniques

According with what has been mentioned, technologies advanced in data collection and storage capabilities during the past decades have led to an information overload in most sciences. Thus, the effectiveness of textmining is limited by the large volume of data, as well as its complex and noisy characteristics and this is a new challenge for researchers. However, Dimensionality Reduction (DR) can provide satisfactory results for both. Consequently, DR has become increasingly important due to the emergence of many data sets with a large number of features.

(Underhill 2007) ”DR is the process of transforming a large amount of data into a much smaller, less noisy representation that preserves important relationships from the original data”.In another way DR deals with the transformation of a high dimensional data set into a low dimensional space, while retaining most of the useful structure in the original data.An example application of dimensionality reduction with numerical vectors can be seen in Figure 2.1 . The underlying assumption for DR that the data points do not lie randomly in the highdimensional space, rather, there is a certain structure in the locations of the data points that can be exploited, and the useful information in high dimensional data can be summarized by a small number of attributes.

Reducing the dimensionality can reduce the noise in the original text collection and thus provide better patterns. With the text representation in place and the dimensionality reduction applied, the researcher analyzes the collection of texts. This section describes domainindependent analysis methods that have been developed .

2.3.1 Stemming

Stemming attempts to identify the root form of a term. Many words in the English language can be reduced/stemmed to their base form, for instance, ”differ”, ”different”, ”differing” and ”differs” would all be represented as ”differ” after stemming had been

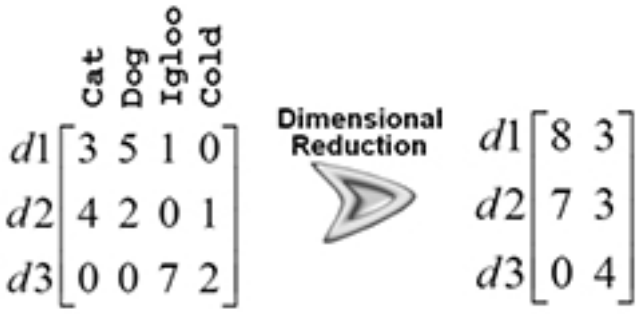

Figure 2.1: An example of dimensionality reduction.

.

applied. Besides, names are transformed into the stem by removing the " 's "(Kumar 2012). The variation "Peter's" in a sentence is reduced to "Pete" during the stemming process. One can characterize heuristic methods to identify the root form of a word as either light (terms are more likely to retain their original form) or heavy (terms are more likely to be stemmed).Therefore, it is hardly used in Information Retrieval systems to improve performance.Additionally, this operation reduces the number of terms in the information retrieval system, thus decreasing the size of the index files.

The Porter stemming algorithm, a light heuristic method, is frequently used for both text mining and information retrieval applications that employ English texts. Heuristic stemmers have also been developed for a variety of nonEnglish languages including Indonesian (Adriani 2007); (Korenius 2004); (Al-Shammari 2008);and the Portuguese, German, French, and Hungarian languages (Savoy 2006).In a subsequent work Porter(2001) has developed a rigorous system for defining stemming algorithms, which is called Snowball. It has been invented, in which the rules of stemming algorithms can be expressed in a natural way. Snowball is quite small, and can be learned by an experienced programmer in an hour or so (Porter 2013).

In particular, when are indexing files with *Lucene*[3] (See more on the chapter next chapter), that is highly used for the purpose of this work, which provides a stemmer for the users in order to gather more relevant results.

[3]Lucene is a fulltext document search library

2.3.2 Stop Words

Terms that occur frequently in text but have little discriminative power are called stop words. In addition, Part of speech (POS) tagging is the process of assigning the part of speech tag or other lexical class marker to each and every word in a sentence (Dr.Soman 2011) , for that reason it is used for identify stop words. There are several ways to identify stop words, such as removing the most frequent terms and POS which is removing words that have the following syntactic categories : determiners, prepositions, modals, auxiliaries, pronouns, and complementizers. In practice, text miners typically use one of the many stop word lists that are available online. Although domainspecific terms may also carry little meaning within the scope of a particular collection, such as the term 'baseball' in a collection of baseball texts, only a few text mining studies have considered domainspecific stop words (Blake 2001); (Ha-Thuc 2008).

2.3.2.1 Frequency Pruning

Each of the previously described methods was developed to reduce the time required to run analysis method, described in the next section. Additionally, Frequency Pruning is another method of feature reduction. It is to prune terms based on their frequency. The distribution of terms in a text collection conforms to a power law distribution ((Blake 2006)); thus, removing words that appear in fewer than n documents (where n is usually around 5) or more than m times (where m is usually a percentage of the number of documents) can significantly reduce the time required to run the analysis methods described in the following section.

Term Frequency Inverse Document Frequency (TFIDF) is a numerical statistic which reflects how important a word is to a document in a collection or corpus.(Salton G 2013) And so, TFIDF calculates values for each word in a document through an inverse proposition of the frequency of the word in a particular document to the percentage of documents the word appears in. Words with high TFIDF numbers imply a strong relationship with the document they appear in, suggesting that if that word was to appear in a query, the document could be of interest to the user. (Ramos) TFIDF is highly used for matching words in a query to documents that are relevant to that query.

2.3.3 Analysis Methods

With the text representation in place and the dimensionality reduction applied, the researcher analyzes the collection of texts. This section describes domainindependent analysis methods that have been developed for text and nontext datasets.

2.3.3.1 Classification

A classification algorithm aims to create a model (the classifier) that accurately maps from a set of features used to represent each document to an existing class. Thus, the goal of a textmining system is to create a classifier that, when provided with a set of features from a new document, accurately predicts the class of the new document. One wellused application of text classification is to assign controlled vocabulary terms to documents automatically. For example, several classifiers have been built for map journal abstracts to Medical Subject Headings.

Width	Height	Sides	Class
2	4	4	standing
3	6	4	standing
4	3	4	lying
7	8	3	standing
7	6	3	lying
2	9	4	standing
9	1	4	lying
10	2	3	lying

Figure 2.2: Example of Training Data (Ian H.Witten 2011)

Classification algorithms are typically supervised: researchers provide the algorithm with training examples that include both the correct class (also called classification or category) and the features used to represent each document (such as the vectorbased representation). The classification algorithm then constructs a model that best maps the given features to each class. When the training data Figure 2.2 only has two possible classes, a binary classifier is constructed; where there are more than two classes, a multi-

class classifier is required. A white box [4]classifier, such as decision trees Figure 2.3, has the desirable property that a user can see which features the classifier uses when making a prediction. In contrast, a black box model, such as a neural network, has no explanation capability. A model built by a good classification algorithm will include features that together effectively discriminate among the possible categories.

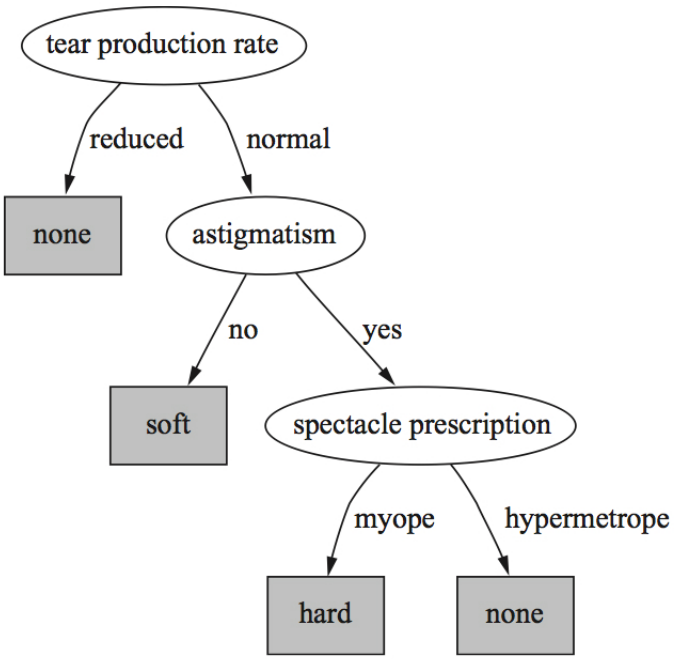

Figure 2.3: Example of a Decision tree (Ian H.Witten 2011)

The following classification algorithms, developed by the KDD community, have had much success when applied to text: Decision tree, the decision rules (Figure 2.4), and neural network algorithms (Figure 2.5) support vector machines (see (Sebastiani 2002), for a longer description of these algorithms). The most straightforward approach is the Naive Bayes algorithm that calculates the class probability of a new document using wordlevel prior probabilities from a training set (Figure 2.6).

Several researchers have compared the accuracy of classifiers built using different classification algorithms, work on automated classification of news stories (Yang 1999); (Yang 1997). Current evidence suggests that supportvector machines consistently pro-

[4]A white box, in contrast to a black box, is a subsystem whose internals can be viewed, but usually cannot be altered.

```
If    leaf condition = normal and
      stem condition = abnormal and
      stem cankers = below soil line and
      canker lesion color = brown
then
      diagnosis is rhizoctonia root rot

If    leaf malformation = absent and
      stem condition = abnormal and
      stem cankers = below soil line and
      canker lesion color = brown
then
      diagnosis is rhizoctonia root rot
```

Figure 2.4: Example of a Rule (Ian H.Witten 2011).

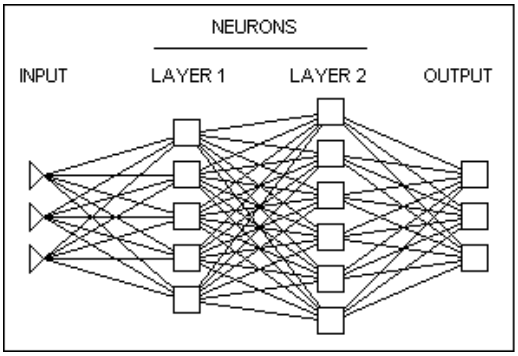

Figure 2.5: Example of a complicated network (Inc 2013)

vide the highest classification and that system performance generally increases with the number of examples in the training set ((Lewis 2004)).

Variations in how authors use natural language can introduce noise to the training data that makes it difficult for the algorithm to identify differentiating features. The large dimensionality of textual collections is also challenging for classification algorithms and can drastically increase running time. Researchers apply the dimensionality reduction techniques described in the previous section before running the classification algorithm to alleviate these challenges.(Blake 2013)

2.3.3.2 Clustering

The objective of a clustering algorithm is to group together documents such that each document group has a high degree of intraclass similarity and low degree of inter-

Sex	Age	Time Addr	ResStat	occup	Time Emp	Time Bank	House Exp	Decision
M	50	0.5	owner	unemploye	0	0	00145	reject
M	19	10	rent	labourer	0.8	0	00140	reject
F	52	15	owner	creative_	5.5	14	00000	accept
M	22	2.5	rent	creative_	2.6	0	00000	accept
M	29	13	owner	driver	0.5	0	00228	reject
F	16	0.3	owner	unemploye	0	01	00160	reject
M	23	11	owner	professio	0.5	01	00100	accept
F	27	3	owner	manager	2.8	01	00280	reject
F	19	5.4	owner	guard_etc	0.3	0	00080	reject
F	27	0.3	owner	manager	0.1	01	00272	reject
M	34	4	rent	guard_etc	8.5	07	00195	accept
M	20	1.3	rent	labourer	0.1	0	00140	reject
M	34	1.3	owner	guard_etc	0.1	0	00440	reject

Figure 2.6: Example of a Training Set (r Akeel Al-Attar 2013)

class similarity. In contrast to classification, clustering is typically unsupervised in that researchers provide a training set comprising only the features that represent a document, and not the known class.

