

# PFM – Experto en Big Data

CREACIÓN DE EXÁMENES CON BIG DATA

Tamara Cortés Estévez

Agradecimientos:

Oscar Gonzalez por la idea del proyecto.

Angel Rey (tutor) por el apoyo prestado.

## Contenido

Introducción al proyecto .....	1
Análisis .....	2
Planificación .....	8
Desarrollo.....	9
Arquitectura .....	9
Cluster .....	9
Nodos.....	9
SPARK .....	9
Hadoop YARN .....	11
Elasticsearch .....	15
Spring-data-elasticsearch.....	16
Kibana.....	17
D3 .....	17
Implementación de la solución.....	18
Parte Batch .....	18
Parte Web .....	31
Implantación en el cluster.....	41
Manual de usuario .....	54
Creación de exámenes .....	55
Visualizacion .....	58
Número de preguntas por tags .....	58
REFERENCIAS .....	60
web .....	60
Libros.....	60

## Introducción al proyecto

Este proyecto nace de la necesidad en las empresas de tecnología de disponer de una persona especializada en la parte técnica para realizar entrevistas a los candidatos a un puesto técnico.

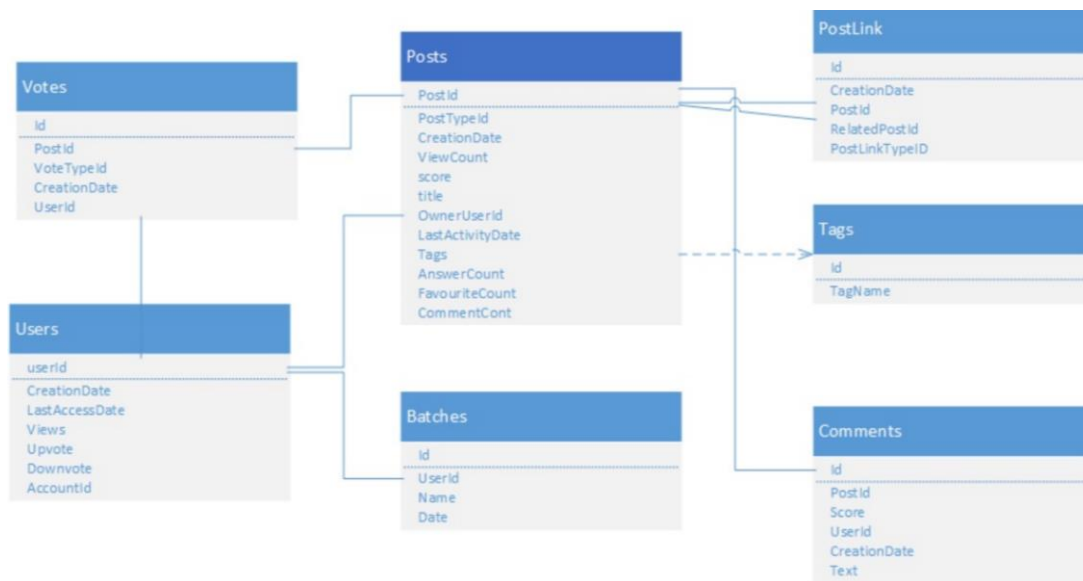
Realizar exámenes técnicos requiere trabajo y se quedan obsoletos muy rápido, por lo que se busca una forma de poder automatizar este proceso.

Existen web enfocadas a dudas tecnológicas que tienen distintos usuarios y que son respondidas con mayor o menor acierto por otros, llegando en la mayoría de los casos a una solución a la problemática planteada. Varias de estas webs ofrecen sus datos al público. Stackoverflow es una de las webs actuales más conocidas para realizar preguntas sobre programación que recoge gran cantidad de datos por lo que se plantea a partir de su dataset con las dudas y sus posibles respuestas la creación de los exámenes.

## Análisis

Cada web perteneciente a Stackoverflow contiene archivos XML separados comprimidos en 7-zip usando compresión bzip2. Este contenido está bajo licencia cc-by-sa 3.0 licensed.

Cada archivo de cada web contiene Posts, Users, Votes, Comments, PostHistory and PostLinks. La relación entre ellos es:



El objetivo de este proyecto es la creación de exámenes por lo que solo se ha necesitado el fichero de Posts xml.

Este fichero tiene los siguientes campos para preguntas y respuestas:

- Id
- PostTypeId
  - 1: Question
  - 2: Answer
  - 3: Orphaned tag wiki
  - 4: Tag wiki excerpt
  - 5: Tag wiki
  - 6: Moderator nomination
  - 7: "Wiki placeholder" (seems to only be the election description)
  - 8: Privilege wiki
- ParentID (only present if PostTypeId is 2)
- AcceptedAnswerId (only present if PostTypeId is 1)
- CreationDate
- Score
- ViewCount

- Body
- OwnerUserId (only present if user has not been deleted)
- LastEditorUserId
- LastEditorDisplayName
- LastEditDate
- LastActivityDate
- CommunityOwnedDate (present only if post is community wikied)
- ClosedDate (present only if the post is closed)
- Title
- Tags
- AnswerCount
- CommentCount
- FavoriteCount

Cada línea del fichero xml representa un post. De los tipos de post posibles solo se necesitan los de preguntas (PostTypeId =1,) y respuestas (PostTypeId=2). Otros atributos importantes son:

- ParentId: Este atributo sólo tiene valor en las respuestas (en las preguntas parentId=0) y representa el id de la pregunta con la que se corresponde.
- AcceptedAnswerId: Sólo existe en las preguntas ya que representa el id de la respuesta que se ha dado como correcta.

De las preguntas existentes solo serán útiles los que tengan respuesta correcta, es decir valor en AcceptedAnswerId.

