UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

DEMATERIALIZATION OF INFORMATION MANAGEMENT PROCESSES

por

Bruno Miguel Andrade de Almeida

PROJECTO

MESTRADO EM ENGENHARIA INFORMÁTICA
Especialização em Arquitectura, Sistemas e Redes de Computadores

2014

UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

DEMATERIALIZATION OF INFORMATION MANAGEMENT PROCESSES

Bruno Miguel Andrade de Almeida

PROJECTO

MESTRADO EM ENGENHARIA INFORMÁTICA
Especialização em Arquitectura, Sistemas e Redes de Computadores

Dissertação orientada pelo Prof. Doutor Dimitris Mostrous
e co-orientado pelo Pedro Saltão Ramos Moutinho

2014

Resumo

Com o objectivo de expandir a diversidade de opções para os seus clientes, a Novabase tem interesse em explorar alternativas open source na área de enterprise search (ES), enterprise content management (ECM) e business process management (BPM).

ES é um motor de busca composto por dois componentes, sendo o mais importante a pesquisa num índice invertido. Os candidatos são o Search Daimon [70], Solr [72] e ElasticSearch [21]. O Search Daimon é uma solução já muito completa, com muitas funcionalidades já feitas mas, não foi possível encontrar um manual ou o quer que seja que ajudasse a navegar e a modificar o código fonte e no excerto que vi, não estava comentado e foi difícil de ler. Uma alternativa de pesquisa open-source é o Lucene. O Lucene [51] é um motor de busca open source de muito elevada eficiência que trabalha sobre um índice invertido. É actualmente mantido pela Apache.

O Solr [72] é um sistema que altera, melhora e diversifica a interface do Lucene permitindo usar o Lucene com um API REST e um conjunto de funcionalidades de alto nível. O Solr é desenvolvido pela Apache no qual, recentemente, foi lançada a funcionalidade de trabalhar em cloud. O ElasticSearch é um search engine que altera, melhora e diversifica a interface do Lucene permitindo usar o Lucene com um API REST.

O Search daimon [70], pouco chega a ser considerado por causa da falta de informação sobre como trabalhar com ele sem ser com a interface gráfica. O Solr tem um manual muito completo [73], bem como o ElasticSearch [27]. As versões mais recentes do Solr permitem funcionar em cloud, o mesmo para o ElasticSearch. Aliás, o ElasticSearch sempre foi pensado para funcionar em cloud ao contrário do Solr que foi adaptado para funcionar em cloud. O ElasticSearch permite alterar muitas das suas opções incluindo adicionar e remover nós do swarm sem ser necessário reiniciar o servidor [28]. O Solr necessita de reiniciar sempre que existe uma alteração a fazer. Para pesquisa, o Solr só funciona usando uma query string. O ElasticSearch funciona com uma query string e também funciona com um objecto JavaScript Object Notation (JSON) bem estruturado [27]. O Solr usa JSON, Extensible Markup Language (XML) e comma-separated values (CSV) para alterar opções e para gravar dados. Destes dois, o ElasticSearch foi escolhido como sendo o melhor.

Na parte de Enterprise content management (ECM), foram escolhidas as plataformas Nuxeo [55] e Alfresco [4]. Como critério, é necessário que os ECM tenham todas as

versões open source. O Alfresco tem uma versão que necessita de uma licença paga mas o seu código é open source. O Nuxeo tem a versão completa, que permite usar o potencial todo do programa, disponível livre de custos.

Sem contar com as interfaces como o utilizador (GUI) de que o utilizador tem acesso, ambos oferecem as mesmas funcionalidades com muito pequenas diferenças (ver quadros na sub-secção "Comparing solutions" do 2.2.2). Embora o Alfresco community e o Nuxeo sejam muito parecidos, o Nuxeo desenvolveu uma ferramenta online paga, Nuxeo studio, que faz o mesmo que o Alfresco Enterprise. O Nuxeo studio [61] é um conjunto de ferramentas que oferecem um GUI [62] para controlar quase tudo no Nuxeo.

Por causa da grande quantidade de semelhanças entre os dois e como os projectos em que ECM pode ser usado, torna-se impossível escolher qual dos dois é a melhor opção para o maior número de projectos. Mesmo assim, tendo em conta os vários parametros e caracteristicas do Nuxeo que estudei vs o equivalente do Alfresco e vice versa e com a nova versão do Nuxeo, concluo que o Nuxeo, embora não seja uma escolha ideal, é um ECM melhor pelas funcionalidades que tem, facilidade de personalização e pela organização e simplicidade. O ECM escolhido foi o Alfresco porque, independentemente que o Nuxeo seja melhor, se não for suficientemente conhecido, é o mesmo que não existir, independentemente do quão bom ele seja e estável, sem erros.

O BPM decidido foi o que estava previsto na proposta inicial.

Tendo já estes dados todos prontos, foi-me dada a tarefa de desenvolver um programa para integrar o ECM escolhido (Alfresco) com o BPM escolhido (jBPM).

Durante a análise do jBPM, a empresa introduziu-me o Activiti como uma alternativa a ter a conta.

Por estar a não usar um framework no primeiro projecto, um segundo projecto, baseado no primeiro, foi iniciado para executar o mesmo que o antecessor. Este projecto foi desenvolvido usando o primefaces e tinha, como um dos objectivos, comparar o Activiti e o jBPM. Com o prazo a acabar e por falta de informação dos superiores, este foi abandonado incompleto e o promeiro projecto foi retomado.

No final, o primeiro projecto ficou uma prova de conceito para demonstrar uma possível interface muito versátil e muito personalizável que usa o Activiti (faltam algumas funcionalidades) ou o jBPM (suporte completo) como fonte para BPM e uma ligação CMIS para uma ligação ECM usando os dois para trabalho em equipa para desenvolver tarefas humanas.

Palavras-chave: ECM, BPM, Enterprise Search, Open Source

Abstract

Enterprise content management (ECM) [77] appeared some years ago with the main purpose to reduce the amount of time dedicated to deal with the paperwork required to run a business.

Assets were (and still are, for most companies) stored in large warehouses with strict sorting and very strict access rights. Without an ECM, when an employee requires a specific document, he would have to ask someone to search the document, get the document, and deliver the document to him, potentially, in a different building. Having an employee whose main purpose is to search and deliver company's assets to whom requires them is probably not the most productive way of using an employee's time. The ECM's main work is to place a mirror to that repository and those employees who would run around delivering the documents (or copies of them) and manage the assets as if it was physical paper documents, except everything is digital while doing many other asset related tasks in the background.

The usual setup is having an ECM working along an enterprise search backend and, optionally, a business process management (BPM) [15]. The enterprise search backend is responsible for indexing all search relevant information that exists in the ECM so that everything can be easily found with the known information about the thing. The BPM takes care of many internal system communications with the workflow management as its main feature.

Keywords: ECM, BPM, Enterprise Search, Open Source

Contents

List of Tables

Chapter 1

Introduction

1.1 Motivation

In the current enterprise world there has been some emerging of the open source code. This open source boom started with small projects and has been around for some years. On the other hand, the development at enterprise level using an open source license can only be found in the latest years.

Now, enough open source licenced programs have been able to show themselves worthy and robust enough to be taken notice if they can handle to solve all the issues the current closed source software solves, in the same enterprise environment their (closed source) counterparts live, with their strict rules and requisites.

The open source choice is an advantage to many companies as those software have no licensing costs and, if changes to the source code are required to be made, then they can be made without any licensing issues.

The main pluses for such choices are that in open source, code can be changed into anything and it (usually) can be used in anyway the company wants. The main downsides when open source is used is usually related to forcing that all code that had been open source will stay open source in the derived programs or that all programs that include the open source code must also be open source (for more information see section 1.6 on Open source licenses).

1.2 Host company

Novabase is an information technologies (IT) company specialized in consulting. It works with several market sectors such as: telco, financial services, government, healthcare, transports and utilities.

The section I was placed in, is the "ECM&BPM CP[1]" section. Their main focus is to deliver Enterprise Content Management (ECM) and Business Process Management

[1] Center of competence

(BPM) solutions towards the client's needs. They always have been working with proprietary software so now, because open source is becoming more and more common, cost effective, and even supported by Portuguese legislation, they decided to include open source ECM and BPM to their proprietary ECM and BPM list they offer support to.

1.3 Objectives

This project is about trying to find good open source alternatives to the widely-known proprietary ones that enterprises use.

Our investigation is made according to the objective features that my host company believes that its clients want, as explained below.

In this research, I will work on finding open source alternatives to enterprise search, enterprise content management (ECM) and business process management (BPM) as a lower cost alternative.

This project is meant to allow creating more commercial options for Novabase's clients using the 3 components (enterprise search, ECM and BPM) as individual component lower cost alternatives to the commercial well known enterprise solutions. For example, a company that bought a very expensive ECM and then sees itself as needing a BPM that can be much more simple, Novabase can suggest an open source BPM/ECM/ES to fill that gap.

1.4 Contributions

I documented a detailed comparison between Search Daimon, Solr and ElasticSearch, Alfresco and Nuxeo and jBPM and Activiti.

I programmed a highly customizable proof-of-concept program that acts as a user task executer for BPM engines which uses the ECM as an asset storage service.

This program is meant to be used by the host companies as the front end user interface for user tasks of workflows that are executing in the backend BPM engine.

1.5 Multiple meaning terms used in this document

While discussing about ECM terms used by Alfresco [4] and Nuxeo [55] (the two Enterprise Content Management studied) with my co-advisor, I found out that there are different terms used by other different systems for the same thing. So to avoid misinterpretation I'll clarify some of those terms:

- **Content type**: (not to be confused with file type or MIME type [53]) Some ECM call it Categories. Defines a set of metadata that an asset (or document) must have.

Systems may also include other changes related to changing a content type, such as forcing to belong to specific categories or forcing to have specific tags. For example, Alfresco allows making specific individual forms for each content type.

- **Category**: Works the same way as a tag but it works in a hierarchy and, depending on the ECM, may only exist one per asset . On the other hand, tags exist unrelated to each other (from the system standpoint) and tested ECM allows multiple tags per asset. Unlike Content types, these only relate as markers for similar content with no other use.

1.6 Open source licenses

Open source is not a license, instead, it is a full set of licenses where each one has its own pros and cons.

Open source means that the source code of the program is open for anyone to read and, most of the cases, alter and redistribute with the changes.

From all the open source licenses, the ones I believe to be the most common are, in no particular order, MIT [54], GPLv2 [33], GPLv3 [34] and Apache license 2.0 [11] [35].

The MIT license is one of the most permissive licenses in this list. To keep the explanation simple, it only imposes that the code used must always give credit to the original owner. This means that you may use that code to whatever you want to and to adapt anyway you want to suit your needs while keeping a single file and a comment at the beginning of the file, to acknowledge the original source.

The GPL license is the same as MIT except that a work using code that was distributed as GPL must stay GPL for the rest of its lifetime. This is called copyleft. Copyleft is the rule that forces the code released under GPL (and all its derivations) to be released under GPL (not the whole application, only the excerpt that was GPL and the result of all changes made to it).
For example, phpBB [67] is a bulletin board (forum) software released under GPLv2. Now imagine that you are developing a commercial application suite and you decide to include phpBB but with some code modifications. You also make some drivers that communicate with phpBB from the other applications.
In this case, if someone asks you for your version of the phpBB code (it does not matter who) you are obliged, by law, to provide that code. But, if they also ask for the drivers you made, you are in your right to deny access to the code as that is not a derivation from phpBB, it is just a translator you have developed.

The LGPL is the same as GPL except it does not force the use of the software under the GPL license (it is the same as GPL without copyleft) [50].

The Apache license allows usage of the code anyway the developer wants as long as the disclaimer and the license are kept intact [12]. This is about the same as the MIT

except that the author of the code may add extra conditions (I was unable to understand the exact limits) on how the software may be used.

For the programs that were researched in this document, both ElasticSearch and Solr are Apache license 2.0, Search daimon is GPLv2 [71] and both Nuxeo and Alfresco are LGPL.

You can get a full list (made by the GNU team) of open source licenses at `https://www.gnu.org/licenses/license-list.html`.

1.7 Disclaimer

All descriptions and explanations on this document are exclusive on the knowledge I was gathering while I wrote in this document which I assumed to be correct. All contents hereby in are **not** necessarily related to **Novabase**'s point of view or anyone else's to such point that contradictions may exist between each party's point of view.

1.8 Document's structure

This document is organized in these chapters:

- Chapter 2 – Related work – Explanation on how I searched to gather my sources and some basic conclusions.
- Chapter 3 – Analysis – Personal analysis based on the data I gathered and my past experience.
- Chapter 4 – Design – A higher level description of all my issues and solution while trying to make both projects I developed
- Chapter 5 – Implementation – A detailed view on the most notable problems that required solving while coding.

1.8.1 Explanation on the colored YES and NO on comparison tables

The "YES" and "NO" on the tables use different tones of green and red to specify how easy or direct it is to do or if the feature is included or not directly with the package I tested.

Here's a guide on how to read them:

Yes – The feature is included and I consider it easy to use or activate.

Yes – The feature exists but there's some quirk or "gotcha" on how it is used or on how to activate it (see message in parenthesis next to it for the explanation)

Yes – It can be made but it requires some sort of complicated gimmick, a workaround or the instructions say it does but I couldn't test it (see message in parenthesis next to it for the explanation).

No – I found a possible workaround to it. The workaround is not the feature and it won't replace the mentioned feature but it is good enough to partially or fully cover the main objective for such feature (see message in parenthesis next to it for the explanation).

No – That feature does not exist or I was unable to find it.

Chapter 2

Related work

In the core of any enterprise program that requires search there's an enterprise search engine. A search engine is not only about searching and displaying, it must follow many performance and security regulations that ensure that no one sees more than what they must, to do their job, based on their permissions, and it must return results such that the most relevant information always comes first for each search.