Several clustering algorithms have been used with text, including hierarchical, portioning (such as the kmeans algorithm (MacQueen 1967)), mixture modeling (such as the Expectation Maximization (EM) algorithm (Dempster 1977)), nearest neighbor algorithm, and self organizing maps (see (Jain 1999) for more details on each algorithm). Although hierarchical clustering is statistically optimal, researchers typically use the kmeans algorithm to cluster text because kmeans is less computationally expensive than hierarchical clustering. The kmeans algorithm first selects k data points at random, which become the centroids of each cluster. The algorithm then assigns each data point to the closest centroid and recalculates the centroid as the average of all data points assigned to the cluster. The algorithm then repeats the process of assigning data points to the closest centroid and adjusting the centroid until there are no more data point reassignments or until the number of changes drops below a given threshold. The draw back of the k-means algorithm is that it is sensitive to the initial choice of centroids and that it is not statistically optimal. Text clustering has been used to identify topics that are then used for browsing documents in a collection (Cutting 1992). Mixture models such as a variation of the EM algorithm and latent Dirichlet allocation (Blei 2003) have been used to identify cluster authors and topics (Steyvers 2004), topics (Wang 2005a), and communi-

ties (Li 2008) and to disambiguate word senses (Boyd-Graber 2007). The topics produced from clustering can be visualized, such as with selforganizing maps, or shown as topics over time ((Havre 2002); (Wang 2005b)). For deeper details about the algorithms see the study made by (Ian H.Witten 2011).

2.3.3.3 Association Rules

Association rules capture cooccurrences between terms in a document collection. Such rules are often generated using the apriori algorithm ((Agrawal 1996)) that presents patterns to the user in the form: A to B, where A and B are sets of terms. The user must supply two parameters to the apriori algorithm: support (the prior probability that A and B cooccur in the data set, which reflects the prevalence of a rule) and confidence (the conditional probability of B given A, which reflects the strength of the implication between A and B).

The KDT (Knowledge Discovery from Text) system was one of the first attempts to apply association rules that were originally developed for numeric and categorical data to text ((Feldman 1995b)). The FACT system, also developed by Feldman, generates association rules based on the keywords assigned to a document ((Feldman 1996a)). In addition to the minimum support and confidence levels, users may specify semantic constraints on the keywords, where the constraints are sourced from the CIA's World Factbook. For example, a user could constrain the lefthand side of an association rule to be a country or a member of the Arab League.

(Ahonen 2 11) demonstrated that association rules generated over news stories could reveal useful phrases for subsequent analysis. The algorithm steps a window of size w over each sentence in the text, at each step recording the words that appear in the window. The system built by (Lent 1997) first calculates the distribution of word terms for each time period, and then allows a user to specify the shape of rules and the minimum and maximum window size, in addition to minimum levels of support and confidence. To improve the quality of association rules, the system imposes heuristics that separate words from different sentences and paragraphs before generating association rules. (Nahm 2000b) uses the Apriori algorithm, but their system extracts information

using a machine learning system called RIPPER.

(Arimura 2000) used a suffix tree to identify patterns in text, which was subsequently shown to be capable of extracting RNA [5] from full text documents. (Kawahara 2013) developed another system that works with document features to identify cooccurrences between authors and journals.

2.4 An explanation of process

Throughout the previous section the more relevant concepts about Text mining have been explained. Hence, there are now conditions to get an overview about the process. Regarding with the process can be divided in two phases: Text refining that transforms freeform text documents into a chosen *intermediate form*, and *knowledge distillation* that deduces patterns or knowledge from the intermediate form. Intermediate form (IF) can be semistructured as the conceptual graphic representation, or structured such as the relational data representation. Furthermore, IF can be documentbased wherein each entity represents a document, or concept based wherein each entity represents an object or concept of interests in a specific domain. Mining a documentbased IF deduces patterns and relationship across documents. Document clustering/visualization and categorization are examples of mining from a documentbased IF. Also mining a conceptbased IF derives pattern and relationship across objects or concepts. Data mining operations, such as predictive modeling and associative discovery, fall into this category. A documentbased IF can be transformed into a concept-based IF by realigning or extracting the relevant information according to the objects of interests in a specific domain. It follows that documentbased IF is usually domainindependent and conceptbased IF is domaindependent (Tan).

The Figure 2.7 is an example about what is a framework for text mining ,hence, now we can explain the process. Text refining converts unstructured text documents into an intermediate form (IF). IF can be documentbased or conceptbased. Knowledge distillation from a documentbased IF deduces patterns or knowledge across documents. A documentbased IF can be projected onto a conceptbased IF by extracting object infor-

[5] RNA Ribonucleic acid

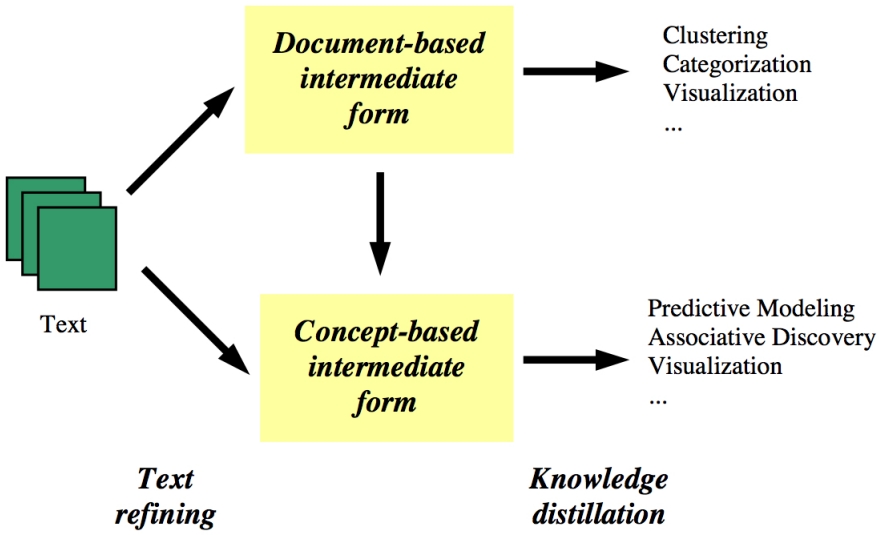

Figure 2.7: A text mining framework.(Tan)

mation relevant to a domain. Knowledge distillation from a conceptbased IF deduces patterns or knowledge across objects or concepts.

For instance , given a set of news articles, text refining first converts each document into a documentbased IF. One can then perform knowledge distillation on the documentbased IF for the purpose of organizing the articles, according to their content, for visualization and navigation purposes. For KD in a specific domain, the documentbased IF of the news articles can be projected onto a conceptbased IF depending on the task requirement. For example, one can extract information related to ”company” from the documentbased IF and form a company database. Knowledge distillation can then be performed on the company database (companybased IF) to derive companyrelated knowledge. This approach was used to developed the application, for that reason was needed to introduce an overview with a slightly explanation of the concepts that are behind.

Chapter 3

A Brief About The Implementation Choice

The EF project introduces the concept of a collaborative internet-based platform with semantic capabilities[1] (by means of ontology modeling) that implements a new methodology (based on Agile Methodologies[2]) for the adoption of a systematic innovation process in globally acting networked SMEs. EF will support SMEs to manage and implement the complex innovation process arisen in a networked environment, taking into account their internal and external links (i.e Web page or PDF files), by enabling an open multi-agent focused innovation (i.e. a customer/provider/supplier/employee focused innovation). Hence, it makes use of Service Oriented Architecture (SOA) that is a paradigm for combining distributed resources in different domains.

The concept purposed for this platform (*EF*) is to support a functionality for gathering new information through a resource. This platform is useful by giving support on the Innovation Process in the *EF*. Firstly, it provides a feature which is able to verify if a certain keyword(s) already exists and notifies the user which keyword(s) have similar meanings that were previously validated/inserted on the database. Note that the user is also capable to proceed or cancel the search with the specified keyword(s). Secondly, af-

[1] The Semantic Web is a collaborative movement led by the international standards body, the World Wide Web Consortium (W3C).[1] The standard promotes common data formats on the World Wide Web. (Berners-Lee 2001)

[2] Agile methodology is an alternative to traditional project management, typically used in software development. (CollabNet 2013)

ter validation, a scheduled survey performs a gathering information about these desired keyword(s).

Throughout this chapter it will be illustrated and justified the choices taken to develop the application. Whereby,it is going to be made an overview concerning the tools that were used to develop the application and how it could help to develop it.

3.1 Lucene

Lucene is a framework provided by Apache and it was developed for indexing documents and it is also a powerful searching engine .The following is a validated definition about Lucene (Lucene-java 2013) 'While suitable for any application which requires full text indexing and searching capability, Lucene has been widely recognized (**?**) for its utility in the implementation of Internet search engines and local, single-site searching'. Meanwhile Lucene can index any text-based information you like and then find it later based on various criteria searching. In spite of working only with text, there are a few other frameworks that allow indexing Word, PDF, XML or HTML documents. Lucene has a very flexible and powerful search capability that uses fuzzy logic to locate indexed items. Lucene is not overly complex. It provides a basic framework that you can use to build fullfeatured search into your web sites (Paul 2013). Nevertheless, the reason to achieve fast search is due to gather from index rather than text directly. (Apache 2013g)

Lucene also provides a good stemmer which is required and used farther on the application. The reason that is necessary to encompass Lucene with the platform is due to its extraordinary text search engine. Lucene has a high-performance, scalable, full-feature, open-source, and written in Java.

Regarding Lucene implementation it is necessary to be aware of a few basics concepts. Hereafter, those concepts that have to be perceived will be explained before the implementation.

- **Documents** A Document is the unit of search and index. Lucene has the indexer that is adding Documents to an IndexWriter [3]. Searching in Lucene involves retriev-

[3] An IndexWriter creates and maintains an index. (Apache 2013d)

ing Documents from an index via an IndexSearcher [4]. A Lucene Document doesn't necessarily have to be a document in the common English usage of the word. For example, if you're creating a Lucene index of a database table of users, then each user would be represented in the index as a Lucene Document.(Apache 2013a)

- **Fields** A Document consists of one or more Fields. A Field is simply a name-value pair. For example, a Field commonly found in applications is the title. In the case of a title Field, the field name is title and the value is the title of that content item.Indexing in Lucene thus involves creating Documents comprising of one or more Fields, and adding these Documents to an IndexWriter. (Apache 2013b)

- **Searching** Searching requires an index that had already been built. It involves creating a Query (usually via a QueryParser) and handing this Query to an IndexSearcher, which returns a list of Hits.(Apache 2013f)

- **Queries** Lucene has its own mini-language for performing searches. The Lucene query language allows the user to specify which field(s) to search on, which fields to give more weight to (boosting), the ability to perform boolean queries (AND, OR, NOT) and other functionality.(Apache 2013e)

Therefore, Lucene is not only used to find keywords in previously parsed documents but also to stem words.

3.2 Tika

Tika is an extensible Java-based framework for content analysis and detection, in other words it is a content extraction provided also by Apache. The Apache Tika toolkit detects and extracts metadata and structured text content from various documents using existing parser libraries (Apache 2013h). PDF files, Microsoft Office files (including Word, Excel, PowerPoint, and so on), images, text, binary formats, and more are a part of today's digital *lingua franca*, as well as the applications tasked to handle such formats (Zitting 2012).

[4]Implements search over a single IndexReader. (Apache 2013c)

Tika on this project is used to parse PDF files, in that sense it is a useful tool for text-mining processes.

3.3 Delving Lucene Process

This section provides a brief overview on how to Index files in Lucene, since the crucial concepts and ideas behind Lucene were explained previously. The following procedures are a standardized form in order to achieve better results with Lucene. This section is divided in two subsection which are:

- **Indexer Process** Providing an explanation about how Lucene does the indexing process.

- **Search Process** Providing an explanation about how Lucene does the search on the previous indexed file.

The Figure 3.1 illustrate a valid structure about how Lucene used the indexer for searcher, for further explanation see the next two subsections.