Ejemplo de pregunta y respuesta obtenido directamente del fichero xml:

Pregunta extraída del fichero:

```
<row Id="311863" PostTypeId="1" AcceptedAnswerId="311868" CreationDate="2016-03-05T16:47:26.767" Score="1" ViewCount="72" Body="<p>The following for example:
</p>&#xA;&#xA;<pre>i
readString(&amp;packet-&amp;gt;data.play_server.updatesign.line1, pbuf,
ps);&#xA;</code></pre>&#xA;&#xA;<p>It has a large amount of
nested structs/unions. Is this generally frowned upon in code cleanliness, or
bad for performance?</p>&#xA;" OwnerUserId="184649"
LastActivityDate="2016-03-05T17:26:57.967" Title="In C, are large 'pointer
chains' bad for performance or code cleanliness?"
Tags="<c++><c><coding-standards>" AnswerCount="1"
CommentCount="5" />
```

Pregunta formateada:

```
Id="311863"
PostTypeId="1"
AcceptedAnswerId="311868"
CreationDate="2016-03-05T16:47:26.767"
Score="1"
ViewCount="72"
Body="<p>The following for example:
</p><pre>i = readString(&packet-
&data.play_server.updatesign.line1,
ps);</pre>It has a large amount of
nested structs/unions. Is this generally frowned upon in code cleanliness, or
bad for performance?</p>"
OwnerUserId="184649"
LastActivityDate="2016-03-05T17:26:57.967"
Title="In C, are large 'pointer chains' bad for performance or code
cleanliness?"
Tags="<code>c++</code><code>c</code><code>coding-standards</code>"
AnswerCount="1"
CommentCount="5"
```

Respuesta extraída del fichero:

```
<row Id="311868" PostTypeId="2" ParentId="311863" CreationDate="2016-03-
05T17:26:57.967" Score="2" Body="<p>Bad for performance is questionable
in your example. True runtime pointer dereferencing in sequence can be bad for
cache coherency. But your example doesn't show this; it only has one runtime
pointer to dereference.</p><pre>C is a statically typed
language. Therefore, at compile time, the compiler knows the sizes and layout
of every member subobject of every object. So
</pre>.play_server.updatesign.line1</code> basically compiles down
to a static integer offset of a pointer. It becomes
<pre>&packet-&data + static_offset</pre> (using
byte addition rather than C pointer
arithmetic).</pre>If those <pre>s
had been <pre>-&gt;</pre> instead, that would be a different
matter. Each such pointer could at runtime be any particular value. And
therefore, the compiler can't boil everything down to a static offset; it has
to be executed at runtime as a chain of accesses, each one reading a pointer
from the next object. And thus, like iterating through a linked list, it can
be bad for cache coherency and so forth.</p><pre>As for
code cleanliness, that all rather depends on where you are and what you're
doing. If <pre>packet-&data</pre> is just a plain data
structure with no need for invariants, then it's fine. Those subobjects are
just aggregates, not smart containers that need encapsulation. If there is some
invariant in play within this data structure, then direct modification of such
a data structure would violate encapsulation and thus be
dangerous.</pre>Then again, providing encapsulation in
C usually requires hiding declarations (forward declarations and so forth),
which tends to make inlining difficult. And if each of those sub-object accesses
were an encapsulated action, that could represent the kind of 'pointer
```

```
chain" that you are concerned about, since the compiler would not easily
be able to compile accesses down to a static offset. So C programs tend to
employ encapsulation at the boundaries of major systems or APIs, rather than
within individual systems themselves.</p>&#xA;&#xA;</p><\/p><\/p>What is
clean to one programmer isn't clean to another.</p>&#xA;<\/p>
OwnerUserId="28374" LastActivityDate="2016-03-05T17:26:57.967"
CommentCount="0" />
```

Respuesta formateada:

```
Id="311868"
PostTypeId="2"
ParentId="311863"
CreationDate="2016-03-05T17:26:57.967"
Score="2"
Body="<p>Bad for performance is questionable in your example. True
runtime pointer dereferencing in sequence can be bad for cache coherency. But
your example doesn't show this; it only has one runtime pointer to
dereference.</p>&#xA;&#xA;</p><\/p>C is a statically typed language.
Therefore, at compile time, the compiler knows the sizes and layout of every
member subobject of every object. So
<code>.play_server.updatesign.line1</code> basically compiles down
to a static integer offset of a pointer. It becomes
<code>&amp;packet-&amp;gt;data + static_offset</code> (using
byte addition rather than C pointer
arithmetic).</p>&#xA;&#xA;</p><\/p>If those <code>.</code>s
had been <code>-&amp;gt;</code>s instead, that would be a different
matter. Each such pointer could at runtime be any particular value. And
therefore, the compiler can't boil everything down to a static offset; it has
to be executed at runtime as a chain of accesses, each one reading a pointer
from the next object. And thus, like iterating through a linked list, it can
be bad for cache coherency and so forth.</p>&#xA;&#xA;</p><\/p>As for
code cleanliness, that all rather depends on where you are and what you're
doing. If <code>packet-&amp;gt;data</code> is just a plain data
structure with no need for invariants, then it's fine. Those subobjects are
just aggregates, not smart containers that need encapsulation. If there is some
invariant in play within this data structure, then direct modification of such
a data structure would violate encapsulation and thus be
dangerous.</p>&#xA;&#xA;</p><\/p>Then again, providing encapsulation in
C usually requires hiding declarations (forward declarations and so forth),
which tends to make inlining difficult. And if each of those sub-object accesses
were an encapsulated action, that could represent the kind of "pointer
chain" that you are concerned about, since the compiler would not easily
be able to compile accesses down to a static offset. So C programs tend to
employ encapsulation at the boundaries of major systems or APIs, rather than
within individual systems themselves.</p>&#xA;&#xA;</p><\/p>What is
clean to one programmer isn't clean to another.</p>&#xA;<\/p>
OwnerUserId="28374"
LastActivityDate="2016-03-05T17:26:57.967"
CommentCount="0"
```



Para ver su correspondencia con la web de stackoverflow, se toma el id:

<http://programmers.stackexchange.com/questions/311863>

Hay que tener en cuenta que el fichero del que se parte está actualizado hasta determinada fecha por lo que los datos de la web pueden variar un poco, pero se considera depreciable dicha variación para el objetivo del proyecto.

The screenshot shows a StackExchange page for the question "In C, are large 'pointer chains' bad for performance or code cleanliness?". The page includes a header with navigation links, a "Join" button, and a brief explanation of how the site works. The question itself is about the performance and cleanliness of nested structs/unions in C, with a code example provided. The question has 100 views and was asked 3 months ago. There is one answer, which is marked as the best answer. The answer discusses the performance implications of pointer dereferencing and the cleanliness of the code. It also includes a section for "Hot Network Questions" on the right side of the page.