2.1 Enterprise Search

2.1.1 Why starting with Enterprise Search?

From the three components, the enterprise search is the most detached and multi-purpose software. This component can be used in systems like an ECM, product search (like in amazon's website or Dell's website where it is called refined computer search) and article search for variable purposes such as product search, full text search (for example wikipedia's search) and limited access search (only people with access rights may see it exists) while the user only inputs a simple string or fills a large form to refine the search at its fullest. Also, in this project, the enterprise search component is also meant to be used as the search backend for the ECM that is chosen later in this research. For last, working with enterprise search before the other two components helps understanding how such component interacts with other components and how other components use it.

2.1.2 Terminology

Inverted index: A kind of index where the results are stored for search instead of the possible queries. Although it is called that way, a result for an inverted index for words and numbers is a single string of characters delimited by a special character or a space character. For example:
"I am a sentence"

Has 4 words. So the index would store:

“I”
“am”
“a”
“sentence”
Sounds like a waste of space, right? Not really. Now think that each full string/document has an unique ID and you only have these 3 sentences in the system (example from wikipedia [37]):

0: ”it is what it is”
1: ”what is it”
2: ”it is a banana”

This would translate to this index:

”a”: [2] ”banana”: [2] ”is”: [0, 1, 2] ”it”: [0, 1, 2] ”what”: [0, 1]

This means that the word “a” appears in the sentence 2, the word “is” appears in the sentence 0, 1 and 2, etc... This kind of index can store variations of this. It may store complete sentences, for example, but it is not that useful because it is quite rare for that to happen.
Here, searching for something using just keywords is a straight forward move. It is also possible to index this inverted index to get even faster searches. For example, to use a binary tree that allows searching for the terms in O(log n) instead of O(n). Anyhow, the number of keys this index will ever have can never be more than the number of words that exist in the language in question which is not that long for a computer to search through in brute force. Additionally, very common words such as “is”, “the”, “as”, “a”, etc... are filtered out of the index due to don’t really helping the search and, sometimes, actually degrades the search because these kinds of searches are, by default, made using an “or” instead of an “and”.

2.1.3 What is Enterprise Search?

Enterprise search (ES) is a search engine that works on an inverted index that allows searching through all company’s (or groups of companies) internal documents using a free text or a fine tuned search form while respecting the company’s policy on user permissions. An enterprise search usually works with two main sub-systems:

- A crawler (optional but recommended)
- An indexer/searcher

The crawler is responsible for accessing and indexing every document that the company has (a document is a file that has enterprise value) and send it to the indexer to make that file searchable. It is also responsible for analyzing the interior of the documents and

extract their metadata and also information about where other documents may be. This component is considered optional for such system because documents can be sent (for index) directly to the search engine.

The Search engine (query engine) is responsible for indexing, interpreting the user input and searching the documents it has indexed as fast as the hardware allows. This engine must be able to work efficiently with multiple metadata at the same time so it should not be a table-like SQL system because each index is meant for a fixed number of columns and each query may only use a single index.

2.1.4 Research

In this part of the work, I searched for open source no-SQL systems that allow doing a full text search of data and allow associating data (metadata) to that text and I also searched for a crawler.

While researching, I came to the conclusion that the most important features were the following:

- Filter based on user permissions (Do not allow accessing to certain results if the user does not have permissions to see them)
- Results suggestion (when a user searches for something it suggests search alternatives to increase the number of results)
- Content suggesting (When searching for something specific, it returns extra information directly in the results outside the direct result)
- Search groups (when searching for something in a specific place, also search in a different place) – Yes (Alias system [22])
- Stemming Expansion (when searching for X, also return results related to X) – Yes (Synonym search [30])
- Alternatives search (did you mean... The same kind that google uses) – Yes (Phrase suggestion system [29])
- Lexical search:
  - Broader term search (When searching for a car, it also searches for other terrestrial vehicles) - ? ([25])
  - Synonym search (When searching for a car, it also searches for automobile as they are synonyms) – Yes ([26])
  - Narrower term search (When searching for “automobile” it intelligently returns “car” or (exclusive or) “bus”) – No

- Limit search to certain sources (Limiting the search so that not only the content is important, its source is also important. For google search users, that's equivalent to using the "site:" search command to limit search inside certain websites (sources)) – No
- Search documents in a specific language (E.g. Search documents only in english or only in Spanish or only in Portuguese) – No (easily overcome)
- Artificially raise the relevancy of a document (Different keywords have different weighs. An "and" is very frequent, so it should have a low weigh a name of a model of a brand is very infrequent) – Yes ([24])

In that search I found four main contestants for the job.

Lucene: A core application that stores, indexes and retrieves data structures from its internal database. It is considered one of the fastest backend search systems and as fast as the proprietary equivalents. Lucene is a search engine that stores and retrieves information from an inverted index (can also be used to store raw information in addition to the index feature). It contains other very useful features like results scoring to try to deliver what is most relevant first. Lucene is developed by Apache.

Search daimon: Initially this software was proprietary and, very recently, it became open source. It has a decent amount of information about how to use it but no real amount of documentation about how to work with its code.

Solr : Displays a better interface to the other applications using HTTP Rest concept and it allows brushing up lucene to its greatest. Recently it was changed to work in a replicated system. Also developed by Apache.

ElasticSearch : Appeared as the main open source rival of Solr and it is designed as a distributed system from the start. It works on top of Lucene just like Solr but provides a different interface to the user with some other different features making it the main rival for Solr.

Using a crawler

A crawler is a software that browses resources in the environment it is crawling and announces its findings to the observers that have registered to be announced. Crawlers, such as Nutch (refer below), also include an option of blacklists or whitelists to prevent it from crawling outside a defined sandbox. While searching, the information that is announced is implementation dependent.

In a file system, the crawler starts in the root of the file hierarchy, follows all directories[1] and announces all files that were found.

In the internet, the crawler starts at some page and then it follows all hyperlinks that it finds to gather the resource list that exists.

Nutch [13] is a very lightweight command line crawler developed by Apache that is capable of crawling through any crawlable environment, including complete websites and all pages that website points to and tree-like file systems with (soft and hard) links, directories and files.

In this project, I was unable to work with Nutch and I abandoned that idea. Nutch is only ready to work on linux and it is part of the project's rules that the solutions must work on Windows (and that a work that works on both Windows and Linux supersedes a work that only works on Windows). So I tried using Cygwin without any luck. That forced me to abandon the idea of trying Nutch.

Fortunately, some solutions do not need a crawler because they already support enough functionality for enterprise search. ElasticSearch, same as Solr, is only the indexing/search subsystem for the enterprise search, it does not include a crawler. So the only crawler that I found that was recommended was Nutch but then, Nutch only works on linux because it is made and tested only on POSIX. So I tried using Cygwin to have it running on windows but it wasn't starting. Then I tried running it on Ubuntu in a virtual machine but it was crashing and the help I tried to get was not allowing me to solve it.

The crawler's job is not required for the search engine to work. It just adds a useful feature to the system. Without a crawler, however, it is required that all information that needs to be indexed to be (manually or automatically) delivered when the content that is searched is created or modified.

Due to the difficulties and because the crawler is not a requirement, I ended up giving up and moving on without a crawler.

2.2 Enterprise Content Management

2.2.1 What is ECM?

An ECM is an enterprise content management. It is a platform that works together to make sure all documents are correctly placed with the right permissions at the right time, reviewed by the right people, correctly archived with the right access rights, easily and quickly found when searched and are correctly destroyed when they are useless and will be forever useless [77][2].

An ECM is made of these components:

[1] Some people call them folders.

[2] All contents are my point of view only. See:1.7

1. Lifecycle management
2. Web content manager
3. Capture
4. Content manager

Lifecycle management: It's the main sub-system that follows an asset from when it is set as a record until its destruction. This system is responsible for maintaining the records in the right place, at the right time or the information about the existence of the record. Records are information assets that have been considered as final so, that means that it becomes impossible to edit no matter the permissions or role of the user. Those documents work under very strict rules about who can view them, where they must be placed and when they can be safely destroyed. This sub-system is used usually for critical documentation like contracts and invoices.

Web content manager: Responsible for delivering a feature rich web interface not only as an API (XML Rest, JSON, etc...) but also a GUI (HTML, CSS, etc...). It is also required to deliver an interface to technical and non-technical administrator (menus and drag and drop) to allow customization of the GUI pages to have the right look and feel for the web pages users.

Capture: This sub-system is responsible for the image management where the documents lack. As these are images, there is no text in them (from a computer point of view) so they can be treated in such way that data may be required to be extracted from the image as if it was text and any other special treatments that images allow.

Content manager: Allows viewing documents in the program's interface, changing metadata (or even the data itself), changing permissions... Just about anything that is related to managing the assets that reside inside the ECM itself is treated by the document manager.

2.2.2 Research

In this part of the work, I searched for ECM's that were open source not only for the "community" version counterpart but also open source for the version that the company is actively developing and supporting officially. In that search I come up with only two results:
Alfresco and Nuxeo.

The reason why only these two were chosen is because all the other ones have an open source version and a proprietary version. Their proprietary version is closed source meaning that it's against the law to make any changes to its source code, no matter what it is. In Alfresco's situation, it's version that is fully tested is also not open source. It has a peculiar difference from the rest, though. As long as the license is paid, the license owner

has access to its source code and has the right do as many changes as he pleases while not voiding the support that was bought. It is under the LGPL license.

In Nuxeo's situation, the full enterprise software can be used for free. Paying in Nuxeo is just for the professional expert support and for access to Nuxeo studio which is a toolset that allows generating .xml files using a graphical interface. Unfortunately, as I was not given a Nuxeo license, I was unable to try out this toolset, so all my knowledge about Nuxeo studio comes from the manual and from the advertisements about such system.

ECM comparison

In this section, a comparison between both ECM (Alfresco and Nuxeo) will be made based on the following topics:

- Adding documents
- File management
- Access management
- Workflow engine
- Search capabilities
- E-mail operations (operations that are possible by sending and requesting e-mails)
- Misc (things that do not fit any category and are not enough to have a category of their own)

Adding documents

Both systems are not prepared to generate files from a scanned file. Instead, there are 3rd party solutions that offer such service.

For alfresco, there's, for example, this: `http://www.alfresco.com/partners/solutions/document-indexing-module-alfresco-share`.

For Nuxeo there's this: `http://www.cobratech.com/cir6-nuxeo/`

When bulk adding documents from the file system, both Alfresco and Nuxeo allow adding a file with the related metadata of each file, though each one has a different way of accomplishing it.

In Alfresco's case (enterprise version has it integrated. Community users have to use an addon [7], the user may use a file with the name ending as ".metadata.properties.xml" prepended with the name of the file it applies to. For example, a file with the name "picture.jpeg" the properties file name would be "picture.jpeg.metadata.properties.xml".

In Nuxeo's case, there is a bulk file importer available as an add-on [56] that supports importing the metadata related to the files imported. There are two ways. There's the

	Alfresco	Nuxeo
Add documents using a scanner	Yes (Requires a third party program)	Yes (Requires a third party program)
Add document from a template	Yes	Yes (Requires editing a .xml file or Nuxeo studio)
Bulk add documents that exist in the file system (with metadata)	Yes ([8]. There's also an extension for the community version [7])	Yes ([56])

Table 2.1: Comparison of features for adding documents

one file for everything within the directory (the default) or the one metadata file for each file/directory. That file is a properties[3] file with an xpath to the information as its key and value. There is also an alternative importer [58] (also an addon) that uses a CSV file with all the information about the import (including where the file is in the file system). See table 2.1.

File management

Both Alfresco and Nuxeo offer just about the same features when working with files so none is better than the other in this category. See table 2.2

Access management

Alfresco has a much larger set of permissions than Nuxeo and it also has a roles system integrated. Nuxeo does not have a role system but the permissions system is quite small so the permissions themselves end up working as Alfresco's roles. To make any changes to the default, the administrator has to alter .xml files or use the paid systems that both offer (Nuxeo studio and alfresco enterprise). See table 2.3

Workflow

Both are equivalent on how the workflow sub-system works from the user stand point and both require changing .xml files in case the user wants to add his personalized workflows or forms for each workflow step. To do this, both offer click-and-drag GUI in their own paid platform. See table 2.4

Search

Nuxeo's default search form is more complete than Alfresco's default search form which allows Nuxeo to do a more refined search for a term without the need of knowing any special search syntax to correctly filter the search as wanted. Otherwise, both Nuxeo and Alfresco have an equally powerful search backend by default. See table 2.5

[3] http://en.wikipedia.org/wiki/.properties#Format

	Alfresco	Nuxeo
File lock/checkout	Yes	Yes
Upload files using drag & drop	Yes	Yes
Preview a file (in the system itself)	Yes	Yes (I was unable to test this due to an error in the version I used)
File version branching	No	No
Metadata automatically read		
Multiple versions of the same file	Yes	Yes
Edit in the ECM itself.	Yes (When i tried (pptx e docx), nothing happened but it works if it is plain text)	Yes (requires an extension) ([59])
Edit using Google drive	Yes (When I tried the network proxy settings prevented the connection between both)	Yes (When I tried the network proxy settings prevented the connection between both)
Control metadata of a document type	Yes (Requires editing a .xml) ([6])	Yes (Requires editing a .xml) ([57])

Table 2.2: Comparison of features for file management

	Alfresco	Nuxeo
Roles	Yes (Can only be changed by editing a .xml file)	No
Limit access to a document to only during certain time or event	Yes (If it is moved when the time is right or then the even happens)	Yes (If it is moved when the time is right or then the even happens)
Permissions	Yes (Only using roles (5 roles by default))	Yes (Only 4 permissions: read, write, remove and manage)
Groups (with their own permissions)	Yes	Yes
Login using external systems	• LDAP • CAS2 • OAuth • Shibboleth • Kerberus	• LDAP • CAS2 • OAuth • Shibboleth • Kerberus
Import groups from LDAP (includes Active directory)	Yes [10]	Yes [63]
Navigational methods to access the documents	• Favorites • Documents I changed • Original organization (tree-like) • Tags • Categories	• Favorites • Documents I changed • Original organization (tree-like)
Stored password's hash	SMD4 (very unsafe)[4] SMD5 (configurable in a .xml) (unsafe) ??	SMD5 (unsafe) ?? SSHA1 (unsafe if not changed frequently)??
Apply different actions to a document depending on its metadata	Yes (Filtering using the whitelist blacklist event system)	Yes (One has to change an .xml file. They also deliver a visual SDK to create these but it is not free (Nuxeo studio))

Table 2.3: Comparison of features for access management

	Alfresco	Nuxeo
Apply a workflow to a document	Yes (As long as it is one of the default ones)	Yes (As long as it is one of the default ones)
Apply a workflow to a document based on its metadata	Yes (As long as it is a very simple workflow such as: If approved go to place X, if denied, go to place Y)	Yes (One needs to change an .xml manually or use their online SDK (Nuxeo studio))
Apply a personalized workflow based on the document's metadata	Yes (Using activiti in the enterprise edition)	Yes (One needs to change a .xml manually or can change using a Nuxeo studio (not free))
Automatic forms that request the metadata to the user based on the document type it mentioned	Yes (One needs to change a .xml manually)	Yes (One needs to change a .xml manually)

Table 2.4: Comparison of features for workflow control

E-mail

Both Alfresco and Nuxeo offer sending files to the system through an e-mail and none allow getting a file by e-mail from the platform. See table 2.6

Misc

In this table, the **simple support** is the minimum price to pay to get the software (in Alfresco's case) and also have access to professional support (support from the official developers of the project). In Alfresco's case, that price is for alfresco running in a single computer [6].