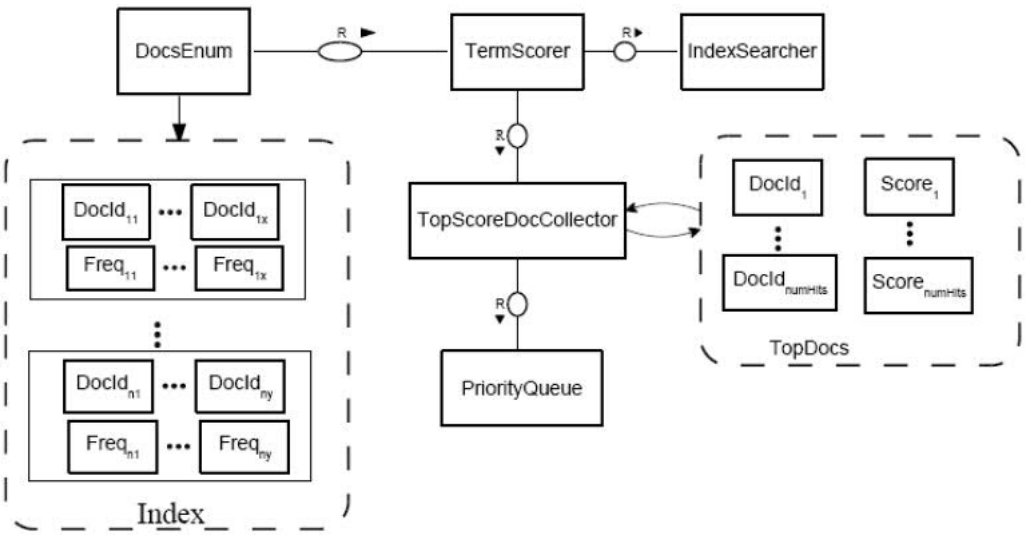

Figure 3.1: Lucene Proces indexing/searching (Zhou 2013)

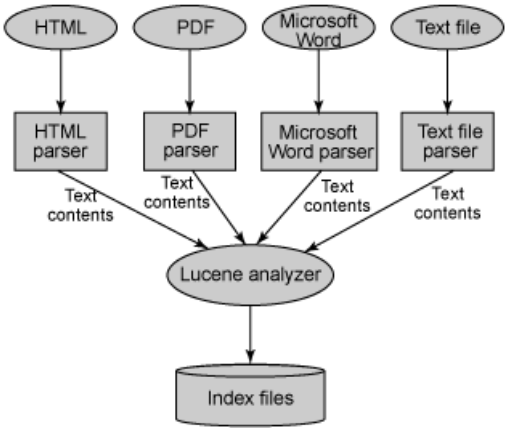

Figure 3.2: Indexing the Lucene architecture(Zhou 2013)

3.3.1 Indexer Process

Lucene has a particular architecture of indexing files as it is shown in the Figure 3.2. Lucene has no capability to use parse documents, in that sense it requires another framework and in this case Tika has been chosen to allow parsing different types of documents. Lucene can index any data that you can convert into a textual format. The focus for EF platform for instance was mainly on HTML and PDF's documents. Regarding this type of documents the first step is to extract the textual information from the documents, and afterwards, send the information to Lucene for indexing. The creation of an Index involves two different processes. In the first one, Lucene populates a document with various fields (key-value-pairs). Once a Lucene specific document has been created successfully, an Index Writer is required in order to create and maintain the Index for this document (and others of course).Regarding the strategy used by the Index Writer suitable for analyzing documents, Lucene provides the Analyzer since the Index Writer was correctly stored. In addition, to prevent concurrency, a Lock is used to avoid other Index Writers to open the same index directory.

The Figure 3.3 illustrates the process of indexing documents on Lucene.

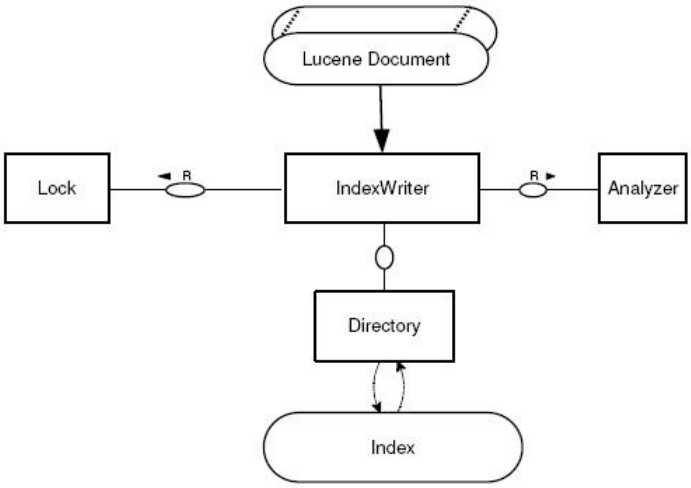

Figure 3.3: Indexing Process (Josiane 2013a)

A Lucene Document is a set of Fields the Figure 3.4 shows a Lucene Document. A Field encompasses a name and one or more values. The name is usually a word (String type) describing, for instance the field like content, path, name or date of creation are examples of field's names. Lucene Document could be used in three cases:

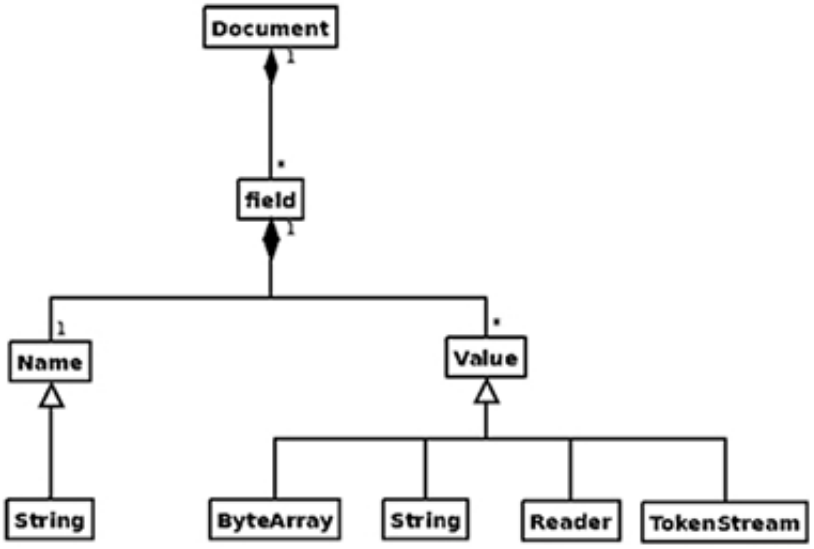

Figure 3.4: Lucene Document (Josiane 2013a)

1. *Logical Representation* of the original documents provided by the document parser, for example Tika parser.

2. There are other components stored, which are :

- Terms of each field: these build the terms dictionary
- Document's identification numbers (DocIds) : These are numbers that are automatically incremented when new Lucene Document is added.

3. *Represent a Result of a Query* : Field's values matching the query are gathered and displayed by the search application as result.

3.3.2 Search Process

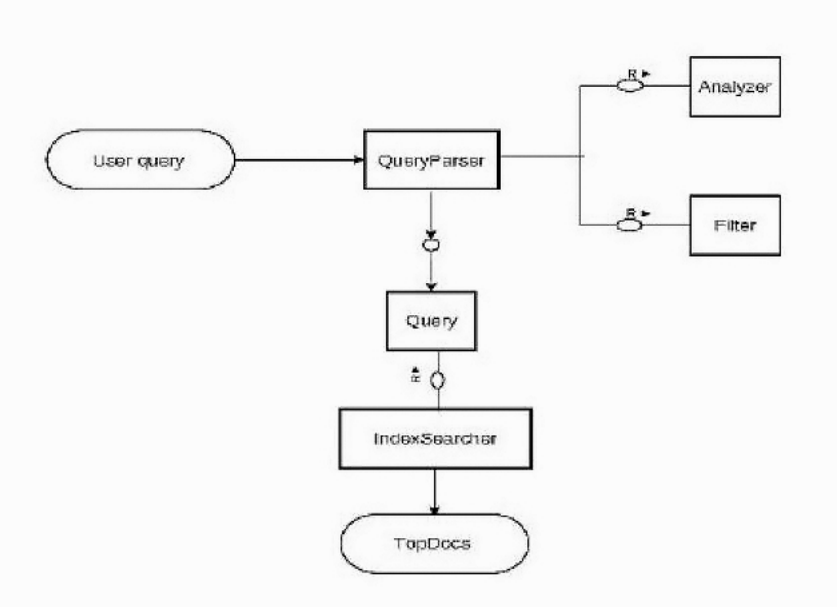

Figure 3.5: Search Process (Josiane 2013b)

In what concerns the searching process Lucene requires five components which are : Query, the QueryParser, IndexSearcher, Filter and Analyzer. The Figure 3.5 is an exhibition, that in fact tries to explain how all of those components are connected. The QueryParser has nothing to do with parser previously mentioned, in fact it is a class provided by the Lucene core. Following the Figure 3.5 since a *user query* is created it passes through the *query parser* process and then it becomes a Lucene Query. The Lucene Query is a group of one more *terms*[5], *symbols* [6] and *operators*[7]. The subsequent process is the IndexSearcher, and it is the main process. For being the main process, it followa all the

[5]A term is a word like 'spider' used int the Termquery
[6]symbol is a caracter like * or ? used in windcardQuery
[7]Examples for operators: 'AND', 'OR' or 'NOT'

process up of the retrieval information end . Whereby, it requires a valid Lucene Query supplied by the *query parser*, and afterwards it retrieves terms in the index that makes correspondence with the solicited query. As a result it supplies the top hits of that query (TopDocs). Finally, the *filter* which is used to select which documents should be allowed in search results.

3.4 Project Specification : Frameworks

The platform that was developed provides a specific front-end in form of HTML web pages tailored for common internet browsers. Certain platform functionality - especially for the fuzzy front end of the application will also be made available in user interface specifically adapted to smartphones. This view Layer has provided by EF.

- **Web Framework**

  The user interface is primarily driven by Apache Wicket, a component-based open source web application framework. Wicket uses plain Java POJOs and HTML, allowing a clear separation of mark up and logic. Wicket offers a wide range of predefined UI components as well as strong support for asynchronous interaction (i.e. AJAX) by leveraging JavaScript libraries such as jQuery. Its open structure and use of Java allows for easy development of custom reusable UI components which will facilitate the agile development approach aimed at in the EF project.

- **JavaScript Framework**

  For asynchronous interaction with the user and communication with the Extreme-Factories platform services, a JavaScript Framework is required. jQuery is a cross-browser JavaScript library designed to simplify the client-side scripting of HTML and fits to the requirements of the *ExtremeFactories* platform. It is released as an open source, allows extensions through plugins and has support for various mobile browsers.

- **CSS Framework**

  For the creation of a clear and intuitive user interface and graphical layout of the platform, which will mostly be web-based, the well-known and mature CSS (Cas-

cading Style Sheet) framework Twitter Bootstrap will be used. It contains HTML and CSS-based design templates for typography, forms, buttons, charts, navigation and other interface components, as well as optional JavaScript extensions. Moreover it also supports responsive design, which means the graphic design of web pages adjust dynamically, taking into account the characteristics of the device used (PC, tablet and mobile phone).

- **Application Server Framework**

  The JBoss applications server is a J2EE platform for developing and deploying enterprise Java applications, Web applications and services, and portals. J2EE allows the use of standardized modular components and enables the Java platform to handle many aspects of programming automatically. And so if one considers to the support of J2EE and the integration possibility of integration Java platform, for this reason, JBoss platform has to be seen as feasible.

- **Build Manager Framework**

  The Apache Maven is really a process of applying patterns to a build infrastructure in order to provide a coherent view of software projects.Maven aids to generate documentation of your project and create a Project Site, not only does it generates the site code, but it can also deploy the website on a server. A Project Object Model or POM is the fundamental unit of work in Maven. It is an XML file that contains information about the project and configuration details used by Maven to build the project. Maven is a versatile framework, for that reason it can build almost all the projects in java.

3.5 Project Specification : Development Environment

The software tools making up the *ExtremeFactories* development environment, together with their version, link and name of the task they are being used for,are listed in the Table 3.1.

Table 3.1: Development environment provide by *ExtremeFactories*

Function	Software Tool	Version	Link
Programming Language	Java	=1.6	`http://www.oracle.com/technetwork/java/`
Java Application Server	Jboss AS	=7.1.1	`http://www.jboss.org/jbossas/`
Build Manager	Apache Maven	=3.0	`http://maven.apache.org`
IDE	Eclipse	=3.7	`http://eclipse.org/`
CSS Framework	Twitter Bootstrap	=2.0	`http://twitter.github.com/bootstrap/`
JavaScript	jQuery + Mobile	= 1.7	`http://www.jquery.org/`
Web Framework	Apache Wicket	= 1.5.5	`http://wicket.apache.org/`

Chapter 4

Implementation

In this chapter it will be explained all about the developed platform implementation, as the name suggests. Afterwards, there will be introduced the choices that had been taken for the modularization by means of a Unified Modeling Language (UML).