StackExchange join this community tour help Search Q&A

PROGRAMPERS\*

Questions Tags Users Badges Unanswered Ask Question

Programmers Stack Exchange is a question and answer site for professional programmers interested in conceptual questions about software development. It's 100% free.

Join

Here's how it works:

Anybody can ask a question

Anybody can answer

The best answers are voted up and rise to the top

In C, are large 'pointer chains' bad for performance or code cleanliness?

asked 3 months ago  
viewed 100 times  
active 3 months ago

The following for example:

```
1 i = readString(&packet->data.play_server.updatesign.line1, pbuf, ps);
```

It has a large amount of nested structs/unions. Is this generally frowned upon in code cleanliness, or bad for performance?

share improve this question

asked Mar 5 at 16:47  
JavaProphet  
130 • 3

2 There is only one pointer dereference -> in the example you're showing. The . operator has no performance cost in C/C++, as the items are embedded together. - Erik Edit Mar 5 at 16:58

3 In the OO world it's usually a sign that you have poor encapsulation and are violating the Single Responsibility Principle. I have no idea about the C world. - MetaFight Mar 5 at 16:58

4 Search for the "Law of Demeter". It can also be applied to C. - 5gon12eder Mar 5 at 17:13

That's why I said 'pointer chains', I'm talking about just long lines of referencing. - JavaProphet Mar 5 at 17:25

1 @JavaProphet Your example did not include "long lines of referencing". Or dereferencing. That's why people are confused. - Nicol Bolas Mar 5 at 17:28

add a comment

1 Answer

active oldest votes

3

Bad for performance is questionable in your example. True runtime pointer dereferencing in sequence can be bad for cache coherency. But your example doesn't show this; it only has one runtime pointer to dereference.

C is a statically typed language. Therefore, at compile time, the compiler knows the sizes and layout of every member subobject of every object. So .play\_server.updatesign.line1 basically compiles down to a static integer offset of a pointer. It becomes &packet->data + static\_offset (using byte addition rather than C pointer arithmetic).

If those .s had been ->s instead, that would be a different matter. Each such pointer could at runtime be any particular value. And therefore, the compiler can't boil everything down to a static offset; it has to be executed at runtime as a chain of accesses, each one reading a pointer from the next object. And thus, like iterating through a linked list, it can be bad for cache coherency and so forth.

As for code cleanliness, that all rather depends on where you are and what you're doing. If packet->data is just a plain data structure with no need for invariants, then it's fine. Those subobjects are just aggregates, not smart containers that need encapsulation. If there is some invariant in play within this data structure, then direct modification of such a data structure would violate encapsulation and thus be dangerous.

Then again, providing encapsulation in C usually requires hiding declarations (forward declarations and so forth), which tends to make inlining difficult. And if each of those sub-object accesses were an encapsulated action, that could represent the kind of "pointer chain" that you are concerned about, since the compiler would not easily be able to compile accesses down to a static offset. So C programs tend to employ encapsulation at the boundaries of

Related

1 Help me construct a list of best approaches for new C and C++ developers

91 Why are pointers not recommended when coding with C++?

16 Enhancing the level of my C/C++ code

48 Strictness in programming methods among Stack Overflow users

5 What's so bad about pointers in C++?

8 Should my team use some common well-regarded coding standard as a basis for its own?

2 Reference vs dereference pointers in arguments C++/C

2 Drawback of mixing C++ code with C for performance purposes

3 No exceptions C++ and partially constructed objects

5 non-optional pointers vs. non-const references in C++

Hot Network Questions

Quick!! You got only 30 seconds

How to keep images from taking over the world?

Battles under the influence (of alcohol)

Travelling with infant to Hong Kong - Need to use Kitchen

What is the longest rhymonym?

How to split text without spaces into list of words?

También se desea poder filtrar por grado de dificultad de las preguntas, para esto se ha aplicado la siguiente lógica.

$$\text{grado de dificultad} = \frac{(\text{fecha creación pregunta} - \text{fecha creación respuesta acertada})}{\text{número de visitas hasta la fecha de la respuesta acertada} *}$$

El número de visitas hasta la fecha de la respuesta acertada no es exacto, puesto que no se dispone de ese dato en el dataset, por lo que se realiza el siguiente algoritmo:

$$x \text{ días} = \text{fecha de creación de la pregunta} - \text{fecha de creación de la respuesta acertada}$$

$$y \text{ días} = \text{fecha de creación de la pregunta} - \text{fecha del fichero Posts.xml}$$

$$x \text{ días} \rightarrow \mu \qquad \sigma = \frac{y * \mu}{x} \qquad \sigma \text{ número de veces visto ponderado}$$

$$y \text{ días} \rightarrow \sigma$$

Se ha realizado un análisis inicial sobre el dataset y se han obtenido los siguientes datos:

Numero de posts: 166742

Numero Preguntas Sin respuestas: 1746

Numero Preguntas Sin respuestas correctas: 15355

Por lo que tenemos un 10% de preguntas que serán descartadas puesto que no tienen una respuesta como válida o simplemente no tienen respuestas.

## Planificación

Se ha establecido un calendario para la planificación del proyecto:

	Task ID	Nombre de tarea	Duration	Start	Finish	Pred	Resource Names
1	Auto	<b>Inicio de proyecto</b>	37 days	Wed 6/1/16	Thu 7/21/16		
2	Man	Inicio de proyecto	0 days	Wed 6/1/16	Wed 6/1/16		
3	Auto	<b>Análisis</b>	6 days	Wed 6/1/16	Wed 6/8/16		
4	Auto	Análisis arquitectura	2 days	Wed 6/1/16	Thu 6/2/16	2	Analista Programador
5	Auto	Análisis datos y proceso	2 days	Fri 6/3/16	Mon 6/6/16	4	Analista Programador
6	Auto	Diseño técnico	2 days	Tue 6/7/16	Wed 6/8/16	5	Analista Programador
7	Auto	<b>Desarrollo</b>	23 days	Thu 6/9/16	Mon 7/11/16		
8	Auto	Procesado de datos con spark (tratamiento y conversión a json)	8 days	Thu 6/9/16	Mon 6/20/16	6	Analista Programador
9	Auto	Almacenamiento en ElasticSearch	4 days	Tue 6/21/16	Fri 6/24/16	8	Analista Programador
10	Auto	Kibana: Explotación de datos	4 days	Mon 6/27/16	Thu 6/30/16	9	Analista Programador
11	Auto	Servicio Restful de obtención de exámenes con los diferentes	3 days	Fri 7/1/16	Tue 7/5/16	10	Analista Programador
12	Auto	Interfaz web	4 days	Wed 7/6/16	Mon 7/11/16	11	Analista Programador
13	Auto	<b>Implantación</b>	5 days	Tue 7/12/16	Mon 7/18/16		
14	Auto	Instalación base	2 days	Tue 7/12/16	Wed 7/13/16	12	Analista Programador
15	Auto	Pruebas funcionales	2 days	Thu 7/14/16	Fri 7/15/16	14	Analista Programador
16	Auto	Ejecución y grabación video	1 day	Mon 7/18/16	Mon 7/18/16	15	Analista Programador
17	Auto	<b>Documentación</b>	3 days	Tue 7/19/16	Thu 7/21/16		
18	Auto	Memoria del proyecto	3 days	Tue 7/19/16	Thu 7/21/16	16	Analista Programador

## Desarrollo

### *Arquitectura*

Componentes de arquitectura sobre los que se sostiene el proyecto y que son necesarios comprender.

### *Cluster*

Conjunto de uno o más nodos.

### *Nodos*

Un nodo es un servidor que forma parte de un cluster. Puede haber tantos nodos como se quieran por cada Cluster. En esta estructura los nodos trabajan conjuntamente, pero la programación es compleja por lo que surgen nuevos modelos que forman parte de la arquitectura big data que permiten una programación sencilla y de manera distribuida.

### *SPARK*

Spark es una plataforma de código abierto para el análisis y procesamiento distribuido de grandes cantidades de datos. Fue creado en la Universidad de Berkeley en California en el AMPLab en 2009.

Su origen tiene como objetivo proporcionar una mejor alternativa para determinados casos de uso de procesamiento de datos en los que el modelo MapReduce no era del todo eficiente.

MapReduce ofrece un modelo relativamente simple para escribir programas que trabajan con grandes conjuntos de datos y que se pueden ejecutar paralelamente en cientos y miles de máquinas al mismo tiempo. Debido a su arquitectura, MapReduce tiene prácticamente una relación lineal de escalabilidad, si los datos crecen es posible añadir más máquinas y tardar prácticamente lo mismo.

Spark mantiene la escalabilidad lineal y la tolerancia a fallos de MapReduce, pero mejora las características gracias a varias funcionalidades: DAG y RDD.

**DAG** (grafos acíclicos dirigidos). Es un modelo para planificar el trabajo, define el flujo de ejecución, en el cual los objetos creados en cada paso en el job se representan como nodos en el grafo donde las operaciones a llevar a cabo en los datos se definen en los vertices y el orden de ejecución se especifica por la dirección de los vértices en el grafo. Acíclico significa que no hay bucles en el grafo. Por lo tanto los nodos son los RDD y las flechas son las transformaciones

Cada tarea de Spark crea un DAG de etapas de trabajo para que se ejecuten en un determinado cluster. El paradigma MapReduce ofrece poca flexibilidad para crear flujos de datos ya que usa el esquema "Map -> Shuffle -> Reduce" (2 estados map y reduce). Si se desea usar otro esquema obliga a tener fases identidad que consumen recursos. Los grafos DAG creados por Spark pueden tener cualquier número de etapas. Además Spark con DAG es más rápido que MapReduce por el hecho de que no tiene que escribir en disco los resultados obtenidos en las etapas intermedias del grafo.

**RDDs ("Resilient Distributed Datasets")**: Modelo de datos utilizado por Apache Spark.

Es una colección distribuida. Esto quiere decir que está particionada entre los distintos workers de Spark.

- Son inmutables: Una vez se crea un RDD, éste no cambiará, sino que cada transformación que apliquemos sobre él generará un nuevo RDD.
- Su evaluación es perezosa. Con los RDD's estamos definiendo un flujo de información, pero no se ejecuta en el momento de definición, sino en el momento en el que se evalúe aplicando una acción sobre el RDD. La creación de un RDD es determinista (aplicada sobre los mismos datos de partida, siempre obtendrá el mismo resultado), en el RDD se van guardando la

secuencia de transformaciones para aplicarla sobre los datos cuando sea necesario.

Los RDDs soportan dos tipos de operaciones:

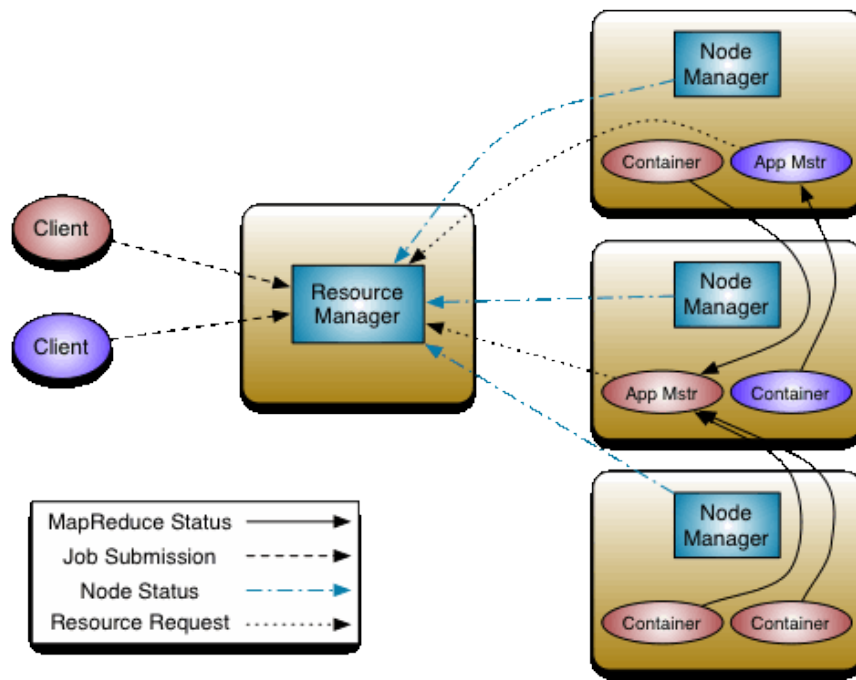
- Transformaciones: Operaciones perezosas que retornan otro RDD.
- Acciones: operaciones que retornan valores.