The **expert support**, in Alfresco's case, enables the full time support, increases the number of users included and enables the use of alfresco in a distributed system. In Nuxeo's case, it reduces the waiting time of response of the professional support and enables more users to use Nuxeo studio; it is claimed to be the most popular option they offer [64].

Both systems have ways to being used in a cluster (distributed/multi-threaded).

When I tried to find if Alfresco is able to be joined with ElasticSearch, I only got a single result which was a selling slideshow [49] about a solution a company made by using a middleware. But the real integration with only ElasticSearch and Alfresco is nowhere to be found. For Nuxeo, I found an addon that adds ElasticSearch as the search provider for Nuxeo. See table 2.7

[6] More information on alfresco pricing: [9].

	Alfresco	Nuxeo
Search inside the documents	No	No (I found documentation that states it does but I found no way of successfully test it. Maybe there's limited support)
Metadata search	Yes (If it is part of the default metadata)	Yes (Much more complete than Alfresco's and allows a much more refined search)
Filter results using the user's permissions	Yes	Yes
Suggest results using synonyms	No	No
Search automatically using a synonym dictionary	No	No
Artificially manipulate search results relevancy	No	No
Contextualized search	Yes (Contexts are limited to "sites" otherwise I was unable to search only within a certain context)	Yes
Select information source groups	No (I found no way of joining websites. Search in all or search in one)	No
Related content	No	No
Did you mean...[5]	No	No
Search by language	No	Yes

Table 2.5: Comparison of features for asset search

	Alfresco	Nuxeo
Ask file by e-mail	No	No
Send files to the system by e-mail	Yes (Each attachment stays in separated document and an extra document is created with the body of the e-mail)	Yes (You need special directories for this; the attachments become attached to the system's e-mail)

Table 2.6: Comparison of e-mail related features

	Alfresco	Nuxeo
License	LGPL	LGPL
Simple support price	$21 000	$25 000
More expert support price	$51 000	$38 000
Use in a cluster (distributed)	Yes (Only Alfresco enterprise with the "more expert support")	Yes
Workflow engine	Yes	Yes
Able to join with ElasticSearch	Yes (Zaizi has a middleware solution to join these components together) [49]	Yes ([65] unable to test properly)
Output in different file formats (Ex: pdf, jpeg, etc...)	Yes	Yes ([60]Using an extension and external libraries)

Table 2.7: Comparison of other relevant features

2.3 Business Process Management

2.3.1 What is BPM?

BPM is an application suite meant to create, edit and manage business processes and to create, execute and manage user tasks.[7]

As business process and workflow are used interchangeably in the BPM area, from now on, I'll call business process, workflow.

A *workflow* is a logical sequence of tasks connected using connections and gateways. If following the standard, workflows are written according to the bpmn2.0 specification [84]. The bpmn is a standard for specifying a BPM workflow that increasingly more and more BPMs are adopting as the de-facto workflow specification.

In workflows a *connection* is an abstract relation of cause-effect between two elements (task, gateway, start, end, etc...) of a workflow.

A *gateway* is a logical port used to converge multiple connections into one and/or to diverge a connection into multiple connections. When diverging, the engine will follow a single or multiple connections, depending on the gateway itself and the diverging conditions used. When converging, the engine may wait for all diverged connections to reach the gateway before continuing the execution to the output connection, also depending on the configuration.

A *task* is a piece of work assigned to an individual (or a group, and later assigned to an individual) or to the system itself. It has always one entry and one exit which is where

[7] All contents are my point of view only. See:1.7

the connections connect, input values and output values. A task can be a lot of things and explaining or enumerating all of the possibilities is outside this document's scope. So I'll stick with the main ones I'm working on: the user task and the service task.

The *user task*, as the name suggests, is a task that is supposed to be executed by a human being. In its essence, when it's time to start a user task, the system retrieves the required data to display to the user and the form that the user has to fill and displays both to the user in the GUI. The user fills in the form and the user submits the form.

The *service task* is a task executed by the system itself. A class is given to the system that contains a method that is executed when the worker reaches the service task. [42]

Then jBPM has a different kind of task called *custom task* [47]. These tasks are as versatile as Java itself is. Any information may be retrieved or inserted into the workflow while any java code can be run at the same time. Trying to work with these was a very hard and unpleasant process which I cover below.

In jBPM, service tasks can only have a single output value. On the other hand, Activiti allows unbounded amount of output because the method that executes the service task receives an object that delivers direct access to all variables of the workflow. But still, jBPM allows making personalized tasks; tasks that execute without any of the standard service tasks restrictions.

2.3.2 Research

This table 2.8 sums up every feature I was able to think of that is worth mentioning and compare between these two BPM.

	jBPM	Activiti
Workflow organization	All workflows are identified by a deployment id with a name long enough to be unique. It follows the same rules as a maven project. Then each workflow in that deployment has a unique id.	Each workflow is given an unique id and a name. The id is given by the system and the name is by the user. In case of name clashes, id can be used
Complete workflows are stored	Yes	Yes
It's possible to keep multiple versions of the same workflow inside the system	Yes (Custom tasks still have to be unique in the whole system, though)	No (Once a new version is made, it's impossible to start an older workflow. It still keeps the old unfinished ones until they are finished, though.)
Execute arbitrary code	Yes (Custom tasks (work item handlers))	Yes (Service tasks)
Multiple versions of arbitrary code at the same time	No (Requires java classes with different names)	No (Requires java classes with different names)
Possibility to revert actions made.	Yes (jBPM runs on git for all operations. Just do a revert or a reset to go back in time)	No (Activiti only goes forward. If there's a mistake or corrupted information, fix it manually or forget it)
Workflow participants	Owner, Assignee	Owner, Assignee, Contributor, Implementer, Manager, Sponsor
Creating comments related to a task (attached to a task)	Yes	Yes
Local API to for remote communication	Yes (In java. Does not include all remote functionality.)	No (Manual REST calls must be made)
Variable size limit	255 characters (availability to increase being developed)	Unknown
Automatic form generator	Yes (For Strings only)	Yes (For all types included with Activiti)
Assisted form generator (allows making forms for who does not know HTML)	Yes	No (Forms are generated automatically only)
Personalized forms	Yes (Up to any desirable HTML (except the input element))	No (Forms are generated automatically only)
()	Yes ()	Yes ()
()	Yes ()	Yes ()
()	Yes ()	Yes ()

Table 2.8: Comparison between BPM

Chapter 3

Analysis

In this chapter we are going to analyze the features and aspects of different solutions for enterprise search, enterprise content manager and business process manager with the objective of selecting the best option based on the information studied in the previous chapter.

3.1 Enterprise search

Enterprise search is a software that allows searching for content in the best way possible. Best being most accurate, fastest, easiest (for the user) and with the smallest search string possible.

3.1.1 The advantages and disadvantages

Now we are briefly going to analyse advantages and disadvantages between Daimon search, Solr and ElasticSearch.

In the following table 3.1, I'm going to list different characteristics that I find useful or important for an enterprise search to include. Between them, I included:

- Permissions
- Completeness of the administration control panel (based on the features each ES has)
- Requiring restart to apply changes.
- Extensiveness of the instructions manual
- Expert Support

	Search Daimon	Solr	ElasticSearch
Permissions	Yes ()	Yes ()	No (Can be emulated. See sections below)
Complete administration panel	Yes ()	No (.xml only)	No (Rest commands and a .xml only)
Requires restart to apply	Yes ()	Yes ()	No (Using Rest commands)
Includes crawler	Yes ()	No (Can be joint with an external one)	No (Can be joint with an external one)
Has search GUI	Yes ()	No (There are external programs that make an interface)	No (Use the REST API instead)
Has complete user documentation	Yes	Yes	Yes
Has complete developer documentation	No (Not obvious or inexistent)	Yes ()	Yes ()
Extensive external API	No (After trying to search for it, I was unable to find it)	Yes ()	Yes ()
Age[1]	Became open source last year (2013)	Born in 2004	Born in 2010
Active community	No	Yes 3.1.1	Yes 3.1.1
Is real-time	? (No information was easily available at time of writing)	Yes (All operations are near-real-time)	Yes (but only when physically possible)
Free support	Yes ()	Yes ()	Yes (Community forums)
Commercial support	Yes ()	Yes ()	Yes ()

Table 3.1: Comparison between the three analyzed enterprise search

Search daimon's advantages

It is a crawler, non-SQL based database, searcher and GUI in a single system ready to use.

Includes a fully functional administration panel with some neat options to customize how it should work. The most notable options for me are:

1. Control over its internal crawler: Where to search, where not to search and when to search.

2. User management: Adding users, removing users, controlling user permissions, etc.

3. IP whitelists and blacklists for accessing the interfaces (search UI or administration panel UI)

4. Control integration with LDAP for user management

It brings a search GUI ready to use.

Search daimon's disadvantages

The GUI (both frontend and backend) is very difficult to modify, especially because there's still no documentation to be found about how adapt the software code to the needs of the developer.

Not enough documentation on how to communicate with it from other software. The one that exists is too incomplete.

Only works on a single machine; no distribution.

All those problems are killers for a successful enterprise search software meant to be delivered to a client, nowadays.

Solr's advantages

Solr has interface in Extensible Markup Language (XML), JavaScript Object Notation (JSON) and comma-separated values (CSV). **ElasticSearch** only has interface for JSON.

Solr knows what a user is and has some concept about what permissions are. **Elastic-Search** has no clue about what a user is (therefore, no notion of permissions). **Solr** has a solid, longer lasting, community, making it more mature than **ElasticSearch**.

Solr's disadvantages

Solr usually requires rebooting for changes to be applied. Although some changes can be made by doing a reload. Unfortunately, though, it requires a temporary pause on all nodes.

ElasticSearch's advantages

ElasticSearch is more recent so it is built with the latest technologies and mindset from the very start.

ElasticSearch is meant to be distributed from the very start making it very optimized for distributed storage and retrieval. **Solr** became distributed more recently.

ElasticSearch has about the same amount of community as **Solr**. Considering that it is more recent, it has grown faster than its rival.

ElasticSearch is real time for simple put and get operations and near-real-time for the rest. **Solr** is near-real time for all operations. [31]
Real time means that the information is ready to be retrieved or found right after the add confirmation is made for the information that was just put inside the engine. Even though that is possible, it does not mean that the indexes are already updated with that information, so only exact matches and accessing the right chard and replica work at that time. Then replication happens as fast as information transmission allows (without killing the servers availability).
Near-real time means that there's a small delay (the actual amount varies on server load and size) between the time when the new data is added or changed and the time when that said data is available for retrieving.

ElasticSearch allows changing most parameters in real-time with no need to reboot the servers including when adding more nodes.

ElasticSearch's disadvantages

ElasticSearch Only allows communication payload in JSON, unlike **Solr** which accepts JSON, XML, CSV, etc...

Differences in Solr against ElasticSearch

These are different characteristics from Solr to ElasticSearch, and vice-versa which, depending on who is using it, may find each characteristic moot, an advantage or disadvantage.

Solr is meant for a generalistic work with no real aim for what it is meant for. **ElasticSearch** has a definitive aim. [32] **Solr** only allows searching using query strings, **ElasticSearch** allows searching with query strings but it also allows a more computer friendly structured search using a JSON object. **ElasticSearch** can guess a data structure based on the first insertion of data, in **Solr**, if you want to change the structure you need to restart the server

Which one is the best?

Selecting **ElasticSearch over Solr** or **Solr over ElasticSearch** is a very tough choice. The main reason is that they are really very similar to the point that there are addons for both to interconnect Solr with ElasticSearch so that each different system may use one or the other depending on the needs. In the end, the final choice falls over some minor details that end up helping ElasticSearch making up to win.

For me, **ElasticSearch is better** because it does not require a reboot every single time a parameter is changed, ElasticSearch has a more computer programmer friendly interface (which supersedes the multitude of input and output MIME [53] types that Solr allows) and ElasticSearch has always been a distributed system since the very start, which delivers some more trust when thinking about improving its code or the stability itself of its code.