The subsection 4.2 will explain the system architecture, dividing the platform into layers, and later on clarify every single layer, as well as, a proper description. Section 4.3 shows all of the views, which means the interaction options with a common user that allows to work with this application for example interaction page is explained as to insert, edit or remove a *Trigger*.

In order to be able to follow through the whole chapter, there is a term frequently used and for that motive it has to be previously introduced: *Trigger*. *Trigger* is a name for a class, which is responsible to store all the relevant information regarding a scheduled survey, for instance it stores the resource(i.e.WebPage), the schedule time, the keyword(s), and so on. Henceforth, this is the meaning for a *Trigger*.

4.1 Unified Modeling Language : UML

The entire section is dedicated to a description of the use cases. Hence, it was used to explain the solution proposed for the platform. The actors involved to develop the platform, were six actors: *Stemmer*, *ExtremeFactories User*, *Retriever*, *Timer Schedule*, *Resource* and *Manager*. All of the actors will be described over this section.

4.1.1 Actors Interaction

This subsection will explain the actors used for accomplish the objectives for this platform. The interactions between the actors as well as their dependencies will also be explained.

1. **ExtremeFactories User** This actor is a common user for the platform (i.e. customer/suplier). It is capable to create a new *Trigger*, by providing certain information for the *Trigger* as well as managing those *Triggers*. The management process allows the user to edit,add or remove a *Trigger*. Therefore, the user is allowed to change the keyword(s) or the resource(s) at any time and then restarts again the survey.

2. **Manager** This actor provides the whole interaction to the user, which means that it has to handle, not only with the events required for the implementation, but also the interaction with the current user. Besides that, it ensures smooth operation of the platform taking care of the validations (i.e. valid initial date), and moreover it assures the perfect synchronism between Timer Schedule, Resource, Stemmer and Retriever.

3. **Timer Schedule** This actor supplies the thread for charging the gathered information about a *Trigger*. With this purpose it triggers every time schedule and launches Retriever. It could also stop a *Trigger*.

4. **Resource** This actor is necessary to guarantee managing a Resource, which means, that it is responsible to store the information related with the path for a PDF file or a WebPage, for instance.

5. **Stemmer** This actor provides the functionality to guarantee whether a new keyword(s) is unique, in other words, there is no keyword(s) with this meaning on the Data Base.

6. **Retriever** This actor is concerning about the text-mining, for that reason it is the most important for the platform. Therefore, it supplies certain tools to gather information through a keyword(s) by a resource.

In addition, the Figure 4.1 illustrates what has said, as well as, the interaction between actors. In addition, the Figure 4.1 is an illustration about the relationship between the actors used for the application.

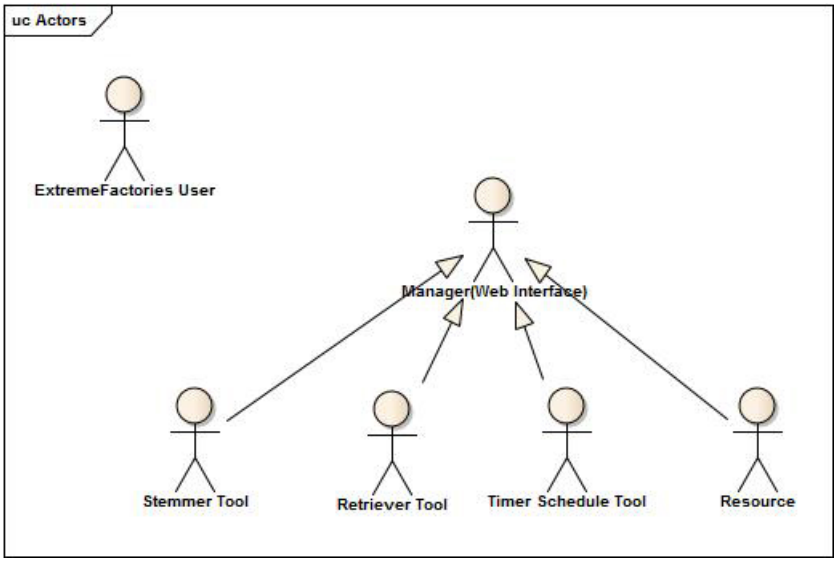

Figure 4.1: Use case : Actors on the System

4.1.2 Use case Diagram

This section provides the entire Use Cases for the platform developed. A table to describe every Use case is also provided, in order to make clear what purpose is used by the application.

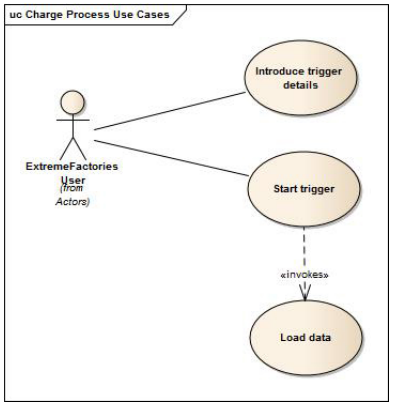

Figure 4.2: Use case : ExtremeFactories Charge Process

Table 4.1: Use case description : Introduce trigger details

Use case Name	**Introduce *Trigger* details**
short description	Create or edit a *Trigger*
trigger	When the user navigates to CreateEditPageTrigger
major inputs	Name, Resource, Keywords, Schedule(i.e. every 1 day, every 2 week), End/Start date
major outputs	Feedback for the trigger, i.e. if it was successful inserted, if not the feedback provides which fields have problems
major steps	1. The user enters and the details 2. Press save

Table 4.2: Use case description : Start *Trigger*

Use case Name	**Start *Trigger***
short description	Provides the Schedule information for the timer
trigger	When the user provided all of the correct information
major inputs	Schedule time
major outputs	*Trigger*
major steps	Validation of the required fields

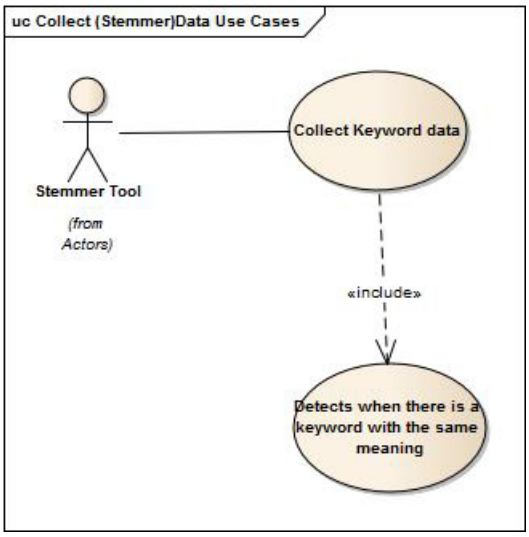

Figure 4.3: Use case : Stemmer

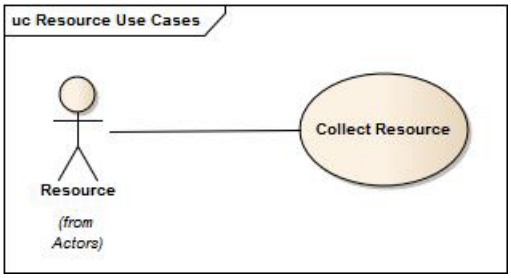

Figure 4.4: Use case : Resource

Table 4.3: Use case description : Load data

Use case Name	Load data
short description	Inserts all of the information on database
trigger	When the user provided all of the correct information
major inputs	*Trigger* fields
major outputs	Validated *Trigger*
major steps	1. Connecting to the database 2. Starting a transaction 3. Inserting the *Trigger* 4. Closing the transaction

Table 4.4: Use case description : Collect Keyword data

Use case Name	Collect Keyword data
short description	Collects all of the keywords inserted by the user
trigger	When a user saves a trigger the keywords are saved
major inputs	All of the keywords
major outputs	List of keywords
major steps	Process the keywords, for being validated.

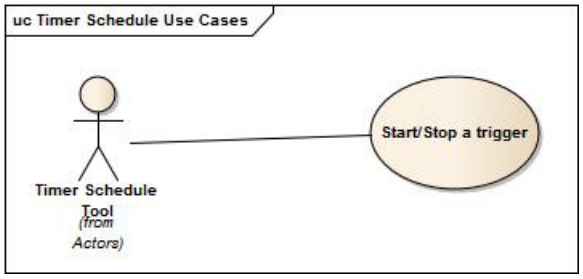

Figure 4.5: Use case : Timer Schedule

Steps could be for example checking if the trigger is running, try to stop the thread of the trigger or checking if this succeeds etc.

Table 4.5: Use case description : Detects when there is a keyword with the same meaning

Use case Name	Detects when there is a keyword with the same meaning
short description	Validates the keyword
trigger	Verifies if there is a trigger with the same meaning
major inputs	All of the keywords
major outputs	returns true if there is a trigger, false if not.
major steps	1. Connecting to the database 2. Fetch all the keyword(s) that are in the database. 3. Make the comparison between the new keyword(s) with the fetched keyword(s) that are on the database. 4. Closing the transaction

Table 4.6: Use case description : Collect Resource

Use case Name	Collect Resource
short description	Insert, edit or remove a resource
trigger	When the user create or edit a resource
major inputs	Details of the resource
major outputs	Resource
major steps	1. The user presses the link *edit resource* a popup message opens 2. In case the users wants to edit a resource, it must be selected if not just enters the resource 3. Press plus to add or minus to remove

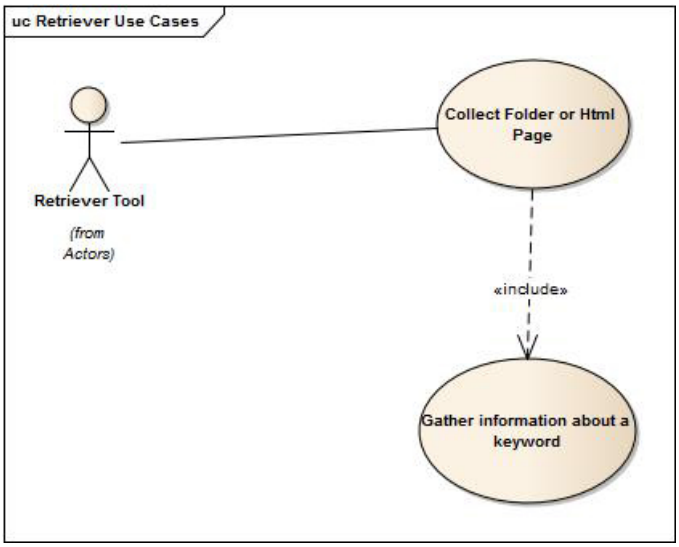

Figure 4.6: Diagram Class : Timer Service

Table 4.7: Use case description : Start/Stop a trigger

Use case Name	Start/Stop a trigger
short description	Manager a trigger
trigger	Since the trigger is successful inserted on the database
major inputs	Schedule time
major outputs	Returns a state for the *Trigger*
major steps	1. Verifying if the *Trigger* is running 2. Try to stop the thread of the *Trigger* 3. Verifying if the thread is successfully stopped

Table 4.8: Use case description : Collect Folder or Html Page

Use case Name	Collect Folder or Html Page
short description	Set the resource details to gather the information by keyword
trigger	When there is a expired *Trigger*
major inputs	Resource (Folder or Html)
major outputs	Update Lucene Document
major steps	Perform the directory to save the webpage/PDF to gather information

Table 4.9: Use case description : Gather information about a keyword

Use case Name	Gather information about a keyword
short description	Gather the information, text-mining part
trigger	Gather the score by a keyword on resource
major inputs	Resource(Folder or Html)
major outputs	Update Lucene Document
major steps	1. Stores the information in Lucene Documents 2. Analyze the Lucene, in order to find the keyword(s)

4.1.3 Class Diagram

This section describes and explains the main classes that were used to develop the platform, as well as, the relationships between them. By the purpose stated above, the five following classes have been developed:

1. **MonitoringMetaData**

   This class supplies information about whether a resource is in the last version or not. The process to perform this validation is based on comparing the old with the new file by checking the date. Hence, whether the new file has the same date, it means that the old file is the same version.

2. **Trigger**

   This classes describes a particular *Trigger*, and it provides the information concern-

ing the scheduled search.