Con el fin de mejorar el rendimiento entre operaciones, se permite la persistencia o el almacenamiento en caché de un RDD entre operaciones.

Spark permite realizar trabajos paralelizados totalmente en memoria, lo cual reduce mucho los tiempos de procesamiento

### *Hadoop YARN*

YARN es la estructura que soporta Hadoop a partir de Hadoop 2.0 (MRv2). Las versiones anteriores de Hadoop estaban escritas para implementar MapReduce, YARN generaliza la arquitectura original para dar soporte a aplicaciones no sólo MapReduce. Es por lo tanto una reescritura de software que desacopla las capacidades de gestión de recursos y planificación de MapReduce del componente de procesamiento de datos, permitiendo a Hadoop soportar enfoques más variados de procesamiento, y una gama más amplia de aplicaciones.



Los elementos que intervienen son:

- **Resource Manager** tiene dos componentes principales: Scheduler and ApplicationsManager
  - El Scheduler se encarga de conocer la disponibilidad los recursos para las distintas aplicaciones agrupándolas según sus restricciones de capacidad, colas, etc... pero no se ocupa de monitorizar estas aplicaciones. Es el encargado de particionar los recursos del clúster entre las distintas aplicaciones.
  - Los ApplicationMaster son los responsables de solicitar al Scheduler los recursos necesarios para las aplicaciones y éstos sí que se encargan de monitorizar el estado de las mismas y su progreso. Son gestionados por el ApplicationManager.
- **Node Manager** hay uno por nodo esclavo, es el responsable de la monitorización y gestión de los recursos. Recoge las directrices del ResourceManager y crea contenedores basados en los requerimientos de la tarea.

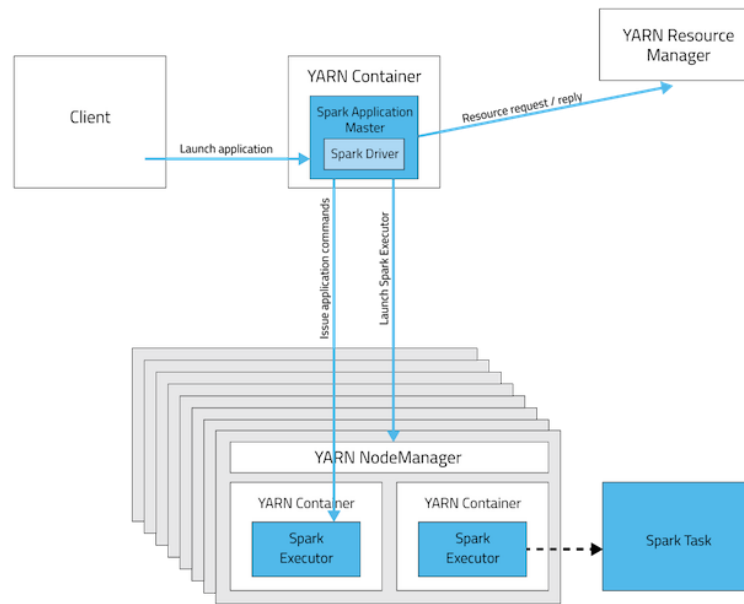
- **Application master** se despliega junto al NodeManager. Es creado por Job y controla la monitorización y la ejecución de las tareas usando el contenedor. Negocio los requerimientos de los recursos para el Job con el ResourceManager y tiene la responsabilidad de completar las tareas. Proporciona la tolerancia a fallos a nivel de tarea.
- **Container** es la unidad básica de la asignación en lugar de un Map o Reduce slot en Hadoop 1.x. El contenedor se define con atributos como la memoria, CPU, disco etc.
- **History Server** mantiene la historia de todos los Jobs. Es un demonio opcional.

#### Spark in hadoop yarn

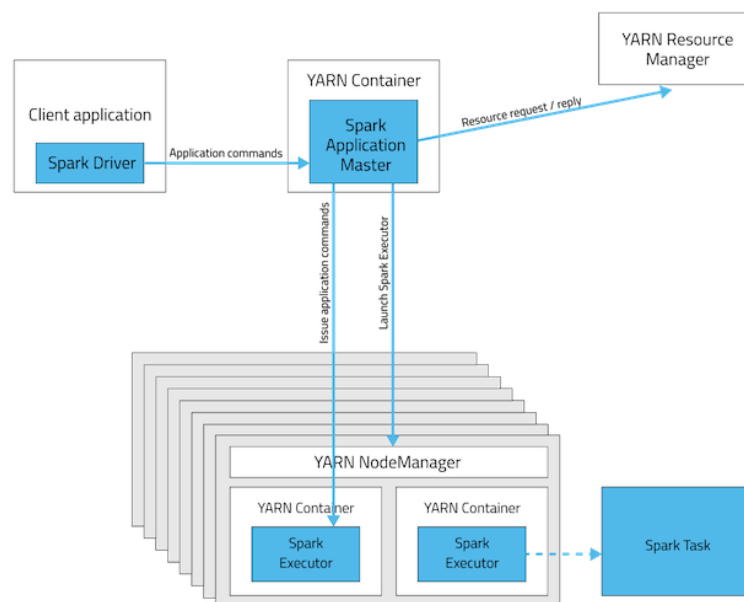
Cuando las aplicaciones Spark corren en YARN, la planificación, manejo de recursos y seguridad son controlados por YARN. Se pueden correr aplicaciones en modo cluster o en modo cliente.

En modo cluster, el driver corre en el Application master en un host del cluster elegido por YARN. Esto significa que el mismo proceso el cual corre en un contenedor de YARN es responsable de conducir la aplicación y pedir recursos a YARN. El cliente que lanza el proceso no necesita continuar hasta que finalice la aplicación.