Although there were more features taken into account, I really believe that it is those features that are the most important that are present in one ES and not in the other for a large scale company that requires as quick and easy scalability and availability as possible.

SearchDaimon didn't have a real appreciation because it does not deliver a good documentation and any changes to it would be too hard due to the lack of community behind supporting it, even though it is a quite complete solution with many available features and a ready to use GUI. But then the GUI itself is not easy to customize[2].

3.1.2 Dealing with the missing features

ElasticSearch, by default, does not contain multiple very useful or mandatory functionalities that an enterprise search requires. Still, it has an interface that allows the program that uses it such that ElasticSearch can be easily simulated as if it actually had those features. In this work, I worked on emulating the following features:

1. Search inside documents (Documents like .docx and .pdf) and
   Metadata extraction from documents (Documents like .docx and .pdf)

2. Search results permissions

3. Limited context search

4. Getting results from multiple source groups/types

5. Search content in a specific language

Now we are going to emphasize those problems and a possible solution for each one of them.

[2]In my case, I didn't find any documentation on how to customize it.

1. Searching inside documents and
2. Metadata extraction from documents

The problem

ElasticSearch search backend is using lucene. Lucene search algorithms are made for full text search; this means that lucene does not support searching inside binary data. Besides, ElasticSearch's interface with the outside is in the form of JSON which also does not support binary data. The solution with JSON is to use Base64 encoded data to store the file but then, both ElasticSearch and lucene cannot decode and guess the contents of the file that was given.

The solution

Send the responsibility of understanding that to the program that uses ElasticSearch and not to ElasticSearch itself and use apache Tika[14] to get the contents of the file as plain text.

Tika is a suite of freeware open source programs made to read the inside of binary document files (.docx, .xls, .pdf, etc...) and gather some information from them. It gathers readable metadata (author, change dates, creation date, etc...), the language in which it was written and a plain text version of the contents of the file.

The main idea is to have the program, for example, the ECM, when storing in ElasticSearch for indexing, to ask for the metadata, the plain text version and the language of the file and then store those data for indexing while storing the file in base64 or just an identifier for the file when trying to index the file to become searchable. Then, when searching for contents inside the file, the ECM will request that search to be made in the plain text version.

3. Search results permissions

The problem

ElasticSearch has no concept of user, which means that it cannot apply permissions in a search context. In that search engine, search is universal and public.

The solution

Store who has access as metadata related to that data and then use the filter functionality[3] to limit the results to only results that the user who is searching is allowed to see.

[3] `http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/search-request-filter.html`

4. Limited context search

The problem

ElasticSearch has no sense of context and no semantic search layer.

The solution

When searching for something in a given context, the program (for example, the ECM) adds more parameters to the search terms so that, based on those parameters, the results can be filtered in order to only contain information in the wanted domain.

5. Getting results from multiple sources

The problem

Each index has its own search engine and you may only select a single index for search each time

The solution

Use the alias system so that an alias includes two or more different indexes then search using that alias index in order to search in multiple indexes at the same time.

6. Searching for documents that are in a specific language

The problem

ElasticSearch has no way of knowing the language that its data is in.

The solution

Add the language of the data as metadata and make it searchable. When searching the program (the ECM, for example) will add the extra parameter to the search terms to force the proper filtering of the results.

3.2 Enterprise Content Manager

If using just as backend, which one is best?

As a backend-only solution, both Alfresco and Nuxeo are quite the same with some minor differences or limitations. One of the main differences is the permissions. Alfresco has many permissions packed into roles; Nuxeo has 4 different permissions. Another main difference is that Alfresco's GUI has much more control and functionality than Nuxeo.

Workflows

A Workflow is a complex task that is executed by following a well defined set of simple tasks in a well defined order. Each task is aimed towards the system itself or a user (it can be a group and then a user from that group picks it up).

Alfresco enterprise has heavy support for a different Alfresco project called activiti [5] which allows making complete workflows (without many limitations) in a boxed-based GUI or using java (making it limited to only the computer power) and/or javascript and xml using a simplified API.

3.2.1 Selecting the most appropriate ECM

Now that ECM have been compared objectively, I have to compare them based on the objectives of my work.

In my work, the requirements are:

1. To use the ECM as an external service to store and retrieve assets.

2. The ECM must be "visible" and that a company owner can quickly agree it is a possible good solution.

Based on rule 1, the best choice has to be made using a subset of the features that both ECM, Alfresco and Nuxeo, offer.

For the general case, Alfresco seem to be a better answer than Nuxeo due to its more verbose and personalizable permissions system (that being the single superior feature of Alfresco over Nuxeo in my opinion). But then, Alfresco's free version is limited unlike Nuxeo's. So, if no official support is required from those companies, as a backend only solution, Nuxeo solves the situation quite well.

Anyhow, most companies do require professional support so that gives Alfresco some extra points here.

So, answering the question:
"Does the client require professional support from the company that made the software?"
Is what decides which one is the best choice given only the 1st rule.

The main conclusions are that:
If there is no need for support (unlikely but possible), Nuxeo answers well enough. Otherwise, Alfresco is the best choice.

Then there's the 2nd rule which states that the ECM must be visible.

For that, the ECM must be in the Enterprise Content Management Magic Quadrant Gartner [81].

According to a copy of the report [82] I was allowed to read, Nuxeo is not in the results because they require the ECM to have, at least, 10M$ (USD) of gross income. At

that time, Nuxeo was at 9.5M$ (USD) of income [17]. I was unable to get a more updated value of their income. Possibly, they will appear in this magic quadrant next year.

Even if Alfresco is the selected option, I still believe that Nuxeo is the best option because I believe that companies will have use for its complete GUI and their GUI-based system they developed to personalize Nuxeo to the company that is using it nearly to the finest detail.

3.3 Business Process Manager

In the previous chapter, I compared jBPM to Activiti for the features that I believe to be the most important that a BPM must follow to fulfill its work for a company.

In this chapter, I will continue that analysis while being more detailed and I'll give more emphasis to features that are the most important towards the objective of this work.

I wasn't going to start completely from zero. While studying and comparing ECM, I had read some information about the BPM Activiti [5] [3] whose engine is embedded into Alfresco. Given this, I was going to search for alternatives to this BPM.

As time went on with the project, I got into multiple tasks being made in parallel.

1. Make a workflow (that has around 25 variables (which some may contain multiple values)) based on a simple real life workflow structure.

2. Verify how service tasks work on jBPM.

3. How to deal with communication with an ECM using CMIS (inside a service task).

The main reason why all these accumulated is explained below.

To make a workflow 1, I made the bpmn diagram based on the bpmn file and image that we were given. Designing the diagram based solely on those two elements revealed to be an easy and straight forward task. The issues arrived when I had to finish up the service tasks and the variables that would contain multiple elements (between 0 and unlimited).

As for the service tasks, I took a lot of time trying to get them to actually work.
For starters, I tried searching and following the manual [45] about the subject. To my disappointment, there's no such thing in the manual. The nearest thing I have is the explanation of the service task [42] but nothing about how to actually make it work. With that I was forced to be creative.

As my first creative step, I went to the option that seemed the most obvious. I was using git to send the .java files directly to the project inside jBPM's git repository (where all the workflow files reside), in the correct corresponding file path. As jBPM file paths rules coincides with maven's filepath rules, the placement of the files become trivial.
Unfortunately, it failed. No matter how I was placing the .java files, the workflow engine was never recognizing its contents. What was keeping me moving was one small detail:

Every time I tried to place the file with syntax errors (to confirm how it is working), jBPM would always complain about the syntax errors in the .java files and would fail to build and deploy. That was giving me the notion that those files had meaning. I also tried placing .class (java bytecode) files and .jar (java archive file) using the same conditions. Also no results.

After many, many hours trying that and the alternatives I went on and searched for the solution. On the file system, under the standalone directory (where the standalone version of jBoss is) there's a directory "lib" with a subdirectory "ext" a search result also helped delivering such impression [76] and it is also potentially backed by a docs element [39] where, in some interpretations, the statement is true. That seemed trustworthy enough as it was completely empty and in works of this kind, the a directory "lib" usually means "place your executables here". So I tried again there. The result was the same.

Lastly, after a question in the IRC (jbpm@freenode.net) I received confirmation that the only way to have it working is by altering the .war jBPM deploys and by adding my .jar with the .class inside WEB-INF/lib directory (that is inside the .jar file).

This means two things:

1. Every time I need to update jBPM, I have to alter the package I get in order to accommodate with the additional personalized content that I have in jBPM.

2. If different deployment versions require different version of classes with the same name they become unusable. Instead different versions require different class names (the difference may be exclusively on the package part of the name without changing the simplified name)

Because of that, solution ends up being a nonsolution. Instead of using jBPM as a complete black box and to have jBPM importing our content into its internals.

Activiti has the same situation among other problems that jBPM does not have and vice-versa. Unfortunately, I was unable to investigate that more deeply because I was taken away from activiti.

Or so I thought until the accounting section called our attention where I was back into working with Activiti and jBPM. With this, I was able to work with both to see which one is the best (continues on next chapter).

Chapter 4

Design

I was asked to make something that would allow joining together Alfresco and jBPM. According to what I understood, I had to investigate jBPM and Alfresco's interfaces and see how to use them.

4.1 The BPM comparator project

The requisites I was initially told were very simple. It is a piece of software, made in java, that provides an interface towards the users for the BPM and ECM backend engines. It was supposed to provide a login interface, a user task list and means of making personalized forms for the human tasks. As for the service tasks, strategies would be worked on so that the BPM engine could talk to the ECM as required to send or retrieve assets. The communication with the ECM is made using the CMIS interface with the idea of keeping the 3 components as decoupled as possible. The interface never communicates with the ECM (in both ways). When assets are required from the ECM to the user, it is the BPM's job to get them from the ECM and then deliver them to the interface java program when the user task is requested.

Given that, I searched for a good implementation of a CMIS client. Unfortunately for me, I was only able to find a single implementation of CMIS. I tried to get more implementations but I was unable to find them so I assumed that openCMIS [66][1] is the best open source client side implementation of CMIS protocol.

In order to get a grasp on the library, I asked (and was told) that the only operations that were required were summarized as GET; PUT; PEEK of files. Keeping that in mind, I made some code that explored those characteristics of the CMIS. After some extra glancing, I made some modifications and adaptations and then, that allowed me to, at the same time, understand the ideology behind how CMIS (and openCMIS) works and to gather a

[1]CMIS is a standard protocol for communication between an external program and an ECM. In full, it allows executing the most common tasks that can be made with files including putting (and assigning metadata), getting and deleting any assets.

useful sub-library for the, apparently, most common operations made when dealing with files and with CMIS.

At that moment, only jBPM [46] was the selected BPM engine for the job. So, after I was done with the CMIS library, I started working on a jBPM interface between the program I was making and jBPM.

To begin with, I went on to find the documentation related to the remote API for jBPM. For someone that is not familiar with it, it is not explicit on how that API works. But still, I was able to understand that there is a java implementation that takes care of the communication details and that I only need to use that same documentation if I'm developing in java.

4.1.1 About the documentation

jBPM's documentation is available online and it explains a lot on how to use jBPM and each part of the KIE workbench [80] what many things are and how functionalities work. There's a catch, though. There's almost no information on how to use their java remote API [41] which is meant to be the best alternative to using their REST API directly. The only useful information is an example code for establishing a connection with jBPM (which is just introductory information), nothing else. This API is made of:

- Session manager → Allows accessing user information
- Task service → Allows accessing jBPM's tasks
- Factory → After giving a username and a password, it returns the session manager and the task service

The main problem with this API is that there's no such thing as documentation explaining each thing. Instead, there's a method list with the method names and parameters. If this was not enough, many of the methods are not fully clear of what they mean if one is not familiar with the internals of jBPM. For example:

getTasksAssignedAsPotentialOwner(String userid, String language) [44]

Then I question:
What is a potential owner?
Trying to respond myself, seems like it is a task that has been assigned to a group (or available for anyone to take) that the corresponding use is able to be assigned.

With that idea in mind, I tried that theory. I made a workflow with a task that would assign the task to a group that the user belongs to.

When I called that method, the result went as expected, the task was listed.
Then, I claimed[2] to the user.

[2]Claiming a task is a mark that allows other system users know who is going to execute it.

The task was still being listed.

Puzzled, I gave up on that theory and moved to the next one.

New theory: A user is still considered potential owner until he starts the task assigned to him.

In order to try this one, I re-used the working elements of the previous and I started the task.

The task was still being listed!

Then I went wondering what the actual meaning of "potential owner" to jBPM is.

As a last resort, just to make sure that I was thinking right, I went to their IRC[3] channel and I asked for help on the subject. From the answer I got, the tasks listed are the ones that are available for claiming (or that are already claimed) by the user asked and are completable (i.e. not complete, not aborted, claimed, started, etc...). That took care of that matter. So, for the application I'm developing, this basically means that it is perfectly safe to use this method as the means to get the tasks list for the requested user.

Now, also for that method: Why do I have to mention the username[4]? Didn't I just mention it when I gave the username and password? Or is it that I'm capable (if I have permissions on my account) to get information from other users?

Another test. I prepared a list of tasks for an admin user with all permissions and I made a second user with the user role permissions[5]. Then I tried logging in as both user and administrator and trying to get their tasks as the control and then each other's tasks. Interestingly enough, trying to get the other's tasks results in the method throwing a RuntimeException[6]. Later, when I was paying more attention, I able to understand that it was not exactly **a** RuntimeException, it was **the** RuntimeException!

So there are two things here:

1. The first parameter seems to be there for no reason at all.

2. If I do something it considers wrong, I get an Exception.

As for the first one, I was lucky someone asked in the IRC[7] right about when I was also about to ask why this happens. Here's the question and the answer (I altered the asker's username):

```
<[asker]> But why then specify the user again in the call?
<krisv> [asker], we reuse the local api
```

[3] `irc://chat.freenode.net/#jBPM`

[4] see parameter userid in previous method.