3. **TriggerTimerService**

   This class is the thread for the trigger, in order to provide the schedule functionality for the platform.

4. **Retriever**

   This class provides the information for the class *MonitoringMetaData*, which means that it handles with information gathering about a keyword(s) by consulting a resource(i.e PDF). This was developed taking into consideration the process introduced in section 2.4, by means of *Document-Based Intermediate Form*.

5. **Stemmer**

   This class has the ability to verify if a keyword(s) with the same meaning already exists on the Data Base.

The Figures below made use of a diagram class to demonstrate the relations between them, and their attributes as well.

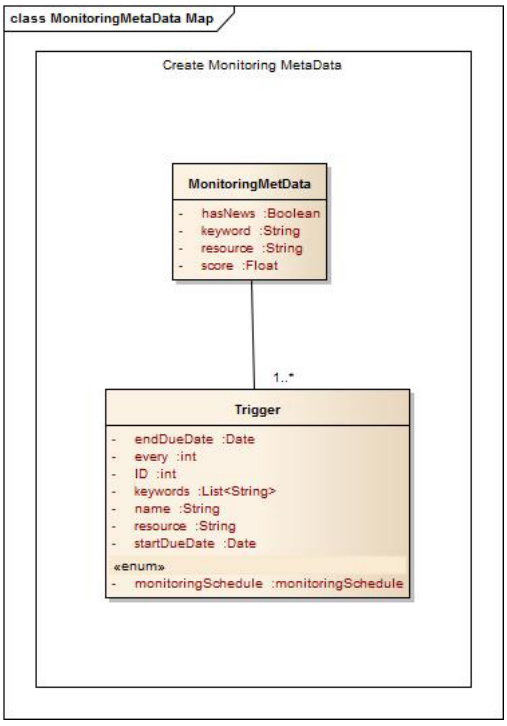

Figure 4.7: Diagram Class : Monitoring Metadata

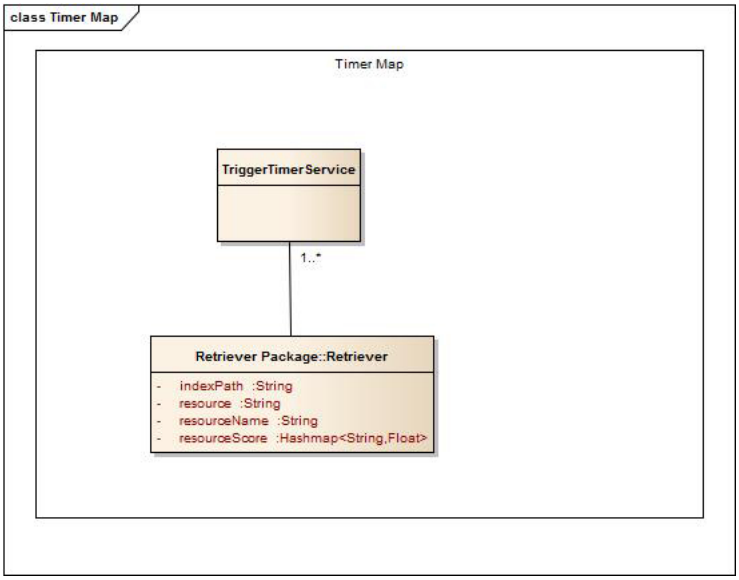

Figure 4.8: Diagram Class : Timer Schedule

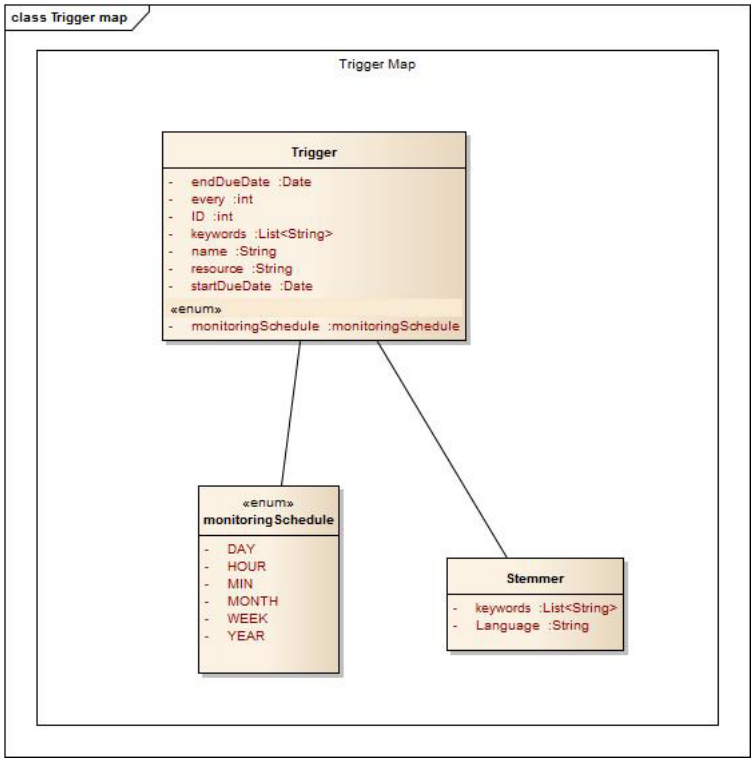

Figure 4.9: Diagram Class : Trigger

4.1.4 Sequence Diagram

The Sequence Diagram models show the collaboration of objects based on time sequence. It shows how the objects interact with others in a particular scenario of a use case. For that reason, Sequence Diagram was chosen to explain the whole process.

Therefore, hereafter an overview about the process will be given. First of all the entire user EF has to create a *Trigger* providing required fields which are name, resource, keyword(s) and the schedule time (i.e start/end date). Since the user has introduced the proper fields a new *Trigger* has to be created. In that sense, the platform is responsible to verify that each field is valid in order to be saved on the database. A particular step that needs to be underlined is the validation of the keyword(s) (i.e. if there is a keyword(s) with the same meaning), for the reason that the class Stemmer has to be triggered. Thereafter, the class TimerSchedule is triggered, resulting on a new thread for this *Trigger*, which means the class Retriever is triggered to gather the information by keyword(s) every expired time. Finally, the last step is to update the information every expired time, apart from that it also provides information every time it gets some updated file (for instance, a file has been changed) .

The whole process is properly illustrated in the Figure 4.10, by means of a Sequence Diagram. This Sequence Diagram tries to explain the entire process previously described to became this process clearer.

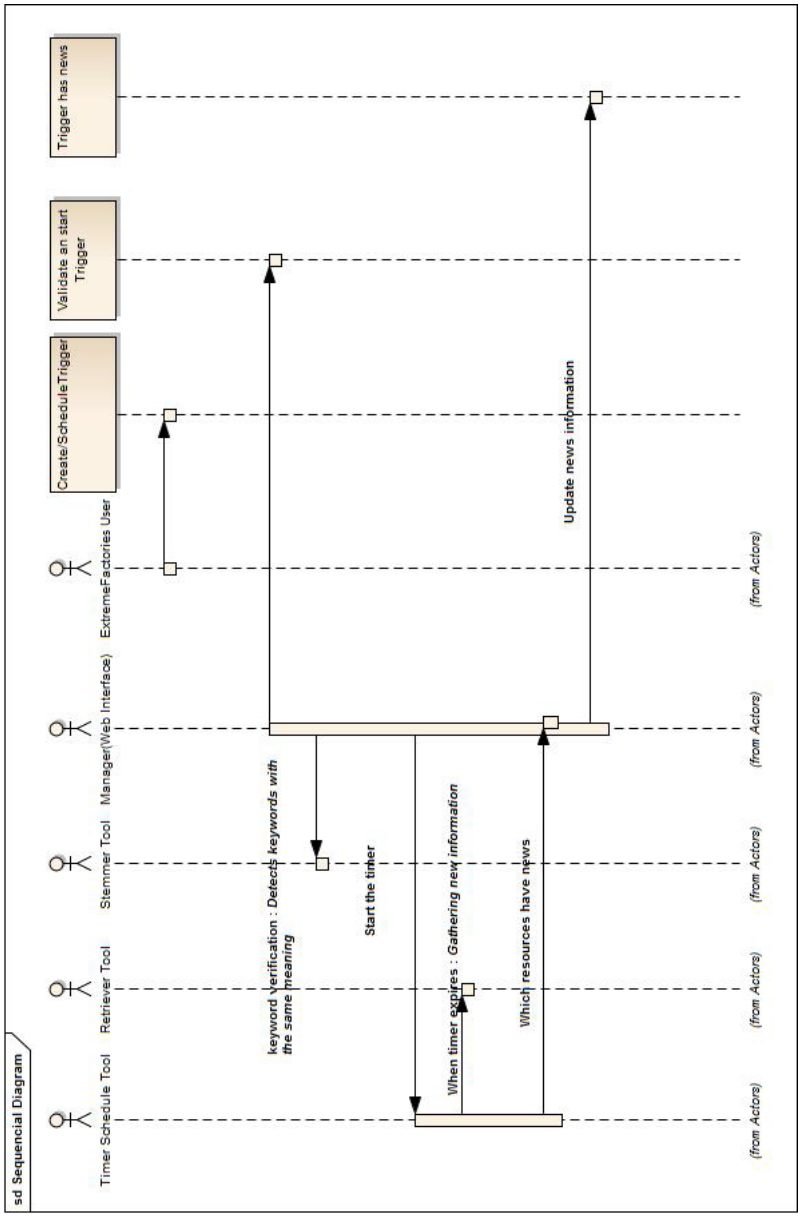

Figure 4.10: System Sequence Diagram

4.2 System Architecture

The platform is designed according to the Model-View-Controller architecture [1] .

As far as architecture is concerned, it is to be said that it has been divided in four layers. However, the layer that has the retriever and stemmer is to be considered as the most important for the project because it is where the text-mining and the stemming word are located and this is the leading purpose for the platform. Unlike the stemmer and retriever, the platform web view was provided by *ExtremeFactories*, and so far this has not been mentioned. Thus, the web view, for instance Cascading Style Sheets [2] , javascript as well as all the html code, likewise the communication and managing with the database have been provided by *ExtremeFactories*. Nevertheless, a couple of new features have been developed and integrated with the platform such as the popup window to add the resources. Actually, there was no need to develop the web view by its own, but there was a difficult study to do, which was to understand and integrate with the platform purpose for the thesis, instead.

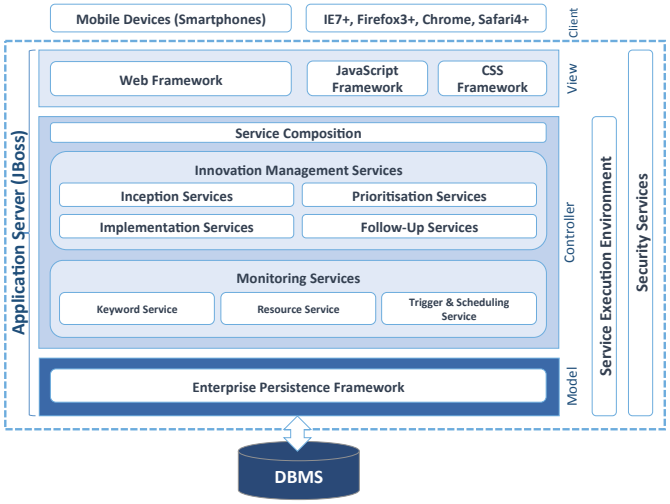

Figure 4.11: System Architecture

The Figure 4.11 illustrates an overview about the whole platform. With this in mind,

[1] Model-view-controller (MVC) is a software architecture pattern which separates the representation of information from the user's interaction with it.