En modo cluster, el driver corre en la misma máquina donde se le envía el trabajo a spark. El applicationMaster solo se encarga de pedir container para los executor a YARN. El cliente se comunica con los contenedores preparados para la ejecución una vez comience.



## HDFS

Sistema de almacenamiento de hadoop, es un sistema de ficheros distribuido.

Los elementos que componen HDFS son:

- **NameNode:** Sólo hay uno en el cluster. Regula el acceso a los ficheros por parte de los clientes. Mantiene en memoria la metadata del sistema de ficheros y control de los bloques de fichero que tiene cada DataNode.
- **DataNode:** Son los responsables de leer y escribir las peticiones de los clientes. Los ficheros están formados por bloques, estos se encuentran replicados en diferentes nodos.

### *Elasticsearch*

Elasticsearch es un servidor de búsqueda distribuido de documentos en formato JSON. Está basado en Lucene, ofrece alta disponibilidad soportando grandes cantidades de datos y cortos tiempos de respuesta. Provee de una interfaz web RESTful. Está desarrollado en Java y está publicado como código abierto bajo las condiciones de la licencia Apache.

Elasticsearch no es una base de datos relacional, pero si se puede realizar una cierta analogía de conceptos:

BD elasticsearch	BD relacional
índice	esquema
tipo	tabla
documento	fila
propiedad	columna

Elasticsearch no solo almacena los documentos, sino que los indexa de manera que sea posible realizar búsquedas en ellos.

Cada instancia de elasticsearch está en un nodo. Varios nodos están agrupados en un cluster. Los documentos de elasticsearch se organizan índices (colección de

documentos que tienen características similares, los índices están identificados por un nombre). Los índices internamente se dividen en shards los cuales pueden residir en diferentes nodos. Se pueden tener nodos primarios o nodos réplica.

Cuando la información que se indexa sobrepasa el límite de una sola máquina, elasticsearch ofrece distintas maneras de saltarnos esa limitación.

El sharding permite dividir estos índices en distintas “piezas” ofreciendo la posibilidad de escalar horizontalmente, además de paralelizar y distribuir las distintas operaciones que se hagan sobre esos índices.

La replicación ofrece un mecanismo para que en caso de fallo el usuario no se vea afectado.

Para interactuar con Elasticsearch existen dos protocolos:

HTTP: Un RESTfull API, por defecto escucha en el puerto 9200 de la máquina donde se está ejecutando Elasticsearch, la información que necesita Elasticsearch viaja en el cuerpo de las peticiones codificado en formato JSON. La mayoría de las respuestas que da el servidor de Elasticsearch son también en este formato.

Protocolo binario nativo elasticsearch: Es un protocolo propio que desarrollo elasticsearch para la comunicación entre nodos. Hay multitud de librerías que encapsulan este acceso al servidor.

### *Spring-data-elasticsearch*

Spring Data es el nombre de un módulo de Spring que a su vez engloba un gran número de sub-módulos cuyo objetivo es facilitar el acceso y explotación de datos en aplicaciones basadas en Spring. Spring-data-elasticsearch es uno de los submódulos.

Spring Data Elasticsearch es una capa de abstracción más, montada sobre elasticsearch, para evitar al máximo posible el código repetitivo necesario en

cualquier aplicación que use la especificación elasticsearch para el acceso a base de datos.

Spring-data-elasticsearch provee de una plantilla de un alto nivel de abstracción para realizar “almacenamiento, búsquedas y ordenación de documentos. Se comunica con elasticsearch con el api nativa de este (transport client, Node client).

### *Kibana*

Herramienta open source (licencia Apache) que permite hacer análisis y búsquedas sobre elasticsearch. Provee una consola interactiva para realizar consultas directamente a elasticsearch desde el navegador.

### *D3*

Librería de javascript que permite producir infogramas dinámicos e interactivos usando HTML,SVG y CSS. D3 utiliza los estándares web para darle al usuario todas las capacidades que tienen los buscadores modernos sin tener que utilizar framework propietario, y combina componentes de visualización y manipulación del DOM.

## *Implementación de la solución*

El proyecto se ha estructurado en tres partes:

- La parte batch, es la que se encarga de procesar los datos
- La parte web, la encargada de la creación de exámenes
- La parte de visualización de datos kibana

### *Parte Batch*

Lo principal es saber que queremos guardar para poder consultar posteriormente.

Para guardar la información se ha utilizado elasticsearch (definido anteriormente).

Se ha definido el índice con el nombre "posts" y el tipo con el nombre "row".

En elasticsearch no es necesario definir un esquema para los documentos a almacenar, al indexarlos elasticsearch se encarga de interpretar los tipos de los datos, pero para que elasticsearch sea capaz de tratar los campos fechas como fechas, y números como números, necesita que se especifiquen los tipos que contienen cada campo.

En mapeo de datos se ha indicado el tipo, formato (donde fuera necesario) y el index.

El atributo index controla como la cadena va a ser indexada, se han utilizado los siguientes valores:

- not\_analyzed: Indexa el campo para que se pueda buscar pero lo indexa con el valor exacto como fue especificado, no lo analiza.
- No: No indexa este campo, no se podrán realizar búsquedas sobre este valor. (Esto se ha utilizado para el campo body puesto que no era necesario para el objetivo del proyecto realizar búsquedas en este campo).