[5] User role permissions means that the user may only list and execute tasks and start workflows.

[6] The reason I was unable to perceive it was **the** RuntimeException is because, with the experience I got, I got used to skip all the package namespacing directly towards the last name which is the class name. By convention, exceptions have a name ending with "Exception", so I was just reading searching for the last occurrence of that in the immediate text.

[7] `irc://chat.freenode.net/#jBPM`

So that explains why there are inconsistencies between multiple methods where I have to re-specify the username... Ok, so moving on.

While testing other parts of the JBPMDriver (the class I was making to make the mid standpoint), I started testing illegal operations. Between multiple different tests, I was not fully paying attention to the resulting exceptions when I was doing it wrong.

For example, I tried claiming a task that I had already claimed, this was appearing. but I was reading as: This was not only me. Others that I had shown this were also reading the same way as I was initially, except that they were having a much harder time believing and accepting what was in front of them.

The exception being thrown was RuntimeException itself! The actual information that was useful to diagnose and process was not in the stack trace, instead it was in the message of the Exception instance.

In result of that, I went back the IRC to try to understand why that happened. Here's the significant parts of the information exchange (I reordered some messages to help understanding the conversation flow):

```
<brunoais> Why does TaskService.claim() throw a
 RuntimeException when the clam fails?
<brunoais> Shouldn't it just throw an UnauthorizedException or
 better, a PermissionDeniedException?
(...)
<mriet> brunoais, in other words, you're probably right, but
designing an API is harder than it looks -- when whoever
coded that method decided on a RuntimeException, it
probably made sense.
(...)
<brunoais> but then all the cause Exceptions are LOST
```

It was between these two exchange of words that I got to know that mriet is the main designer and programmer (but not the lead programmer and designer (see below)) of jBPM's remote API.

Later, after some exchange of words, he explained me the situation:

```
<mriet> the remote Java Api is basically a client that
 sends requests to (and processes the responses
 from) a remote REST server.
<mriet> the question is: if you have a client that is processing
requests and responses, what's the correct exception to
throw when the something goes wrong with the request
or the response?
<mriet> when i was designing/writing the code, there were 2
```

```
 things I was thinking about:
<mriet> 1. how to show that the client had failed.
<mriet> 2. whether it would be helpful to fail "silently", fail
 with logging... etc.
<mriet> 3. also, the client implements a "hard-coded" interface.
 In other words it implements an interface that I can change
 (KieSession, TaskService).
(...)
<mriet> (I was initially against implementing the KieSession
interface and the TaskService interface as they were because
of this problem. (...)
```

As for the exception, he mentions that he only found 3 options to deal when something goes wrong.

```
(...)
<mriet> 3. so the only thing I could do is fail "big", with a
 runtimeexception (that I then of course didn't have to modify
 the interface for, because it's a runtimeexception)
```

And he went for the 3rd with a RuntimeException.

Personally, I think that all 3 are wrong but I didn't really know what were the limitations here, so I made a suggestion.

```
<brunoais> Option 4: Throw a RuntimeException with the stack
 of "caused by" in which that stack contains the correct and
 full exceptions that happened inside it.
```

Not an option, unfortunately.

(You may find an uncut version in the attachments section **??**) I confirmed myself. The transformation is made directly in the server in such way that it is virtually impossible to properly reconstruct in the client.

That lead me to create a ticket in their tracker `https://issues.jboss.org/browse/JBPM-4251`. Here I report the situation and I recommend making a class that extends RuntimeException and using that same class to report the exception instead of using RuntimeException itself.

Later, really near the end of the project, the version 6.1.0.RC was released and that feature I had suggested had been implemented.

As for my solution, I decided to make a try/catch specific for that call where I re-wrap the exception as a cause of an exception that I own and that is checked. Then I throw that new exception. Like this I'm sure that some code will take care of that exception some time in the call stack without being forgotten.

Even with the list of operations available with the library mentioned above, there are operations that are not available through the library mentioned above. This is one of such requests [48]:

```
/runtime/{deploymentId}/withvars/process/instance/{procInstId}
```

This returns a JaxbProcessInstanceWithVariablesResponse. Now, what is a JaxbProcessInstanceWithVariablesResponse? I have absolutely no idea. No manual says what it is or how it is formatted except a small clue:

> Returns a JaxbProcessInstanceWithVariablesResponse that contains:
>
> - Information about the process instance (with the same fields and behaviour as the JaxbProcessInstanceResponse
> - A key-value list of the variables available in the process instance.

Ah, ok. So a JaxbProcessInstanceWithVariablesResponse is a set with a JaxbProcessInstanceResponse and a list of key-pair values. The key-pair values, is probably an implementation of a hash lookup with a string as the key and some value. There's also the JaxbProcessInstanceResponse. What is a JaxbProcessInstanceResponse? That too is not defined. I can read the source code of it but that tells me nothing about the actual output format. Even for those key-value pair, it can be made in multiple ways! XML is very flexible but, for program interfaces, things defined like that do not help at all. Besides, even if I successfully test and get to a specific format for this version's output, what is there to guarantee that the format of the output will be the same in the version after? It can be that, later, it is found that it actually makes more sense to format the XML differently or that they found a way that uses less bandwidth? If so, what then?

Based on those, I was left without ways of knowing how to solve that dilemma.

With a class that relates the interactions with the BPM and a class that relates the interactions with the ECM, I moved on to make the code that uses those two to work. So I moved towards the interface.

4.1.2 BPM comparator organization

For the interface, (as mentioned) I decided to go with JSP for being very simple to work with, for being stateless and for making everything a list of very small simple problems. It was decided that it would be only made of the login, a task list and the forms to complete the tasks, the system was thought like this:

1. index.jsp → The index page. It contains the menu on the left size for the options. It never got anything usable outside getting to the login screen or the list of tasks.
2. login.jsp → The page with the login form.

3. tasks.jsp → The page with the list of tasks.

4. executeTask.jsp → The page where the form for the specific task requested to be executed appears.

How executeTask.jsp would work

The user task (which relates to a form) to be executed is identified by the task id. In order to identify the form that appears for a specific task, the task name (as defined in the workflow) and the full deployment id are used.

In this program, the path is built based on the deployment id identifier as specified in jBPM. A deployment is composed of 3 parts delimited by a column(":"). The first part is the group id, the second is the artifact id and the third is the version.

In this idea, the form for a task is composed of a .html file that is custom made depending on the specific needs for the task that is being done by whoever is answering it. After answering the form, the server would gather the information from the form controls that the user had submitted and then it would use that information to send to jBPM. The .html file contains the HTML that is placed directly inside the executeTask.jsp including all relevant javascript and CSS used while answering that task.

Each .html file, is accompanied by a brother .json file which specifies the names of the fields that are relevant and it also specifies where in the ECM the files submitted are sent to.

Now the json file is more than just specify where files are sent to, it now needs to specify much more information. Also, the plan for this client needs to change a lot as it also needs to process service tasks somehow.

4.1.3 Managing ECM using BPM

After many many hours trying to find a way to have jBPM communicating with the ECM using CMIS and finding a way for it to get the assets required so that they would display to the user, it was just not making sense. I was using too much time to do a task that is supposed to be trivial, specially due to being as inexperienced in the field as I was.

While I had been working on this, I had already developed a really early demo that was showing the communication between the interface towards the BPM which was already doing remote commands to the BPM (although it wasn't reading anything from the BPM yet). Additionally I asked for advice from my colleagues if the whole system organization was being done correctly and according to the way that it is supposed to be done (by-the-book) and based on their experience.

Unfortunately, the advice I had in response meant I was wrong. The way of doing integrations like this one, is to treat each of the applications being integrated as pure black boxes with a "well defined" API and my application is the one responsible for making

the connections between, in this case, both ECM and BPM programs. BPM would take care of the workflows and everything directly related to it and the ECM would work exclusively as a file repository.

Ok then... Change of plans.

Starting with the easiest. Alfresco. The CMIS mini-library I had made was already doing most of the work except it was not ready for the interface to talk to it. It only required some minor changes for the adaptation.

As for the jBPM (BPM) communication, it is now much easier and direct because I don't need to work on large data transfer between the interface, just small data that is used for control.

4.1.4 New BPM inclusion in the project

I was introduced to Activiti[8] and it was decided that our section in Novabase would investigate and compare jBPM and Activiti.

With this new requirement, I was happy that what I was developing was already quite prepared for such occurrence, instead of having to delete and write a lot of new code. The decoupling of the part dedicated towards the strictly related to the BPM communication and the internal original pieces of code made this task rather trivial. What took longer here was just analyzing both BPM's REST api and find a common interface for both.

Thankfully, while I was still analysing both BPM's API, I was introduced to a new colleague. He would be investigating with Activiti and I would continue my investigation on jBPM only.

For the next week, each one investigated in his own assigned BPM and we both worked on deciding the common BPM interface that would be used for BPM communication.

It is supposed to test and analyze with Activiti and jBPM at the same time and shows side-by-side execution between both platforms. The main purpose is to compare each one's limitations and features.

After an initial study on Activiti, me and my colleague discussed together how a common interface for jBPM and Activiti would be. As we went through we noticed that, although they use fairly different interfaces, both make available just about the same information about a user task.

Based on that conclusion, a new Interface (BPMTaskSummary) was born. This interface exposes all information required to display to the user while listing existing tags waiting to be fulfilled.

[8]It worked more as "re-introduced" because I had already had some minor research while studying Alfresco as Activiti is integrated in Alfresco.

4.1.5 Finding out how to make custom tasks in jBPM

According to the guide [69], a good example is the "Customer Relationship example" from the jBPM-playground[9] repository.

Making the example work was quite straight forward. It was mostly copying and pasting files, following the guide. My main issue was finding where WEB-INF was. I had searched nearly everywhere inside the directory that I had unzipped from jbpm-install-full.zip. Everywhere, except inside the jbpm-console.war file which appears inside *jBPM-installer/jboss-as-7.1.1.Final/standalone/deployments/* after running the installation script.

Out of all my trial and failure and my findings, here are the most relevant ones:

Adding users In order to be able to work with workflows that include groups, such as the one mentioned above, I have to add users and groups to the system. There are two ways of adding users. They can coexist but using both for the same user yields undefined results.

First one, is using the *users.properties* and *roles.properties* files. They are formatted just like any other. properties file[10]. Personally, I recommend using this method for development because it is fast and easy but I seriously do not recommend using it for production because all data is stored in plain text.

users.properties stores the users and passwords in a key-value fashion. The username is the key and the password is the value. All data is in plain text.

roles.properties stores the users and roles/groups in a key-value fashion. The username is the key and the groups/roles are the value. Groups/roles are delimited by commas (",").

After running the installation script, you may find these files in *jBPM-installer/jboss-as-7.1.1.Final/standalone/configuration/*

There's an undocumented feature here. In the root directory (jBPM-installer), there's a directory named "auth". In there, lies both files I mention above. These files are copied to their correct place every time the installation script is run and they **replace** the previous ones.

With the custom tasks problem solved, I was now able to execute arbitrary code in the BPM.

The next thing that needs to be done is how to do an easy way of generating an interface for the programmers who create new user tasks. The HTML files are in place and they are already being imported to the application when the time is right. What is left now is to find an easy and direct way to define the rules of each user task.

[9] This is a repository that, by default, is automatically downloaded by the installation script that comes with the jbpm-install-full.zip on sourceforge[40].

[10] Explaining what a properties file is outside of the scope of this document. This wikipedia article should help `http://en.wikipedia.org/wiki/.properties`.

BPM comparator client user task setup

As part of the BPM comparator's functionalities, it requires being able to execute send and receive information from the ECM In order to define where to get the each piece of data required for a user task to run, I started to define a structured JSON file. For this project, at this stage, its specification had never become stable.

4.2 Using complex frameworks

Learning primefaces

In order to follow the programmer's praxis, I started with the "hello world". I started with a guide Novabase had given me. Although it helped me about the setup and as a simple "hello world". It didn't teach me anything about how to program using jsf/primefaces.

Moving on, my colleague was trying to learn primefaces by studying their examples list on primefaces' website [68] so, as my previous attempts had failed miserably, I tried copying and pasting content and then try to make a reasoning out of it.

One of the first problems I had to face is understanding how to build a table. Until now I always built a table top-bottom/left-right, one line at a time with me having the full control of each character that is outputted to the browser for it to parse and display. The concept is quite simple here. In HTML a table is build simply as this:

```
<table class=''someClass''>
    <thead>
        <tr><th>somehead1</th><th>somehead2</th></tr>
    </thead>
    <tbody>
        <tr><td>data1</td><td>data1.2</td></tr>
        <tr><td>data2</td><td>data2.2</td></tr>
    </tbody>
</table>
```

In jsf, a table is built like this:

```
<h:dataTable value="${order.orderList}" var="o"
                styleClass=''someClass">
    <h:column>
        <!-- column header -->
        <f:facet name="header">Order No</f:facet>
        <!-- row record -->
        ${o.orderNo}
    </h:column>
```

```
    <h:column>
        <f:facet name="header">Product Name</f:facet>
        ${o.productName}
    </h:column>
</h:dataTable>
```

Where the contents to be shown have to be in a java class somewhere. Although it seems quite straight forward, there are many details here that require a lot of attention, specially if you want to really understand how to use this.

After solving the above issue, I had the "styleClass" issue. If you see above, the h:dataTable has an attribute called "styleClass". That's actually mapped to the "class" HTML attribute. I don't know the reason why that happens, it is not explained anywhere. This was a huge source of confusion on my side as I would continuously type the attribute "class" when I actually meant "styleClass".

4.2.1 The development of the project and design issues

Me and my collegue decided to go with Activiti's interface.

We started with HTML + CSS. For that part, I took the lead and I used the opportunity to teach him HTML and CSS and many of their tricks and useful gimmicks. With that we built two pages. The login page and then we build user task list page. Then we moved on to primefaces and server communication.