[2] Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation semantics (the look and formatting) of a document written in a markup language.

the platform is divided in four layers , and starting with the top to the bottom :

- **Client** This layer is responsible for the relationship between the EF user's and the platform, which means that the insertion of data, as well as, schedule a gathering for a keyword is provided by this layer.
- **View** Provides all of the interaction with *ExtremeFactories* user.
- **Controller** The controller layer is divided in two parts. Firstly, it contains the whole management process provided by EF, since they have to be monitored. Secondly, it is the supplier that contains all implementation for monitoring, in that sense the Stemmer tool is located in Keyword Service and the Retriever is in Trigger & Scheduling Service. The major concern of developed work for the platform is located in this layer. Due to this, this layer supplies all tools/features to information gathering, by means of a keyword(s), but also it guarantees that a keyword(s) is unique, in other words, there are no two different keyword(s) with the same meaning. Process and validate all of the data provides by the *ExtremeFactories* user.
- **Model** This layer(Enterprise Persistence Framework) provides all of the interaction with the database, which means, it handles to adds,edits or removes a *Trigger* from the database. Besides, this database makes use of Framework *Hibernate*. Hibernate is an Object Relational Mapper (ORM) Framework, thus the underlying database can be changed without having to change the code of the platform. This creates, an effect, a "virtual object database" that can be used as a form within the programming language.

4.3 Website Description

This section is dedicated to explain the web interface. The relevant parts of the application with respect to the monitoring functionality are realized in mainly three different pages:

- **Main Page** Provides not only a view about which *Triggers* that are in course but also gives an interface to add, edit or remove a *Trigger*. In addition, it allows the user to

make a search for a specific *Trigger*.

- **Trigger Page** Provides an interface for the user to insert or edit a *Trigger*.
- **Similar Page** Show the *Triggers* that have keywords with the same meanings, allowing the user to save or not to save.

Taking into account the above mentioned pages, the following subsections will clarify and explain the necessary details and features for every single page.

4.3.1 Main Page

In this screen, the user could search for a *Trigger* by name, and moreover have a visualization about what are the existing *Triggers* in the database. The user is also allowed to select a certain *Trigger* from the list, in order to make an edition of that selected *Trigger*. For instance, if a project report of project is in the red-line to be delivered, the schedule time needs to be edited in order to control it more frequently. Besides, the user could also create a new *Trigger* by himself instead of making changes in an existing one. Moreover, there is a detail that needs to be underlined: every page provides the information about whenever a *Trigger* has got news. A notification about news is not updated in real time, but only every 10 seconds. This feature is not only used to notify the user whether the *Trigger* has news, but also a pop-up window that when clicked shows which resources have news by keyword(s). The Table 4.10 lists all the interaction that a user is allowed to and a description about its functionalities.

Table 4.10: Main Page

Name/Number	Definition
1	shows all of the *Triggers* in course with whole information
2	feature to notify when there is news
3	looks for a trigger in particular
Bottom Actions (edit)	to edit the selected *Trigger*
Bottom Create *Trigger*	to create a new *Trigger*
Bottom Delete *Triggers*	to delete all of the selected *Triggers*

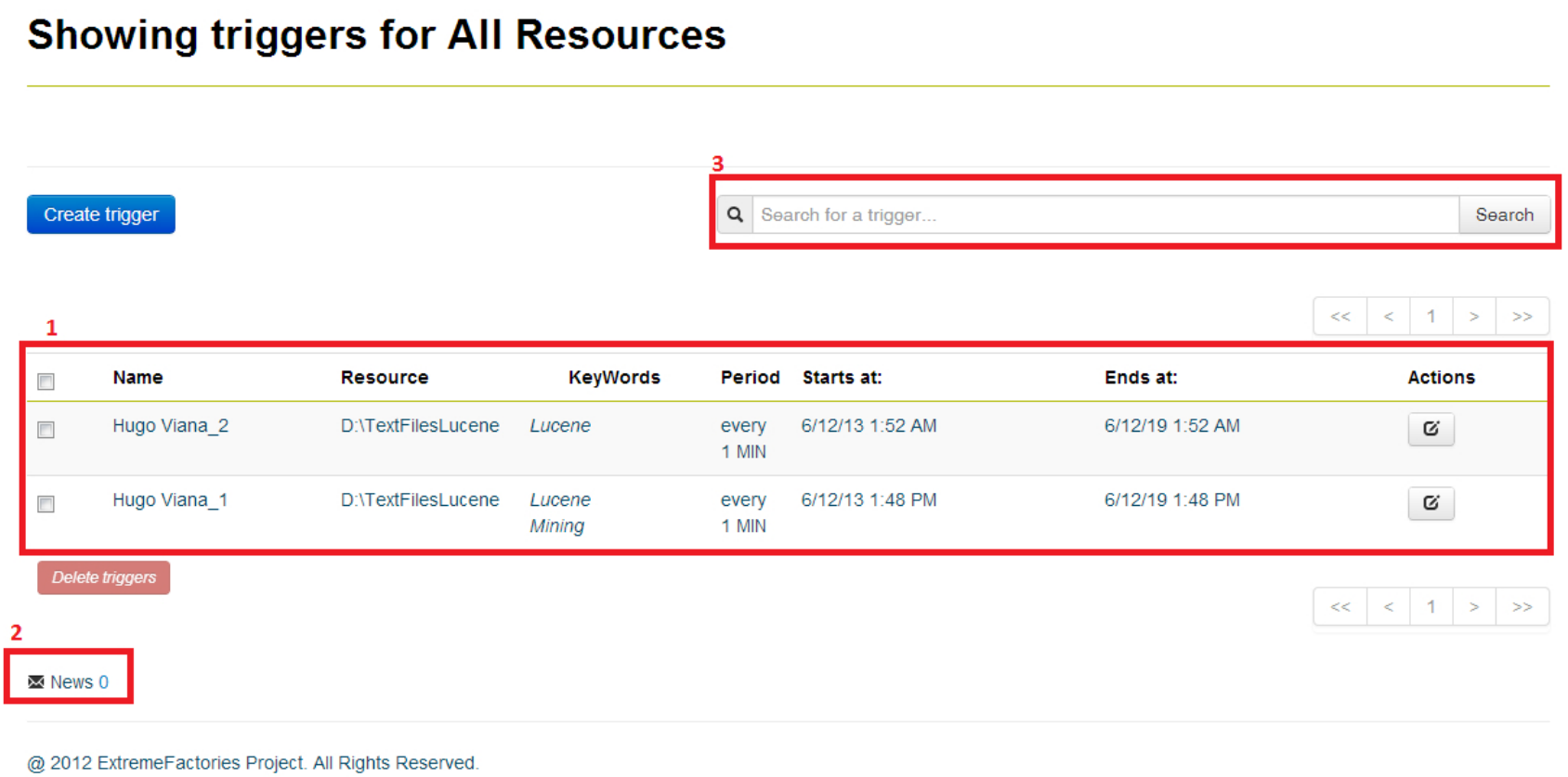

Figure 4.12: Main Page

Following the description of the interface given above and keeping in mind that a picture is worth a thousand words, Figure 4.12 shows the main user interface for the monitoring functionality. According with what has been described previously, Figure 4.13 is when the user clicks on the notification, indicating that there are a new or updated resources available.

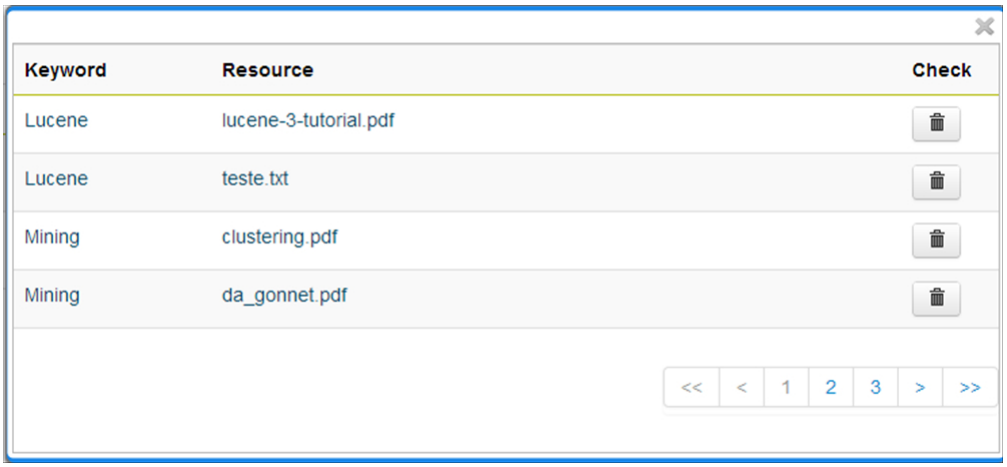

Figure 4.13: Popup to show the keywords with news

.

4.3.2 Create Trigger

This is the screen that allows the user either to edit or create a *Trigger*. This screen provides likewise a popup window where the user could add, edit or only view every single resource. This screen provides likewise a popup window where the user could add, edit or only view every single resource. It is the one which was most complex to implement compared to others, as it has to validate the whole data that is inserted by a common user. There are a few problems regarding the validation issues that need to be slightly explained. Validation has to be applied for several fields. Providing a start date for a *Trigger*, for instance that is later than the ending date of that *Trigger* should cause the validation to fail. And what is more validation should also detect if a user specifies a *Trigger* to be executed every week, but time frame between start and end date only last 4 days. The previous described cases are the worst for that application, in that sense not validating them could potentially break the timers and *Triggers* and finally the whole monitoring functionality. The Figure 4.16 is the HTML page that was rather described,

and the Table 4.11 is a description of every single interaction supplied for the user.

Table 4.11: Edit and Create Page

Name/Number	Definition
1	shows all of the resources , it could also add or edit
Bottom Save	Save and validate a trigger
Bottom Cancel	Cancel a trigger

The follow figure is an image of the popup window that handles with Resource. This window provides to a common user an interaction window to add, edit or remove a Resource(s). Besides, it could consult a list with information about the Resource that was already in the database. Once again, the Table 4.12 about each description is listed with description of every single interaction supplied for the user.

Table 4.12: Edit and Create Resource

Name/Number	Definition
1	Create a new resource
2	Edit or delete a resource
Bottom Plus	Add a selected resource
Bottom Minus	Delete a selected resource
Bottom Delete all Resources	Delete all resources

The last popup window provided for this application is shown in the Figure 4.15. It is only shown in case that the stemmer found another keyword(s) with the same meaning.

Nevertheless, this is only like a warning message for the user, in fact the user could store the keyword(s) without compromising anything. It supplies an option to consult which are the keyword(s) with the same meaning without saving the new keyword(s).

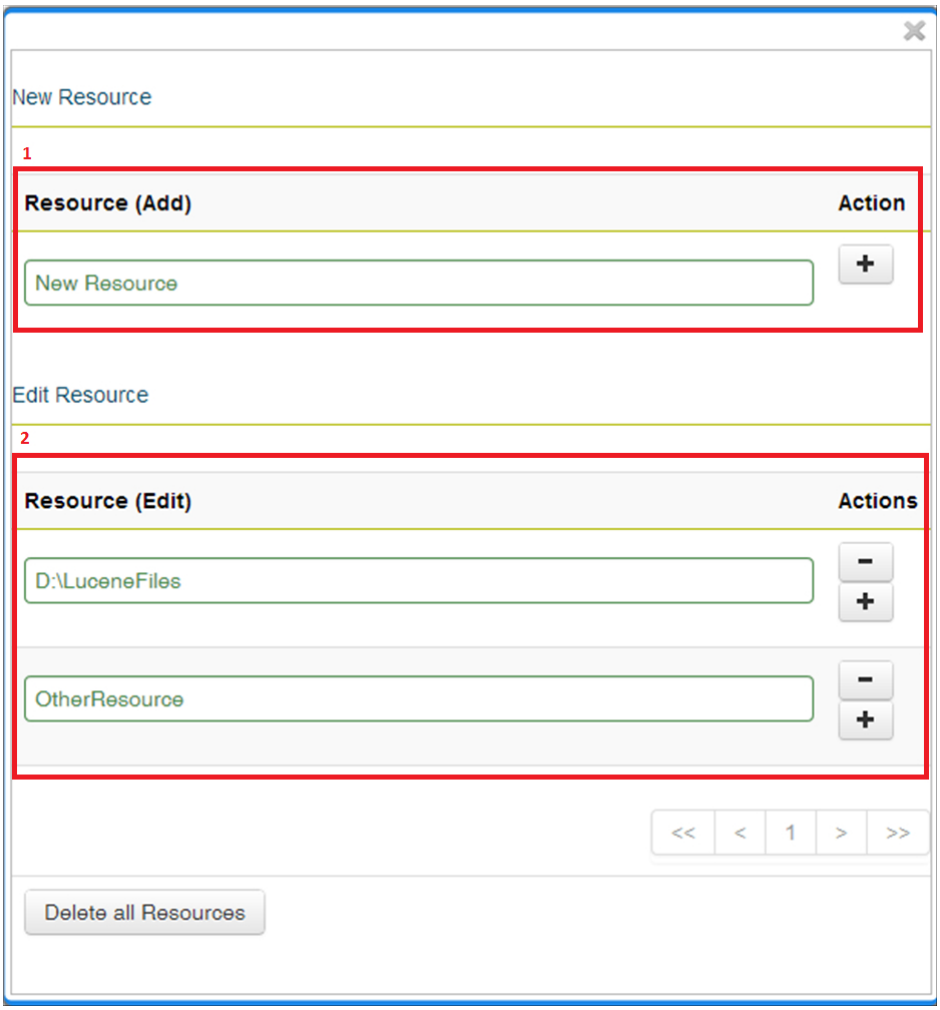

Figure 4.14: Popup to add, edit and show a resource

.