Si accedemos a la url donde está el cluster de elasticsearch podremos verlo:

[http://host:9200/posts/\\_mapping/row?pretty](http://host:9200/posts/_mapping/row?pretty)

```

{
  "posts" : {
    "mappings" : {
      "row" : {
        "_id" : {
          "path" : "id"
        },
        "properties" : {
          "acceptedAnswerId" : {
            "type" : "long"
          },
          "acceptedAnswerIdString" : {
            "type" : "string"
          },
          "answerCount" : {
            "type" : "long"
          },
          "body" : {
            "type" : "string",
            "index" : "no"
          },
          "commentCount" : {
            "type" : "long"
          },
          "creationDate" : {
            "type" : "date",
            "format" : "yyy-MM-dd HH:mm:ss.SSS"
          },
          "creationDateString" : {
            "type" : "string"
          },
          "favoriteCount" : {
            "type" : "long"
          },
          "gradoDificultad1" : {
            "type" : "double"
          },
          "id" : {
            "type" : "long"
          },
          "idString" : {
            "type" : "string"
          },
          "lastActivityDate" : {
            "type" : "date",
            "format" : "yyy-MM-dd HH:mm:ss.SSS"
          },
          "lastActivityDateString" : {
            "type" : "string"
          },
          "ownerUserId" : {
            "type" : "long"
          },
          "parentId" : {
            "type" : "long"
          },
          "parentIdString" : {
            "type" : "string"
          },
        }
      }
    }
  }
}

```

```
    "postTypeId" : {
      "type" : "long"
    },
    "score" : {
      "type" : "long"
    },
    "tags" : {
      "type" : "string",
      "index" : "not_analyzed",
      "store" : true
    },
    "viewCount" : {
      "type" : "long"
    }
  }
}
}
```

Para obtener estos datos, partimos de un fichero xml que se ha subido a HDFS.

Este fichero tiene la siguiente estructura ya vista en la introducción:

```
<posts>  
  <row Id="1" PostTypeId="1" AcceptedAnswerId="13" CreationDate="2010-09-01T19:34:48.000" Score="100" ViewCount="25786" Body="<!--p&gt;A coworker of mine believes that <!--em&gt;any</em&gt;<br /-->  
  <row Id="3" PostTypeId="2" ParentId="1" CreationDate="2010-09-01T19:36:50.053" Score="29" Body="<!--p&gt;Ideally, code should be so well coded that it should be auto explicative. In the re  
  <row Id="4" PostTypeId="1" AcceptedAnswerId="26" CreationDate="2010-09-01T19:37:39.957" Score="66" ViewCount="5367" Body="<!--p&gt;When starting a project for a company that's not primari  
  <row Id="7" PostTypeId="2" ParentId="1" CreationDate="2010-09-01T19:42:16.797" Score="10" Body="<!--p&gt;I think the answer is the usual <!--qot;It depends&quot; one. Commenting code just t  
  <row Id="9" PostTypeId="1" CreationDate="2010-09-01T19:43:04.957" Score="39" ViewCount="5775" Body="<!--p&gt;Sometimes, the things I have to do for my job are interesting and engaging. Sc  
  <row Id="12" PostTypeId="2" ParentId="4" CreationDate="2010-09-01T19:44:47.413" Score="13" Body="<!--p&gt;IMO, i've found that the transparency offered by agile processes (e.g. Scrum, SysC  
  <row Id="13" PostTypeId="2" ParentId="1" CreationDate="2010-09-01T19:45:33.183" Score="168" Body="<!--p&gt;Only if the comment describes what the code is doing.<!--p&gt;#fixx;#fixx;#fixx;#fixx;p&gt;  
  <row Id="16" PostTypeId="1" AcceptedAnswerId="3675" CreationDate="2010-09-01T19:46:45.303" Score="32" ViewCount="4270" Body="<!--p&gt;I have read a few articles on Internet about programmi
```

El tratamiento del fichero se basa en el proceso:

1. No procesar las líneas que no tengan "row" (así evitamos tener que tratar el fichero para eliminar la primera línea y cualquier otro error que este pudiera tener).
2. Quedarse solo con los posts de tipo 1 y 2 (como comentamos en la introducción son los de tipo pregunta y respuesta).
3. Obtenemos las fechas de las respuestas correctas.
4. Eliminamos las preguntas que no tengan respuestas correctas.
5. Asignamos a esas preguntas el grado de dificultad basándonos en el criterio comentado anteriormente.
6. Insertamos en BBDD.

## **Estructura del proyecto batch**

Se ha creado un proyecto maven para el procesamiento de los datos. Este proyecto maven está realizado con java y hace uso del framework de spark.

Para el control de versiones se está utilizando git con bitbucket:

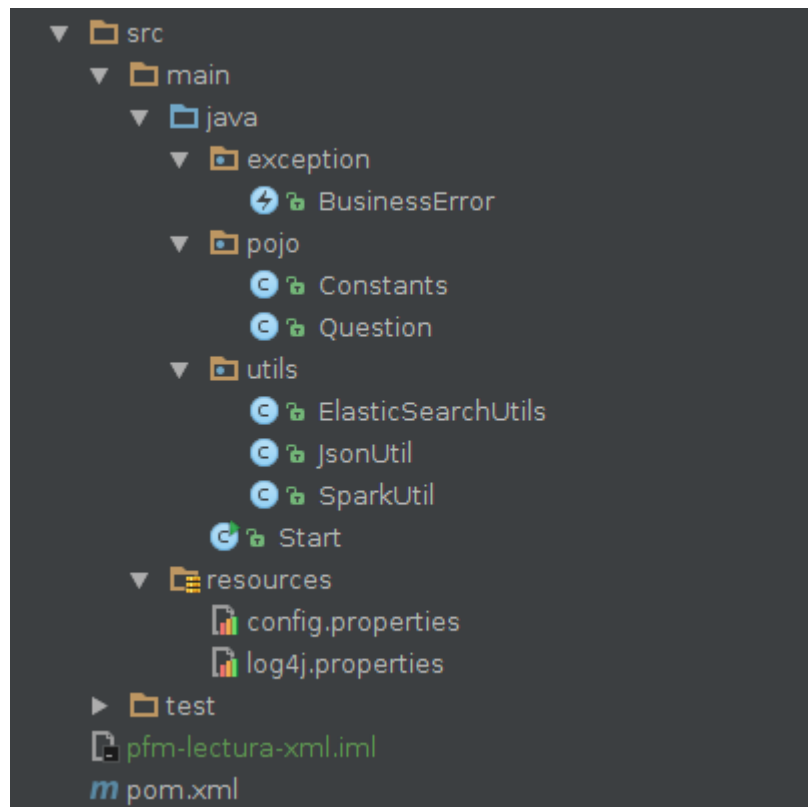
[https://coesta@bitbucket.org/coesta/pfm\\_batch.git](https://coesta@bitbucket.org/coesta/pfm_batch.git)