Allow me to clarify about using Primefaces. There are 2 great things about primefaces itself (i.e. not JSF) that I think that deserves praise.

1. Unlike JSF's documentation, primefaces' documentation is nearly complete and it is quite easy to read. My own issue is that, in my opinion, it is just not complete enough if you are not used to jsf, so it hardly can teach you anything of the actual JSF

2. Unlike JSF's code, primefaces' code is well organized and quite easy to read. When I had doubts on how some controls work and how some usages were not working as expected, its source code responded well my needs of understanding and it allows me to fully understand when I was using a functionality wrong when I thought I was using it right.

Summarizing this, the one who is most complicated for me here is just JSF, what primefaces does above jsf is somewhat easy enough for me to understand and use, specially after reading its code.

Here's an example where I had placed this in practice:

It is part of my work methodology for javascript in DOM to work with events only. Each different "module" of code talks with another one using events. Events report things

that have happened. Whoever is interested to them just listens to those events and acts accordingly. All asynchronous and decoupled.

Another extreme must for my methodology is that no javascript is allowed embedded in HTML. Javascript inside "<script >" tags among HTML is OK but it must be small as in an exception or dynamic output. All javascript goes to their corresponding .js files.

So, after having a user task list as a p:dataList [88] working properly, we started being interested in getting the value of a selection when the user selects an option. After some hours trying to understand how to place an event listener in the correct place for the list so that it would correctly capture the "change" event... Nothing was happening. Our code was not being executed. Primefaces' documentation mentions an "onchange" attribute but I seriously don't want any javascript on the xhtml file so I was using a "change" event instead but nothing was happening.

Then I tested the "click" event. For this one, it was being fired twice. The first one was the trusted click event caused by user interaction and the other one... I assumed it was generated by Primefaces as it was untrusted [89].

At first, I blamed primefaces for it. For me Primefaces was throwing the "click" event but not the "change" event. Seriously... I couldn't be more wrong! In reality, after reading its source code, I was able to find that it is actually requesting jQuery a "change" event to be dispatched and bubbled. Primefaces was doing it right!!! I was wrong to point the blames to them. In reality, the one who was doing it wrong was jQuery.

Primefaces uses jQuery's trigger() [75] method. The documentation on that page reads like this:

> As of jQuery 1.3, .trigger()ed events bubble up the DOM tree

That part would not be complete without this one:

> To trigger handlers bound via jQuery without also triggering the native event, use .triggerHandler() instead.

These are the bug reports related to it that were disregarded:
`http://bugs.jquery.com/ticket/8701`
`http://bugs.jquery.com/ticket/11047`
`http://bugs.jquery.com/ticket/15143`

So that's it. While the documentation states that it triggers native events, when we complain it does not trigger native events, they mention it is not supposed to trigger native events.

By reading the source code, you can notice there's a catch there. It will trigger the native event if there is a method in the DOMNode object with the same name as the

event name. Curious enough, "click"[11] is a method that the DOM has, so when calling $(...).trigger('click') it will call the native click event. Convenient, right? It is convenient for some uses where the event needs to be executed from non-trusted sources. The real problem happens when there's multiple instances of jQuery or the usage of the native DOM. That's where it does not work as described.

In result, I wrote a jQuery plugin that creates a custom event on the DOM when trigger() is called that follows what the jQuery's manual mentions for that method.

4.2.2 The return of BPM comparator

Building my own assignment

In order to make the work as realistically achievable as possible given the time available, I decided the best way is having assignment made in 3 parts.

The first part is made of the basic operations and, mostly, just showing the operations working. The second part completes the first part by implementing all interactions inside the system itself but without using external resources in order to function. The third part uses external resources in order to fully function.

More in detail, the three parts were made as follows:

The part 1

At this stage, the objective was to insert this workflow into the system (where the service tasks are empty code) and to have my project communicating with it and the ECM Alfresco.

In order to do this task, I had to nearly finish my project.

With that, I finally completed the definition of the *.rules.json file.

Defining *.rules.js *.rules.js is a kind of file I developed for the project which is a metadata file with instructions on how to manage the information that both ECM and BPM returns and then how to send the form submission data back to the ECM + BPM with the user inputted data.

This file can be separated into 2 major parts for each form:

1. Filling spaces and displaying.

2. Gathering data and deliver to the responsible to keep it.

As for filling spaces (1), the elements inside "inputReplacements" (see below) specify that. This is a JSON object with all BPM variable names as keys and the corresponding

[11] The methods that the DOM has that are events are "click", "blur" and "focus". Additionally, the HTMLFormElement ("<form >") contains the "submit" method.

values are the strings that are replaced with the value of the BPM variable.
For example:

If the key is like below: “replaceMe” with the value “replaceMe” and the BPM variable with the name “replaceMe” has the value “theStig”.

The final HTML output to the user will have all instances of “replaceMe” from the original *.form.html file replaced by “theStig”.

```
"inputReplacements":{
    "replaceMe":"{{{replaceMe}}",
    "fancyOutput":"{fancyOutput}",
    "ECMContentID": {
            "findInForm": "ECMOutput",
            "translatorClass": "full.class.name",
            "translatorMethod": "staticMethodName"
        }

}
```

Do note that this is just string replacements so you can use whatever you want to mark each replace.

As for gathering data (2), the elements inside “elements” (see below) specify that.

This is a JSON object with the input names as property names and a JSON object with the data specific for each input as the values by its name. The attributes “mapsTo” and “type” are always required. “path” is required and only valid if the “type” is “file”, otherwise, it is ignored. There’s also that extra parameter “CMISBaseFolder”. “CMISBaseFolder” is the “root” of all files created in the ECM. All files generated by executing this human task will be placed in a subdirectory inside the path specified there.

“mapsTo” is the variable name in the BPM. If this was a value that was gathered before and is not updated, for example, if this is to update the above mentioned “fancyOutput”, this variable’s value would be “fancyOutput”.

“path” indicates an extra file path that is used for this specific file besides “CMISBaseFolder” (explained above) and before the specification of the processInstanceID.

“type” is the type of the variable. For now the only options are “String”, “Integer”, “Double” (double precision float point number) and “file”

“forceValue” Is a key value map between the value it receives and the final value that is sent to the server. In the example below, if the user submits with the “Accept” value, it will translate it to “Accepted” and the value stored in the BPM is “Accepted” . “forceValue”’s sole reason of existence is due to an IE (Internet Explorer) bug that, for button elements, while submitting a form, ignores the value specified inside the “value” attribute and it sends the value specified inside the content of the tag.

For now, the result of having multiple input elements with the same name is undefined as it was deemed unnecessary to support.

Below lies an example of that part of the JSON file.

```
    "elements":{
        "nonExistingElement": {
            "mapsTo": "doesNotMatter",
            "path": "",
            "type": "file"
        },
        "example_in":{
            "mapsTo": "str_out",
            "type": "String"
        }
        "integerVal":{
            "mapsTo": "val_out",
            "type": "Integer"
        },
        "result": {
            "mapsTo": "opinion_out",
            "type": "String",
            "forceValue": {
                "Accept": "Accepted",
                "Reject": "Rejected",
            }
        }
    },

    "CMISBaseFolder" : "/wfcontents/registerAccount"
}
```

So far, I only found a single design flaw on this system. It is not prepared for data with loops. This means that there's no way of writing information with variable number of items directly into it. Instead, a java class is required to make the loops and output the required HTML. While it may seem a large issue, it is actually possible to make generalized classes that deal with the same kind of data and output from the BPM and then reuse them on different user tasks. This was caused by how I saw that BPM works and stores information. I didn't perceive that such thing is used in BPM based systems.

For a complete uncut version of these files see the Attachments section A. It also includes the corresponding *.form.html file for this *.rules.js file.

Chapter 5

Implementation

In this chapter, you'll find detailed descriptions of my main challenges I had during the development of the proof-of-concept projects.

5.1 1st stage

For all below, XMLHelper is a class I developed for a different project that simplifies some of the work related to XML in a remote server. It does a request to the server with credentials and then gets an XML response followed by finding the requested nodes from the response document given an XPath

5.1.1 The jBPMDriver class challenge

Reading the configuration

The system is composed of two configuration files.

1. General configuration
2. Workflows configuration

As for 1, it is stored in the initVariables inside the "context.xml" file of the TomCat server. For example:

```
<Parameter name="BPMConsoleConfigPath"
 value="${catalina.home}/conf/BPMConsole/
config.xml"/>
```

This tells the program that the config file is at the directory "/conf/BPMConsole/" relative to the tomcat home and it has the name "config.xml". If this configuration does not exist, the program stops working.

This config file contains the relative path to the workflows configuration file, the URLs to contact the ECM and the URL to contact the BPM. If the contents change, it is required to redeploy the application.

As for 2, it specifies the workflows the system has, where they are defined and how they are called in the BPM. The format is simple:

```
<workflows>
    ...
    <workflow>
        <driver>driverName</driver>
        <internalName>path/to/files</internalName>
        <jBPMName>WorkflowName</jBPMName>
        <jBPMDeployment>org:name:0.0</jBPMDeployment>
        <activitiName>activiti:own:name</activitiName>
    </workflow>
    ...
</workflows>
```

Each workflow tag has a driver, internalName, jBPMName, jBPMDeployment, activitiName.

The driver tag, contains the identifier of the BPM used. Currently, either "jBPM" or "activiti".

"internalName" relates to the path inside any of the classpath directories. The system already includes "$catalina.home/BPMConsoleWorkflows" as part of the classpath.

"jBPMName" is the name of the workflow in jBPM.

"jBPMDeployment" is the deploymentId for the jBPM workflow.

"activitiName" Is the unique identifier for an activiti workflow.

The directory structure is calculated from the following mask:

```
{baseDirectory}/{internalName}/{workflow taskName}
```

The "{baseDirectory}" is all directories that are in the classPath of the program. By default, the "$catalina.home/BPMConsoleWorkflows" is included in the classpath and is the recommended place as the root for all files directly related to the workflow forms.

Getting the files required for an execution

Getting the *form.html and *.rules.html is not a complete trivial task. They are stored in a directory that relates to them.

In this system's case, they need to be stored in a path relative to all classpath directories. That path is setup in the workflows configuration file. You can see the full explanation on how that file works and how it is used in the previous chapter. Its contents are

stored in a global class ExternalNameToInternalNameTranslator which, given the unique identificator for a workflow, it returns the path to the task's files (the form and the rules file). This is then used by each of the responsible parts of the program for each required file.

This is how I get the path to the files.

```
BPMCommunication driver =
 drivers.get(task.getDriver());
BPMTaskInfo taskName =
 driver.getTaskname(task);
String internalName =
 externalNameToInternalName.get(
 task.getDriver(),
        taskName.getDeploymentId() +
         SEPARATOR_CHAR +
        taskName.getName());
return internalName + "/" +
 taskName.getTaskStep();
```

... which I use, later, to get the corresponding files.

In this case, this excerpt is from "executeTask.jsp"

```
String baseFormPath =
 manager.getFormPath(tasks.get(taskNum));

String formPath =
 JSPUtils.getResourcePath("/" +
  baseFormPath + ".form.html");
String cssPath =
 JSPUtils.getResourcePath("/" +
  baseFormPath + ".css");
String jsPath =
JSPUtils.getResourcePath("/" +
 baseFormPath + ".js");
```

This JSPUtils is a simple method that translates the relative path to an absolute path and returns the absolute path to the file.

... And this excerpt is from processManagement.java

```
    URL taskFileURL =
     ProcessManagement.class
     .getResource("/" +
```

```
  getFormPath(taskSummary) +
   RULES_FILE_EXTENSION);
if (taskFileURL == null) {
    throw new
     FileNotFoundException(
     Utils.buildWfFileNotFound("/" +
      getFormPath(taskSummary) +
       RULES_FILE_EXTENSION));
}
String rulesData =
JSPUtils.loadFile(
        new File(taskFileURL.getFile())
        .getAbsolutePath()
        );
```

Getting the deploymentId of a task

In jBPM, a workflow task is uniquely identified by deploymentId + workflowName + taskName. A user task is identified by deploymentId + taskName.

jBPM has the function discussed above called "getTasksAssignedAsPotentialOwner()" which returns a list of tasks for the user. Each user task returned, is of type "TaskSummary" which is an interface. In the interface definition, it states that there is a method called "getDeploymentId()" which, supposedly, returns the deploymentId of the workflow task that this user task relates to.

In both jBPM 6.0.* and jBPM 6.1.*, it always returns null..

When claiming a task, executing a task, etc... I need to provide the deploymentId of the workflow to where the task relates to, otherwise, the server will not accept the request. In other words, there is no direct way in jBPM 6.*'s remoteAPI, that allows:

1. Get user task list.

2. Select a task from the list.

3. Using the task selected, do an action with it.

Instead, it requires a workaround. My first workaround is to try to get the task data and then get the deploymentId from there.

```
return taskService
.getTaskById(task.getId()).getTaskData()
.getDeploymentId();
```

In jBPM 6.0.*, it was returning the deploymentId as stated. So it was OK for some months as I kept developing. Then I updated to jBPM 6.1.*. It stopped working. Instead of the taskData, I was getting null (so I couldn't ask for the deploymentId). Questioning in #jBPM (IRC) didn't produce any useful results, so I went to a new workaround.