Figure 4.15: Popup to alert that is similar keywords

.

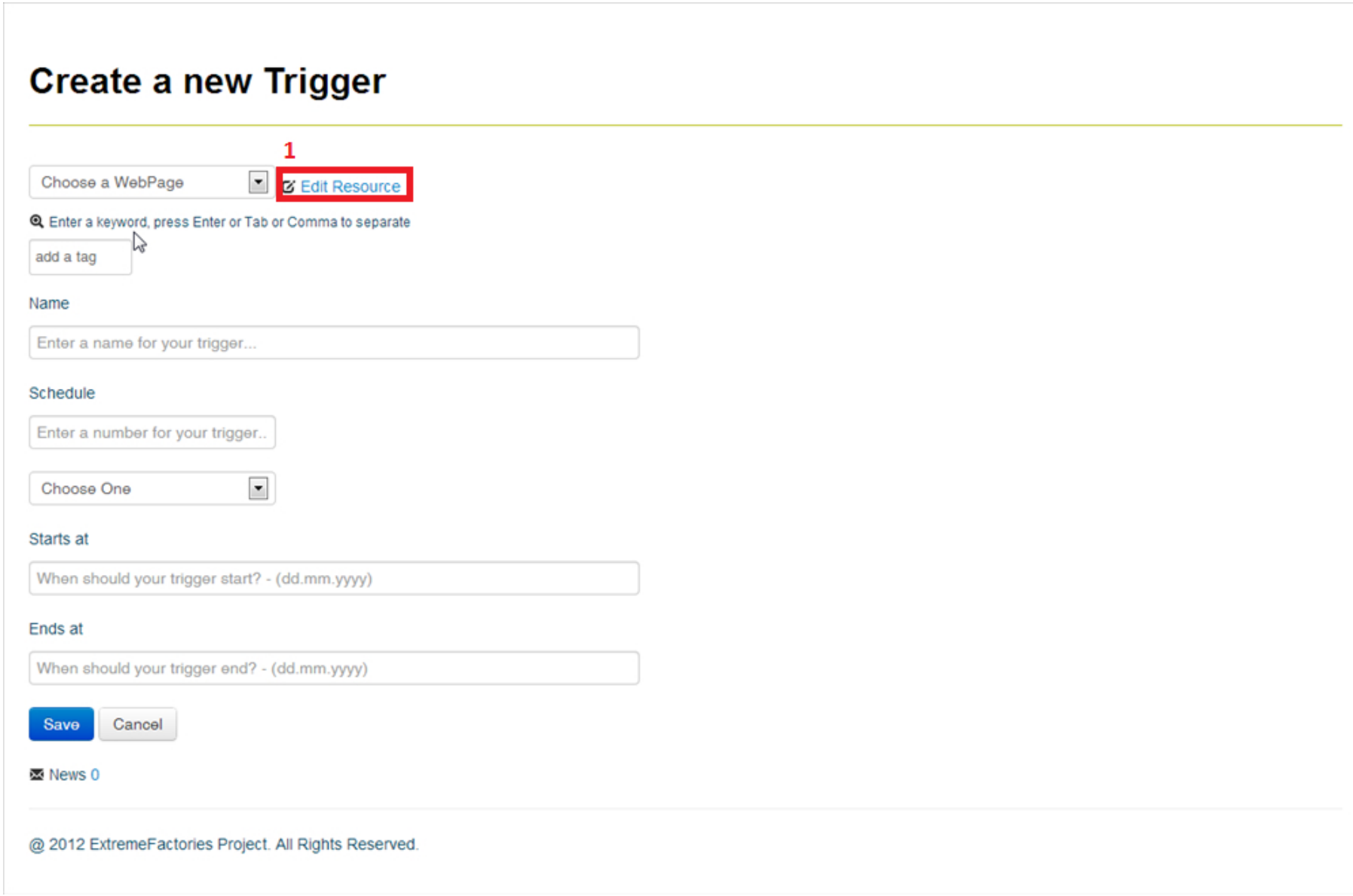

Figure 4.16: Page to Edit or Create a trigger

.

4.3.3 Similar Trigger

This screen has the ability to give a view about a particular resource, it would screen just the *triggers* that have keywords with the same meaning. The screen list sorts by name and gives the name of the *trigger*, whom have created the *trigger* and so on.The user may choose from keeping the *Trigger* or canceling it. The Figure 4.17 illustrates this screen, and once again, the Table 4.13 is a description of every single interaction supplied for the user.

Since a new *Trigger* is created the following screen is shown to the user, whenever a similar *Trigger* (i.e. with keyword(s) that has the same meaning) is detected in the system. The screen list is sorted by name and displays the name of the *Trigger*,the creator of the *Trigger* and so on. The user may choose from keeping the *Trigger* or cancel it instead. The Figure 4.17 illustrates this screen, and once again, the Table 4.13 is a description of every single interaction supplied for the user.

Table 4.13: Similar Trigger Legend

Name/Number	Definition
1	Name of new trigger
2	Views of the new trigger
3	Views of triggers with same meaning
Bottom Confirm	To force saving
Bottom Cancel	Cancel operation

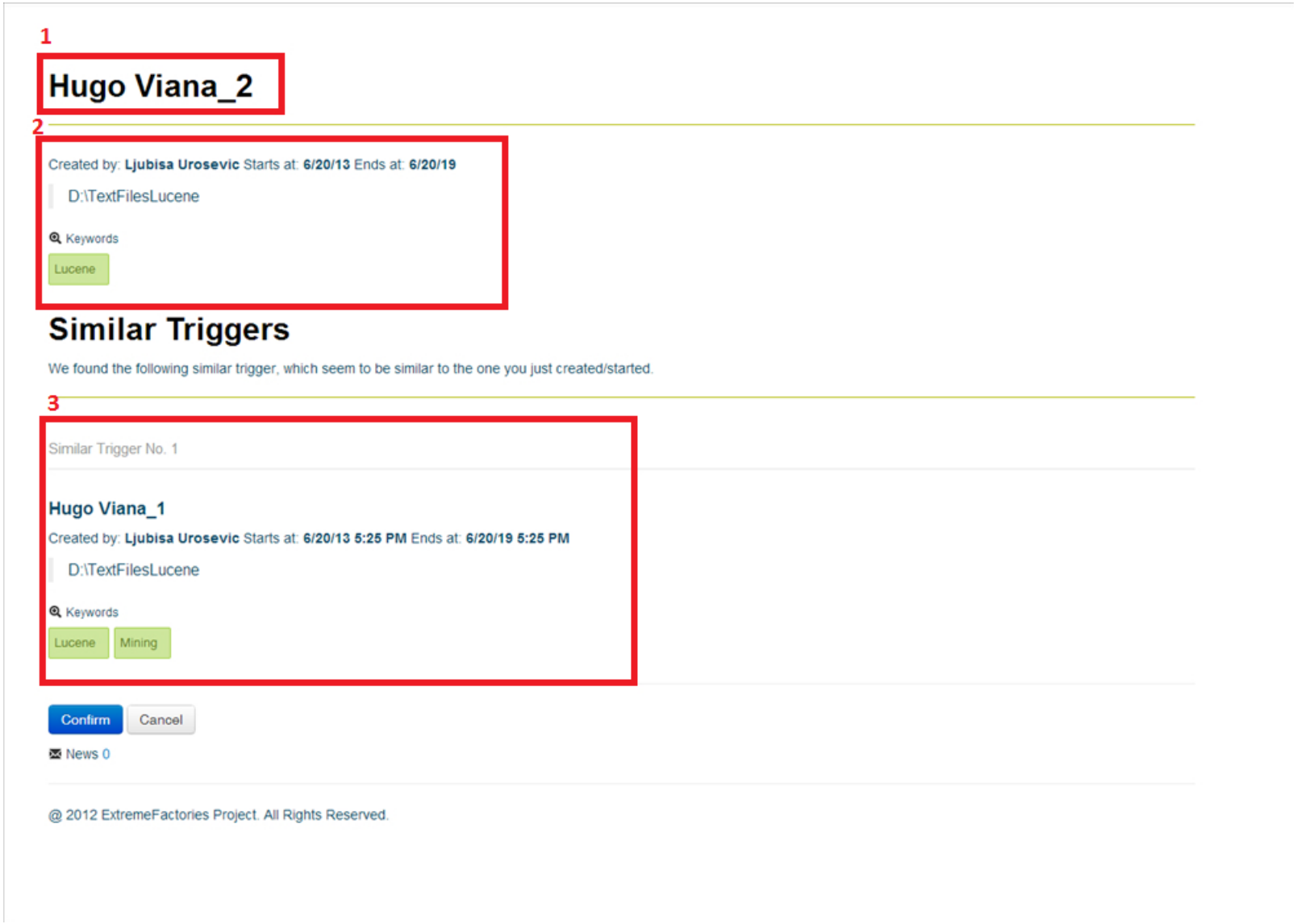

Figure 4.17: Similar Page

.

Chapter 5

Conclusions and Future Work

In the introduction, I expressed the objectives to be obtained for the platform.In this final chapter, I will conclude by describing the solution made for the purpose. I will also suggest some future work to be developed that could provide more efficiency for the commum user of EF.

5.1 Conclusions

In the thesis, a platform to aid ExtremeFactories project to gather information through a keyword, by means of a Resource[1] was described and implemented.

The thesis explained the appropriated decisions that have been taken to provide an efficient result for the platform. Chapter 3 explained which framework was chosen, as well as, a brief explanation about the choices. In order to be able to perceive the process about the Text-Mining developed for this platform, section 3.3 provides an adequate definition, by means of a schematic process. The whole development environment used for implement the application are supplied and illustrated in section 3.5.

Regarding the implementation, chapter 4 is important for two reasons. First, it provides the UML models, which defines, all actors, classes and interactions, but also, an explanation of every diagram. Furthermore, it provides the whole web interaction (i.e. Front-End) through figures, which means, all of the interaction with a common user is explained is this section. In the beginning, this platform has been developed only to gather

[1]Resource is defined as any kind of document source

information through a previous specified Resource, by means of a keyword(s). Meanwhile, a problem to validate a keyword(s) arises, from this point of view a feature to stem a word is needed. This new requirement provides to a common user a feedback page where one may verify whether there is any keyword(s) with the same meaning, in order to guarantee that there is no repeated survey for the same keyword(s).

This platform was developed to help SMEs to become more competitive. In order to accomplish this objective a schedule monitoring functionality was developed. This functionality tends to monitor not only the process of innovation ideas, but also the current market (i.e.prices).The result achieved by the usage of this platform was optimal. It has the ability to retrieve information from a directory with many files (i.e. PDF≪s) and after gathered and scored information by keywords(s), every file with the same name is compared with a previous file and verified if the score for that keyword(s) has been changed. Finally, it detects the changes either in files or in Web Pages.

However, the following two examples have been chosen to become clearer why this platform is indispensable for the SMEs. Firstly, as SMEs, you need to guarantee that your prices are, at least, the same that your opponents to maintain your profits. Secondly, in order to became more profitable, many of the SMEs have to be proactive, meaning that new new solutions/technologies must be pursued. Therefore, this functionality has the ability to control and verify, for instance a project report (called Resource), and inspect it, for the purpose of guaranteeing that all of the research/process for a new idea are going on a proper way.

The framework used to detect if a given file or WebPage have recent information is called Apache Lucene, which is a powerful tool for text-mining.Tests were made using PDF documents and WebPages through random keyword(s).

The platform was developed taking into consideration the possibility that a user may use any kind of browser, such as Firefox, Internet Explorer, and Opera and so on. In particular, Google Chrome was the chosen to one test this platform.

On the overall, this platform was successfully developed, and afterwards will be integrated with EF project.

5.2 Future Work

There is several suggestion for future work. One of them could be to check not only if a keyword is unique by using something like stemming and if something is written equally but also add ”semantic” uniqueness. This could be realized for example by using existing dictionaries which can help by identifying synonyms for example using WordNet that is a large lexical database of English. As an example: WordNet ”knows” that a ”Chihuahua” as well as a ”puppy” are dogs. This means that if somebody would look for a ”chihuaha” keyword, it could suggest to also add ”dog” or ”puppy” etc. as keywords for the same resource.

Extend the current platform to include more resource types, for example E-Mails, Word Documents, etc.

Extend the ”notification” mechanism to send e-mails to the creators of *Triggers*, so they don’t have to log into the platform to be aware of updates of resources.

References

Adriani, M., e. a. (2007). Stemming indonesian: A confix-stripping approach. acm transactions on asian language information processing. pp. 1–33.

Agrawal, R., e. a. (1996). Fast discovery of association rules. in u. fayyad, g. piatetsky-shapiro, p. smyth, & r. uthurusamy (eds.), advances in knowledge discovery and data mining. pp. 307–328.

Ahonen, e. a. (2-11). Applying data mining techniques for descriptive phrase extraction in digital document collections. proceedings of the ieee forum on research and technology advances in digital libraries.

Al-Shammari, E. T. & Lin, J. (2008). Towards an error-free arabic stemming. proceedings of the 2nd acm workshop on improving non-english web searching, conference on information and knowledge management. pp. 9–16.

Apache, L. (2013a, Jun). Documents. `http://www.lucenetutorial.com/basic-concepts.html`.

Apache, L. (2013b, Jun). Fields. `http://www.lucenetutorial.com/basic-concepts.html`.

Apache, L. (2013c, Jun). Indexsearcher. `http://lucene.apache.org/core/old_versioned_docs/versions/3_5_0/api/core/org/apache/lucene/search/IndexSearcher.html`.

Apache, L. (2013d, Jun). Indexwriter. `http://lucene.apache.org/core/old_versioned_docs/versions/3_0_1/api/all/org/apache/lucene/index/IndexWriter.html`.

Apache, L. (2013e, Jun). Queries. `http://www.lucenetutorial.com/`

basic-concepts.html.

Apache, L. (2013f, Jun). Searching. RetrievedJunhttp://www.lucenetutorial.com/basic-concepts.html.

Apache, L. (2013g, Jun). Searching and indexing. http://www.lucenetutorial.com/basic-concepts.html.

Apache, T. (2013h, Jun). Apache tika - a content analysis toolkit. http://tika.apache.org.

Arimura, H. (2000). Text mining: From the view of optimization data mining. paper presented at the international workshop on web knowledge discovery and data mining, boston.

Berman HM, e. a. (2000, Jan). The protein data bank.

Berners-Lee, e. a. (2001, May). The semantic web.

Blake, C. (2006). A comparison of document, sentence, and term event spaces. international committee on computational linguistics and the association for computational linguistics. pp. 601–608.

Blake, C. (2013). Information retrieval.

Blake, C. & Pratt, W. (2001). Better rules, fewer features: A semantic approach to selecting features from text. proceedings of the ieee data mining conference. pp. 59–66.

Blei, e. a. (2003). Latent dirichlet allocation. journal of machine learning research. pp. 993–1022.

Boyd-Graber, e. a. (2007). A topic model for word sense disambiguation.proceedings of the joint conference on empirical methods in natural language processing and computational natural language learning. pp. 1024–1033.

Califf, M. E. & Mooney, R. J. (1999). Relational learning of pattern match rules for information extraction. proceedings of the sixteenth national conference on artificial intelligence. pp. 328–334.

Caruana, R. & Hodor, P. G. (2000). High precision information extraction. paper presented at the kdd-2000 workshop on text mining, boston.

CollabNet (2013, Jun). Agile methodology. `http://agilemethodology.org`.

Cutting, D. R., e. a. (1992). Scatter/gather: A clus-ter-based approach to browsing large document collections. proceedings of the 15th annual international acm sigir conference on research and development in information retrieval. pp. 318–329.

Day, D., e. a. (1997). Mixed-initiative development of language processing systems. paper presented at the fifth conference on applied natural language processing, washington, dc.

Deerwester, e. a. (1990). Indexing by latent semantic analysis. journal of the american society for information science. pp. 391–407.

Deerwester, S. C., e. a. (1988). U.s. patent no. 07/244,349. washington, dc: U.s. patent and trademark office.

Dempster, A. P., e. a. (1977). Maximum likelihood from incomplete data via the em algorithm. journal of the royal statistical society. pp. 1–38.

Dr.Soman, A. . (2011). Parts of speech tagging for indian languages: A literature survey.

Eikvil, L. (1999). Information extraction from world wide web. pp. 23–30.

Feldman, R. & Hirsh, H. (1996a). Mining associations in text in the presence of background knowledge. proceedings of the 2nd international conference on knowledge discovery.

Feldman, R. & Dagan, I. (1995a). Knowledge discovery in textual databases (kdt). in proceedings of the first international conference on knowledge discovery and data mining (kdd-95). pp. 112–117.

Feldman, R. & Dagan, I. (1995b). Knowledge discovery in textual databases (kdt). proceedings of the ecml-95 workshop on knowledge discovery. pp. 175–180.

Feldman, R. (1996b). Mining unstructured data. tutorial notes of the fifth acm sigkdd international conference on knowledge discovery and data mining. pp. 1–36.

Fukuda, K., e. a. (1998). Toward information extraction: Identifying protein names from biological papers. pacific symposium on biocomputing, 3. pp. 707–718.

Ha-Thuc, V. & Srinivasan, P. (2008). Topic models and a revisit of text-related applications. proceeding of the 2nd phd workshop on information and knowledge management. pp. 25–32.

Hadoop, A. (2013, Jun). What is apache hadoop? http://hadoop.apache.org/#What+Is+Apache+Hadoop%5C%3F.

Havre, S., e. a. (2002). Themeriver: Visualizing thematic changes in large document collections. ieee transactions on visualization and computer graphics. pp. 9–20.

Hearst, M. A. (1997). Text data mining: Issues, techniques, and the relationship to information access. *Presentation notes for UW/MS workshop on data mining*, 112–117.

Hearst, M. A. (2013a, Jun). Untangling text data mining. proceedings of the 37th annual meeting of the association for computational linguistics. people.ischool.berkeley.edu/~hearst/papers/acl99/acl99-tdm.html.

Hearst, M. A. (2013b, Jun). What is text mining? people.ischool.berkeley.edu/~hearst/text-mining.html.

Ian H.Witten, e. a. (2011). *Data Mining : Practical Machine Learning Tools and Techniques* (3th ed.). Morgan Kaufmann Publishers Inc.

Inc, S. (2013, Jun). Data mining techniques. http://www.fmi.uni-sofia.bg/fmi/statist/education/textbook/ENG/stdatmin.html#neural.

Jain, A. K., e. a. (1999). Data clustering: A review. acm computing surveys. pp. 264–323.

Josiane (2013a, Jun). Creating a lucene index :how-to. http://sebol.webs.com/creatingaluceneindex.htm.

Josiane (2013b, Jun). The search process. http://sebol.webs.com/searchingtheindex.htm.

Kawahara, M. & Kawano, H. (2013, Jun). An application of text mining: Bibliographic navigator powered by extended association rules. proceedings of the 33rd hawaii international conference on system sciences. ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=926651&isnumber=20043.

Korenius, T., e. a. (2004). Stemming and lemmatiza- tion in the clustering of finnish

text documents. proceedings of the acm international conference on information and knowledge management. pp. 625–633.

Kumar, A. A. (2012). Text data pre-processing and dimensionality reduction techniques for document clustering.

Law, H. C. (2006). Clustering, dimensionality reduction, and side information. pp. 5–10.

Lent, B., e. a. (1997). Discovering trends in text databases.

Lewis, e. a. (2004). Rcv1: A new benchmark collection for text categorization research. journal of machine learning research. pp. 361–397.

Li, H., e. a. (2008). Scalable community discovery on textual data with relations. proceedings of the acm conference on information and knowledge management. pp. 1203–1212.

Liddy, E. D. (2013, Jun). Text mining. bulletin of the american society for information science, 27(1). `www.asis.org/Bulletin/Oct-00/liddy.html`.

Lucene-java (2013, Jun). Luceneimplementations. `http://wiki.apache.org/lucene-java/LuceneImplementations`.

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate obser- vations. proceedings of the berkeley symposium on mathematical statistics and probability. pp. 281–297.

Michael McCandless, e. a. (2010). *Lucene in Action* (2th ed.).

Nahm, U. Y. & Mooney, R. J. (2000a). Using information extraction to aid the discovery of prediction rules from text. proceedings of the kdd-2000 workshop on text mining. pp. 51–58.

Nahm, U. Y. & Mooney, R. J. (2000b). Using information extraction to aid the discovery of prediction rules from text. proceedings of the kdd-2000 workshop on text mining.

Paul, T. (2013, Jun). The lucene search engine. `http://www.javaranch.com/journal/2004/04/Lucene.html`.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. philosophical magazine. pp. 559–572.

Porter, M. (2013, Jun). Snowball: A language for stemming algorithms. http://snowball.tartarus.org/texts/introduction.html.

r Akeel Al-Attar (2013, Jun). Data mining - beyond algorithms. http://www.xpertrule.com/tutor/mining.htm.

Ramos, J. Using tf-idf to determine word relevance in document queries.

Riloff, E. (1996). Automatically generating extraction patterns from untagged text. proceedings of the national conference on artificial intelligence. pp. 1044–1049.

Salton G, B. C. (2013, Jun). Tf-idf. Term-weighting approaches in automatic text retrieval.

Savoy, J. (2006). Light stemming approaches for the french, portuguese, german and hungarian languages. proceedings of the 2006 acm symposium on applied computing. pp. 1032–1035.

Sebastiani, F. (2002). Machine learning in automated text categorization. acm computing surveys. pp. 1–47.

SHRESTHA, D. (2009). *Text Mining with lucene and hadoop: Document clustering with feature extraction.*

Siganos, C. S. . D. (2013, Jun). Neural networks. http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html#What%20is%20a%20Neural%20Network.

Simoudis, E. (1996). Reality check for data mining.ieee expert.

Srivastava, A. and S. M. (2009, Jun). Text mining: Classification, clustering, and applications. boca raton.

Steyvers, e. a. (2004). Probabilistic author-topic models for information discovery. proceedings of the tenth acm sigkdd international conference on knowledge discovery and data mining. pp. 306–315.

Tajani, A. (2013, Jun). 'crucial role' for smes in kick-starting growth. Public Service Europe. http://www.publicserviceeurope.com/article/1916/crucial-role-for-smes-in-kick-starting-growth.

Tan, A.-H. Text mining:the state of the art and the challenges.

Underhill, D. G. (2007). *Exploring Dimensionality Reduction for Text Mining* (1th ed.).

Wang, X., e. a. (2005a). Group and topic discovery from relations and text. paper presented at the proceedings of the 3rd international workshop on link discovery.

Wang, X., e. a. (2005b). Group and topic discovery from relations and text. paper presented at the proceedings of the 3rd international workshop on link discovery.

Yang, Y. & Pedersen, J. O. (1997). A comparative study on feature selection in text catego- rization. proceedings of the fourteenth international conference on machine learning. pp. 412–420.

Yang, Y. & Liu, X. (1999). A re-examination of text categorization methods. proceedings of the 22nd annual international acm sigir conference on research and development in information retrieval. pp. 42–49.

Zhou, D. P. (2013, Jun). Delve inside the lucene indexing mechanism. http://www.ibm.com/developerworks/library/wa-lucene/.

Zitting, C. A. M. . J. L. (2012). *Tika in Action* (1th ed.).