Thankfully, the REST API, which responds in XML, has a call that responds with the same information that "getTaskData()" is supposed to provide. With that point, the previous code became this:

```
TaskData taskData =
 taskService.getTaskById(task.getId())
 .getTaskData();
if(taskData != null){
    return taskData.getDeploymentId();
}
return
 manualGetDeploymentId(
 ((jBPMTaskSummary) task)
.getTaskSummary().getProcessInstanceId());
```

... And manualGetDeploymentId():

```
XMLHelper helper =
 new XMLHelper(url.toString() +
  REST_PATH + "/runtime/" +
   EMPTY_DEPLOYMENT +
    "/history/instance/" + processInstanceID);
helper.load(username, password);
String deploymentID = helper
.getNodeList(
            "//process-instance-log[@id='" +
             processInstanceID +"']//external-id"
            ).item(0).getTextContent();

    return deploymentID;
```

The XPath above was built based on example queries and by analyzing their results. There is no documentation on the format of the XML that the API uses (see previous chapters for more information on this subject)

5.1.2 The *rules.js file

As part of my development, I thought that a definitions file for each user task for each workflow would be the best approach to the situation. It is the most versatile way of

personalization on what is done with the information I gathered of the use cases for the developers.

The *rules.js file challenge

The variables substitution As I noted above, the inputReplacements JSON key internal implementation is just simple string substitutions but they cannot be done in any way. Doing in the wrong way means breaking the output which is not the wanted result.

The naive way of doing this would probably be using the String's replace() method for each word mentioned and that's done. It's no use to even try that. If one of the resulting substitutions include a substring meant to be replaced by a different value. E.g. If "replaceMe" is executed before "fancyOutput" and the result of executing "replaceMe" includes the substring "{fancyOutput}", that instance of "{{{replaceMe}}}" would be replaced by the value of "fancyOutput", which is definitely not the desired outcome.

An alternate approach, then, would be searching the whole .form.html for all instances of all the replacement variables, store where each substring is and then replace the multiple elements in the string. This also does not work.

E.g. A string starts at 5 and ends at 10 and another string starts at 15 and ends at 20. By replacing the string at 5 to a string of size 10, the other string will now start at 20 instead of 15. It's true you could keep track of those changes but that would be a huge mess to make sure you correctly keep track of those information.

As for my solution, for each variable in the BPM received that is defined in the *.rules.json file (a key of the "inputReplacements" object) it finds all instances of the corresponding replacement and stores the location of the first and the last character of the substring to be replaced along with the string that replaces it in a list.

After all variables have been identified, it sorts the list by the position of the first character for the substitution.

Then it replaces all strings in reverse order (from the end to the beginning of the output). The end result is what one would expect from a token replacer.

Non implemented features

Although I left this with a structure to make it easier to extend, some features that were planned have not been implemented. Here's a non-extensive list of them:

1. Allow executing arbitrary code before and after the user task.

2. Only the variable types "String", "Integer" and "Double" are accepted, other types were not implemented.

3. Enforce server side validation. It was deemed not necessary for the example but it can always be done with some tweaks to the code and by extending the object

inside *.rules.json file

How to get variables out of the BPM

The first surprise I had when trying to get variables values' from the BPM is that there's no such method in the remote API library. The manual states it returns a JaxbProcessInstanceWithVariablesResponse. For detailed information from the design perspective see the previous chapter.

There's a particular thing about the evolution of JaxbProcessInstanceWithVariablesResponse. It's definition for jBPM v. 6.0.* and v.6.1.* is different. Although slightly different, both are mutually incompatible.

In the 6.0.* series, a JaxbProcessInstanceWithVariablesResponse has a format like this:

```
<variables>
...
    <entry>
        <key>VariableName</key>
        <value>VariableContent</value>
    </entry>
...
</variables>
```

In the 6.1.* series, a JaxbProcessInstanceWithVariablesResponse has a format like this:

```
<variables>
...
    <entry key="VariableName"
    class-name="full.class.Name">
        VariableContent
    </entry>
...
</variables>
```

That means that, depending on the version, the same name means different things. This also means, to me, that a name and the thing it relates to is not 1:1 for different sub-versions. Only for the same version.

Here's how I do it.

```
        XMLHelper helper =
         new XMLHelper(url.toString() +
```

```
  REST_PATH + "/runtime/" +
   deploymentID +
    "/withvars/process/instance/" +
     processInstanceID);

helper.load(username, password);
NodeList variables = helper.getNodeList(
            "//variables/entry"
            );

 Map<String, BPMResponseVariable> variablesMap =
  new HashMap<>(variables.getLength());

for (int i = 0; i < variables.getLength(); i++) {
    Node entry = variables.item(i);
    NamedNodeMap attributes =
     entry.getAttributes();
    variablesMap.put(
        attributes.getNamedItem("key")
        .getTextContent(), new
         jBPMResponseVariable(
         attributes.getNamedItem("key")
         .getTextContent(), attributes
            .getNamedItem(
            "class-name").getTextContent(),
             entry.getChildNodes()
             .item(0).getTextContent()));
}
return variablesMap;
```

5.2 2nd Stage

5.2.1 The interface

DOM events not firing

Note: You may find the full code in the attachments section B.
This is made on 3 parts.

1. I have to ensure that the events thrown in jQuery were being properly detected.

2. I have to ensure that jQuery wouldn't successfully listen to the DOM's version of the event.

3. I have to ensure that jQuery wouldn't re-dispatch the event when I do it in the DOM.

Breaking it down,

1: First thing's first. In order to be able to place the event on the DOM, I need to know it is being dispatched somewhere. In order to achieve that, I looked into how jQuery plugins are made and how they work.

In a nutshell, jQuery plugins usually work by adding functions or by replacing in its prototype.

In this case, what better way is there to know when an event is being dispatched than to know when the method that dispatches the events themselves is called, right? With that in mind, I gather the original method from jQuery

```
...
var originalTrigger = jQuery.event.trigger;
...
```

... which I use inside directly inside my own function

```
...
var extraOnTrigger = function(event, data,
 elem, onlyHandlers){
    originalTrigger(event, data, elem,
     onlyHandlers);
    ...
```

... and then I replace jQuery's original method with my own.

```
...
$.event.trigger = extraOnTrigger;
...
```

This means that a call to this method will result in:
callee → myFunction() → trigger() → myFunction() → callee
Instead of:
callee → trigger() → callee

Given that, I now know every single time an event using jQuery is dispatched. So this part is completed.

2: Preventing jQuery callbacks from detecting the event I'm sending to the DOM.

At first glance, this will sound something tricky or just about impossible, except that jQuery already has this mechanism in place. There's an attribute in the jQuery event object called "triggered" (full name: "jQuery.event.triggered") which stores a string. What

happens is:
If an event with the same name as the value stored in this attribute is caught in jQuery, it is ignored by jQuery's DOM event listener methods.

Using this attribute bares a small risk, though. It is not documented. I only know this because I read the relevant jQuery source code regarding this attribute.

It still does help a lot by solving this issue in one go allowing me to do a simple:

```
jQuery.event.triggered = type;
elem.dispatchEvent(evt);
jQuery.event.triggered = undefined;
```

evt is a dispatchable event object

As a "bonus", this also solves my issue of not letting jQuery call the callback functions for each event more than once for each dispatched event (3) is automatically solved with that parameter in the "event" object.

Note : With this method, the event is dispatched inside jQuery before being dispatched in the DOM.

Chapter 6

Conclusions

For this work, during 9 months, I searched, studied and compared different open source alternatives to closed source Enterprise Search, ECM and BPM. Resulting in a total of tree Enterprise Search, two ECM and two BPM which I evaluated and compared.

In the following sections, I'll display my own conclusions on each one of the programs in its own category out of the three ones (mentioned above).

6.1 Enterprise Search

I found three major Enterprise Search programs (four, if you count in lucene).

Just to remove the most obvious from the equasion, **Lucene** is a really low level key-value and full text search engine which requires a program to use it. It provides no remote interfaces (remote API) nor connectors to communicate using a file system or anything of the same kind. It acts as a library for other programs to use.

So, first up: Daimon Search.

Daimon Search was the last one I found of the open source enterprise search. It was formely closed source and it became open source at the year of 2013. At the time I investigated it, it lacked the developer manual and full comments on its code and the only documentation it had easily accessible was its user manual. On the up side, It has an extensive user manual, comes with a crawler, the administration panel is complete for all functions and it already has an uncostumizable, as far as I could understand, web interface for its users.

In my opinion, due to the lack of documentation for developers, this project still needs to mature more until it can be considered as "reliable enough" for a company.

Then there's the other two search engines. For these two, there is no "X is better than Y". It all depends on how you want to work with them and which community you prefer. Here are the two programs:

Solr is an interface and manager for lucene. It is a general purpose program used to store and locate information. It comes with a simple control panel, integrated users (and permissions) system, remote API and the search engine itself. As far as I could tell, the full manual is quite complete for both developers and users. Search is made using a single query string using special characters and sub-strings to personalize the search. Results can be in different formats such as JSON or XML. More recently, Solr has become distributed with the help of ZooKeeper.

From the user standpoint,

ElasticSearch is simply a distributed JSON object putter and getter with verbose possibilities of searching and getting exactly what is wanted. It brings no ready to use user interface, only a minimal one to try it out. ElasticSearch provides an extensive search API as powerful as Solr except it is writtin using JSON anotation allowing it to be more structured and human and machine readable.

For all features that Solr doesn't have, it is possible to have ElasticSearch providing them by storing extra metadata with the data itself and then enforcing the use of that same metadata when searching. ElasticSearch is as real time as physically possible, Solr is a bit slower on that aspect as it takes some time until results are available in searches after an element is "put" in the swarm.

Out of Solr and ElasticSearch, there is no definitive winner. It depends on how the user wants to use the program and the enviourment where it will work. ElasticSearch has proven to be more ready to be working in distribution but it doesn't come with some features that Solr has such as users and permissions.

With all in account, in my opinion, ElasticSearch is the best option due to its simplicity, robusticity and when working in a distributed enviournment.

6.2 Enterprise Content Management

I investigated two major enterprise search programs.

First up: Alfresco.

Alfresco was the first ECM that I investigated. For an open source ECM it actually is quite complete with all the basic features and some extra useful features that companies like to use. It is costumizable using XML files. It comes with a partly personalizable interface. It is served in a free community edition or in a payed enterprise edition. The community edition is feature incomplete. For example, It doesn't include distribution.

Nuxeo is a ground breaking open source ECM. Just like Alfresco, it is feature rich and quite complete and highly costumizable. The permissions system doesn't have roles and

it has only 3 permissions for each user or group: read, write and manage. Unlike Alfresco, Nuxeo comes in a single fully featured package. Additionally, for paying costumers, besides the professional support, Nuxeo includes a payed platform (included in professional support) that allows, using a drag and drop and textbox GUI, to easily and quickly costumize Nuxeo.

Alfresco mentions that **Nuxeo** is not its competitor, in an e-mail I got as an answer but I don't believe that is true. They have nearly the same features. The differences are limited to details. It doesn't mean that those details do not matter, some do! Two of the main ones are the permissions system that is simpler in Nuxeo and more complete in Alfresco and the mesh asset viewer, also known as DAM (Digital Asset Management) in Nuxeo that doesn't exist on Alfresco.

In my opinion, if only using the ECM through CMIS:

If no distribution is required: Alfresco has more permissions allowing more flexibility so it should be the best option.

If workload distribution is required: Nuxeo free version should be able to answer the necessity.

If also using the interface that comes with the ECM, Nuxeo is the best option as it contains some more features than Alfresco, including the very useful DAM viewer.

6.3 Business Process Management

The decision was to use jBPM.

At that time, the latest stable version of jBPM was 6.0.0. It had multiple signs of "rushed product". Documentation was incomplete, there was no documentation in the java interface files, many icons were missing, basic functionalities were missing, and the list goes on and on...

Some months later, 6.0.1 was released. This one acted more like an alpha or beta version of the actual product. Many new icons were added (in a maintenance version?!?) and some new minor features too.

By the time 6.1.0.CR1 was released, I was nearly at the deadline of my trainee program. In my opinion, this version is what 6.0.0 should have been in the first place. It still had noticeable bugs but they were minor or hard to find. The documentation is more complete, the java interface documentation is quite clear and quite complete and many functionalities work well and as expected.

In retrospect, it would have been better to choose an older version (5.4) which was already used in other projects of Novabase and benefit from better support and an existing codebase.

After being introduced to **Activiti**, I can, then, compare jBPM and Activiti.

Both jBPM and Activiti have a very rubust and, mostly, bug-less BPM engine. I was

unable to find any actual bugs in processing workflows but the cause can easily be due to not making complex workflows.

While Activiti only has service tasks, jBPM has custom tasks and service tasks. I don't know why jBPM still has service tasks as they are better in every way towards custom tasks, I suspect it is due to backwards compatibility or just to provide a wider bpmn2.0 compatibility.

jBPM includes a git repository to where all workflow files are placed and versioned, unfortunately, no own code files may be placed there. Activiti has no real versioning for the workflows. A workflow only exists if there's, at least, 1 execution instance or if it is the latest version.

Unfortunately, both suffer from the problem that custom code has to be added to the package .jar which means that every single time one updates to a new version, one also has to alter the .jar of the application with the personalized code and all other files.

Finally, jBPM's interface is heavy and a bit too slow but it is well organized. Activiti's interface is fast and doesn't have much color but some menus may be hard to get the hang on and may cause someone to be lost in them and want to start over. Unfortunately for both, none are easily personalizable. From the outside perspective, it's easier to make one yourself and use the remote API than to change theirs.

jBPM's community was really well recieving and I got really good help at IRC when I was having truble with undocumented behavior or just exceptions with huge stack traces that I couldn't understand what the actual error was. With Activiti... Help takes a long time to come 1 day when lucky and, usually, much more than that. On the other hand, the documentation (at that time) is more complete than jBPM's. In my opinion, it's easier to fix documentation than to change how the community works and jBPM has been working on that in my last months as a treinee.

The best between Activiti and jBPM is a very personal choice. Both have the same crutial basics, so it comes down to very useful extras. Both have really good advantages between eachother and both have strong issues or disadvantages. In the end, it all comes down to personal preference on what is important to the work method used. In my opinion, **jBPM** is the **best** of those two due to the great respect I was treated with by the community and the representatives I could contact with and by the current and potential versatileness of the software they have in hand. Moreover Activiti seems to have some critical limitations regarding security.

To wrap up, the up side is that this was a great and different experience from everything I've had so far.

Bibliography

[1] http://php.net.

[2] http://jquery.com/.

[3] Activiti home page. http://activiti.org/.

[4] Alfresco. http://alfresco.com/.

[5] Alfresco and activiti. http://www.alfresco.com/news/press-releases/alfresco-launches-activiti-bpmn-20-business-process-

[6] Alfresco control metadata of a document type. http://blogs.alfresco.com/wp/developer/tag/metadata/.

[7] Alfresco extension bulk file importer. http://code.google.com/p/alfresco-bulk-filesystem-import/.

[8] Alfresco internal bulk file importer. http://docs.alfresco.com/4.0/index.jsp?topic=%2Fcom.alfresco.enterprise.doc%2Fconcepts%2FBulk-Import-Tool.html.

[9] Alfresco pricing. http://www.alfresco.com/products/compare/details.

[10] Alfresco with ldap. http://wiki.alfresco.com/wiki/The_Synchronization_Subsystem#Triggering_a_full_ldap_sync.

[11] Apache licence. http://opensource.org/licenses/Apache-2.0.

[12] Apache licence. http://en.wikipedia.org/wiki/Apache_License.

[13] Apache nutch. http://nutch.apache.org/.

[14] Apache tika. http://tika.apache.org/.

[15] Business process management. http://en.wikipedia.org/wiki/Business_process_management.

[16] Cc license. http://creativecommons.org/examples.

[17] Content management platform vendor nuxeo adds $3.8m in funding. http://www.nuxeo.com/media-center/content-management-platform-vendor-nuxeo-adds-3-8-million-in-fundin

[18] Documentum administrator user guide.

[19] Dom classlist. https://developer.mozilla.org/en-US/docs/Web/API/Element.classList.

[20] Dom classname. https://developer.mozilla.org/en-US/docs/Web/API/element.className.

[21] Elasticsearch. http://www.elasticsearch.org/.

[22] Elasticsearch alias searches. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-aliases.html.

[23] Elasticsearch community as active as solr. http://blog.sematext.com/2013/01/22/solr-vs-elasticsearch-userdev-communities/.

[24] Elasticsearch custom scoring. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/query-dsl-function-score-query.html#_script_score.

[25] Elasticsearch lexical broader search. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/analysis-snowball-tokenfilter.html.

[26] Elasticsearch lexical synonym search. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/analysis-synonym-tokenfilter.html.

[27] Elasticsearch manual. http://www.elasticsearch.org/webinars/getting-started-with-elasticsearch/.

[28] Elasticsearch quick start video. http://www.elasticsearch.org/webinars/getting-started-with-elasticsearch/.

[29] Elasticsearch search suggestions. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/search-suggesters-phrase.html.

[30] Elasticsearch synonym search. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/analysis-synonym-tokenfilter.html.

[31] Elasticsearch vs solr real time. http://stackoverflow.com/a/21252886/551625.

[32] Elasticsearch vs solr real time. http://www.elasticsearch.org/overview/elasticsearch/.

[33] Gpl 2.0 licence. http://opensource.org/licenses/GPL-2.0.

[34] Gpl 3.0 licence. http://opensource.org/licenses/GPL-3.0.

[35] Gpl, the most used open source licence. http://en.wikipedia.org/wiki/GPL_license.

[36] Introduction to bpm - business process management. http://www.youtube.com/watch?v=8gpqwtkWOFY.

[37] Inverted index. http://en.wikipedia.org/wiki/Inverted_index#Example.

[38] Java server faces website. https://javaserverfaces.java.net/.

[39] jboss classloading precedence. https://docs.jboss.org/author/display/AS7/Class+Loading+in+AS7#ClassLoadinginAS7-ClassLoading&Precedence.

[40] jbpm on sourceforge. http://sourceforge.net/projects/jbpm/.

[41] jbpm remote api. http://docs.jboss.org/jbpm/v6.0/userguide/jBPMRemoteAPI.html#d0e12348.

[42] jbpm service task. http://docs.jboss.org/jbpm/v6.0.1/userguide/jBPMBPMN2.html#d0e2796.

[43] jbpm service task. https://docs.jboss.org/jbpm/v6.0.1/userguide/jBPMBPMN2.html#d0e2796.

[44] jbpm taskservice gettasksassignedaspotentialowner(). http://docs.jboss.org/drools/release/6.0.1.Final/kie-api-javadoc/org/kie/api/task/TaskService.html#getTasksAssignedAsPotentialOwner%28java.lang.String,%20java.lang.String%29.

[45] jbpm user guide. http://docs.jboss.org/jbpm/v6.0.1/userguide/.

[46] jbpm user guide. http://jbpm.jboss.org.

[47] jbpm user guide section custom task. http://docs.jboss.org/jbpm/v6.0.1/userguide/jBPMDomainSpecificProcesses.html.

[48] jbpm user guide section "with variables". docs.jboss.org/jbpm/v6.0/userguide/jBPMRemoteAPI.html#d0e10882.

[49] Joining alfresco with elasticsearch zazi middleware. http://www.slideshare.net/zaiziltd/searching-alfresco-with-solr-cloud-4-elastic-search-and-amazon-clou

[50] Lgpl vs gpl. https://www.gnu.org/licenses/why-not-lgpl.html.

[51] Lucene. http://lucene.apache.org.

[52] Magic quadrant gartner ecm. http://www.rosebt.com/uploads/8/1/8/1/8181762/5729823_orig.pn.

[53] Mime type. http://en.wikipedia.org/wiki/MIME_type.

[54] Mit licence. http://opensource.org/licenses/MIT.

[55] Nuxeo. http://nuxeo.com/.

[56] Nuxeo bulk importer. http://doc.nuxeo.com/display/public/NXDOC/Nuxeo+Bulk+Document+Importer.

[57] Nuxeo control metadata of a document type. http://doc.nuxeo.com/display/public/NXDOC/Document+types.

[58] Nuxeo csv importer. http://doc.nuxeo.com/display/public/USERDOC/Nuxeo+CSV.

[59] Nuxeo live edit. http://doc.nuxeo.com/display/DMDOC55/Installing+Live+Edit.

[60] Nuxeo multiple output formats. http://docs.huihoo.com/nuxeo/5.1/nuxeo-reference-guide/transformation-service.html#d4e3520.

[61] Nuxeo studio. http://connect.nuxeo.com/.

[62] Nuxeo studio manual. http://doc.nuxeo.com/display/public/Studio/Nuxeo+Studio+Documentation+Center.

[63] Nuxeo using ldap. http://doc.nuxeo.com/display/public/ADMINDOC/Using+a+LDAP+Directory.

[64] Nuxeo using ldap. http://www.nuxeo.com/en/services/connect/pricing.

[65] Nuxeo with elasticsearch. https://github.com/tiry/nuxeo-elasticsearch.

[66] Open cmis. http://chemistry.apache.org/java/opencmis.html.

[67] phpbb, open source bulletin board. http://phpbb.com.

[68] Primefaces website. http://http://primefaces.org/.

[69] Salaboy's guide on how to jbpm 6 (used to get how to make custom tasks). http://salaboy.com/2013/10/22/kie-wb-jbpm-console-ng-configurations/.

[70] Search daemon. http://www.searchdaimon.com/.

[71] Search daemon is now free. http://www.searchdaimon.com/blog/searchdaimon_enterprise_search_is_now_free_and_open_source_on_github/.

[72] Solr. http://lucene.apache.org/solr/.

[73] Solr wikipedia article. http://wiki.apache.org/solr/.

[74] Tomcat home page. http://tomcat.apache.org/.

[75] trigger() jquery api documentation. http://api.jquery.com/trigger/.

[76] Use jre lib/ext classes in the application. https://community.jboss.org/thread/196781.

[77] What is ecm. http://www.aiim.org/What-is-ECM-Enterprise-Content-Management.

[78] Wikipedia link java server faces. http://en.wikipedia.org/wiki/JavaServer_Faces.

[79] Wikipedia link java server pages. http://en.wikipedia.org/wiki/JavaServer_Pages.

[80] Michael Anstis. Goodbye guvnor. hello drools workbench. http://planet.jboss.org/post/goodbye_guvnor_hello_drools_workbench.

[81] Gartner. Enterprise content management magic quadrant 2013. http://www.rosebt.com/blog/enterprise-content-management-mq-2013.

[82] Gartner. Enterprise content management magic quadrant report 2013. https://www.gartner.com/doc/2594722.

[83] Proof of concept that, using class loaders, my idea that placing the service task's code as .java in jbpm's git repository is not crazy.

[84] Bpmn2.0 specification.

[85] Microsoft. Sharepoint manual. http://technet.microsoft.com/en-us/sharepoint/hh126808.aspx.

[86] Estado português. Diário da república, 1.ª série — n.º 252 — 31 de dezembro de 2012. http://www.fc.ul.pt/sites/default/files/fcul/institucional/siadap/L_66_B_2012.pdf. Online.

[87] Primefaces. Primefaces showcase commandbutton. http://www.primefaces.org/showcase/ui/commandButton.jsf.

[88] Primefaces. Primefaces showcase datalist. http://www.primefaces.org/showcase/ui/data/dataList.xhtml.

[89] W3c. Document object model (dom) level 3 events specification. http://www.w3.org/TR/2014/WD-DOM-Level-3-Events-20140925/#trusted-events.

Appendix A

*.rules.json example files

This is an example .rules.json and .form.html file pair. To read in context, go to section 4.2.2.

A.1 userTask.rules.json

```
{
    "inputReplacements":{
        "replaceMe":"{{{replaceMe}}}",
        "fancyOutput":"{fancyOutput}",
        "ECMContentID": {
                "findInForm": "ECMOutput",
                "translatorClass": "full.class.name",
                "translatorMethod": "staticMethodName"
            }

    },

    "elements":{
        "nonExistingElement": {
            "mapsTo": "doesNotMatter",
            "path": "",
            "type": "file"
        },
        "risk":{
            "mapsTo": "risk_out",
            "type": "Integer"
        },
        "result": {
```

```
            "mapsTo": "riskAnalysisResult_out",
            "type": "String",
            "forceValue": {
                "Accept": "Accepted",
                "Reject": "Rejected",
            }
        }
    },

    "CMISBaseFolder" : "/wfcontents/registerAccount"

    "actions":{
        "before":[

        ],
        "After":[

        ]
    }
}
```

A.2 userTask.form.html

```
<p>
    <input type="text" name="example_in" />
</p>

<p>
    <input type="number" name="integerVal" />
</p>
<p>
    <input type="text" name="fromIntoWf_out" value="{{{replaceMe}}}" />
</p>
<p>
    {fancyOutput}
</p>
<p>
    ECMOutput
</p>
```

```
<p>
    <button type="submit" name="result" value="Accept">Accept</button>
    <button type="submit" name="result" value="Reject">Reject</button>
</p>
```

Appendix B

Send jQuery events to DOM

This is the source code of the jQuery extension sendjQueryEventsToDOM.js.

You may find the related content and explanation on section 5.2.1.

```
/*
 * resends events made
 */

(function(window, document, undefined){
    // custom event polyfill -> https://developer.mozilla
    .org/en/docs/Web/API/CustomEvent#Polyfill
    (function () {
        if(!window.CustomEvent &&
         document.createEvent){
            function CustomEvent ( event, params ) {
                params = params || { bubbles: false,
                 cancelable: false, detail: undefined };
                var evt = document.createEvent(
                 'CustomEvent' );
                evt.initCustomEvent( event,
                 params.bubbles, params.cancelable,
                  params.detail );
                return evt;
             };
            CustomEvent.prototype =
             window.Event.prototype;
            window.CustomEvent = CustomEvent;
        }
    })();
```

```
var $;

var eventOnDOM = {
    NATIVE_EVENTS: ["dblclick", "change", "focus",
     "blur"],
    BLACKLIST: ["blur", "focus", "click"] // keep
     these last in the same order
};
if(!document.createElement("input").click){
    // remove the click, focus and blue events from
     the blacklist as the browser does not have
      those
    eventOnDOM.BLACKLIST.pop();
    if(!document.createElement("input").focus)
    eventOnDOM.BLACKLIST.pop();
    if(!document.createElement("input").blur)
    eventOnDOM.BLACKLIST.pop();
}

// .click() exists in the DOM as a method. No need
 to add it

var execution = function (){
    var originalTrigger = $.event.trigger;

    var extraOnTrigger = function(event, data,
     elem, onlyHandlers){
        originalTrigger(event, data, elem,
         onlyHandlers);
        if(onlyHandlers){
            // by jQuery's internal spec, when this is
             used, it means that the event should not
              go to the DOM
            return;
        }
        var type = event && event.type || event;
        if($.eventOnDOM.BLACKLIST.indexOf(type)
         !== -1){
            return;
```

```
            }
            if($.eventOnDOM.NATIVE_EVENTS
            .indexOf(type) !== -1){
                var evt =
                 document.createEvent("HTMLEvents");
                evt.initEvent(type, true, true ); //
                 eventType,bubbling,cancelable
            }else{
                var event = new CustomEvent(event, {
                        bubbles: true,
                        cancelable: true,
                        detail: data
                    });
            }
            if(evt){
                // prevent jQ from calling callback twice
                 (taken from jQuery's source code)
                $.event.triggered = type;
                elem.dispatchEvent(evt);
                $.event.triggered = undefined;
            }

        };
        $.event.trigger = extraOnTrigger;

    };

    // wait for jQuery
    var waiting = function(){
        if(window.jQuery.fn){
            $ = window.jQuery;
            window.jQuery.eventOnDOM =
             eventOnDOM;
            execution();
        }else{
            setTimeout(waiting, 200);
        }
    };
    waiting();
```

```
}(window, document));
```