El fichero pom.xml tiene las dependencias del proyecto:

```
<properties>
  <spark.version>1.6.0</spark.version>
  <es.version>1.3.2</es.version>
  <jackson-databind.version>2.4.4</jackson-databind.version>
</properties>

<dependencies>
  <dependency>
    <groupId>org.apache.spark</groupId>
    <artifactId>spark-core_2.10</artifactId>
    <version>${spark.version}</version>
  </dependency>
  <dependency>
    <groupId>org.apache.spark</groupId>
    <artifactId>spark-sql_2.10</artifactId>
    <version>${spark.version}</version>
  </dependency>
  <dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-databind</artifactId>
    <version>${jackson-databind.version}</version>
  </dependency>
  <dependency>
    <groupId>org.elasticsearch</groupId>
    <artifactId>elasticsearch</artifactId>
    <version>${es.version}</version>
  </dependency>
  <dependency>
    <groupId>org.elasticsearch</groupId>
    <artifactId>elasticsearch-spark_2.10</artifactId>
    <version>2.3.0</version>
  </dependency>
  <dependency>
    <groupId>log4j</groupId>
    <artifactId>log4j</artifactId>
    <version>1.2.17</version>
  </dependency>
  <dependency>
    <groupId>com.holdenkarau</groupId>
    <artifactId>spark-testing-base_2.10</artifactId>
    <version>${spark.version}_0.3.3</version>
    <scope>test</scope>
  </dependency>
  <dependency>
    <groupId>org.apache.spark</groupId>
    <artifactId>spark-hive_2.10</artifactId>
    <version>${spark.version}</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```





Se ha configurado log4j y se ha creado un fichero de propiedades para los diferentes parámetros: nombre clusters, ruta fichero hdfs,.... Lo ideal sería que en producción el fichero de configuración estuviera en el classpath pero fuera del war y cualquier cambio en estos parámetros sería bastante sencillo y no implicaría generar un nuevo war.

Clase a clase:

### **BusinessException:**

Simplemente una clase que hereda de Exception para manejar los mensajes para las excepciones.

### **Constant:**

Clase que carga las properties del fichero de propiedades.

**Question:**

Es un objeto POJO una clase de dominio de la aplicación, es decir es la clase que se representa los posts, las preguntas y las respuestas. Simplemente tiene un constructor y getters y setters, aunque se le ha añadido la funcionalidad extra de transformar a json el objeto en uno de sus métodos.

```
public String toJson(){
    try {
        ObjectMapper mapper = new ObjectMapper();
        mapper.configure(SerializationFeature.WRITE_DATES_AS_TIMESTAMPS, false);
        mapper.setDateFormat(new SimpleDateFormat("yyyy-MM-dd HH:mm:ss.SSS"));
        return mapper.writeValueAsString(this);
    } catch (JsonProcessingException e) {
        return "";
    }
}
```

**ElasticsearchUtils:**

Manager para interactuar con elasticsearch.

Para conectarse al cluster de elasticsearch se ha utilizado el cliente Transport client. Este cliente utiliza el módulo de transporte de elasticsearch, este módulo publica sus servicios entre el rango de puerto 9300 y 9400. Transport client se conecta y se comunican con el cluster utilizando el puerto 9300. Este puerto 9300 también es el utilizado por los nodos del cluster para comunicarse entre ellos.

Transport client se conecta al cluster y envía las peticiones a uno de los nodos, se configura de la siguiente forma:

- Se le indica uno, varios o todos los nodos del cluster remoto. En la aplicación sólo se le indica uno por lo que se pone la propiedad **client.transport.sniff** a **true** para que a partir del nodo que se le ha pasado dado, descubra el resto de nodos que forman el cluster.

- Si el cluster tuviera el nombre de los cluster de elasticserch por defecto no haría falta indicarle el nombre del cluster, pero en este caso si es necesario indicárselo ya que no es elasticsearch, sino cluster-elasticsearch.
- Así estaría establecida la conexión con lo que para la comunicación las peticiones serán enviadas en *round robin* a la lista de nodos que forman el cluster.

```
public void openConexionBBDD(String clusterName, String server,
int port) {
    log.debug("Abriendo conexion a BBDD");
    Settings settings = ImmutableSettings.settingsBuilder()
        .put("cluster.name", clusterName)
        .put("client.transport.ignore_cluster_name", false)
        .put("client.transport.sniff", true)
        .build();

    InetAddress address = InetAddress.getLocalHost();
    client = new TransportClient(settings).addTransportAddress(new
        InetSocketAddress(address.getHostAddress(), port));

    log.debug("Finalizado Abriendo conexion a BBDD");
}

public void closeConexionBBDD() {
    log.debug("Cerrando conexion a BBDD");
    client.close();
    log.debug("finalizado cerrando conexion a BBDD");
}
```

Método encargado de crear el índice, si existe lo borra primero:

```
public void createIndex() throws BusinessException {
    log.debug("Creando indice");
    try {
        deleteIndexIfExist();

        // Create Index and set settings and mappings
        CreateIndexRequestBuilder createIndexRequestBuilder = client.admin()
            .indices().prepareCreate(index);
        XContentBuilder mappingBuilder = createMapping(type);
        createIndexRequestBuilder.addMapping(type, mappingBuilder);
        createIndexRequestBuilder.execute().actionGet();
    } catch (Exception e) {
        throw new BusinessException("No se ha podido crear el indice");
    }
    log.debug("finalizado creando indice");
}

public void deleteIndexIfExist() {
    log.debug("Borrando indice");
    final IndicesExistsResponse res =
        client.admin().indices().prepareExists(index).execute().actionGet();
    if (res.isExists()) {
        final DeleteIndexRequestBuilder delIdx =
            client.admin().indices().prepareDelete(index);
        delIdx.execute().actionGet();
    }
    log.debug("finalizado borrando indice");
}
```

Método encargado de crear el índice e indicar el mapeo de los datos:

```
public void createIndex(String indexName, String documentType) throws
IOException {
    // Create Index and set settings and mappings
    CreateIndexRequestBuilder createIndexRequestBuilder = client.admin()
        .indices().prepareCreate(indexName);
    XContentBuilder mappingBuilder = createMapping(documentType);
    createIndexRequestBuilder.addMapping(documentType, mappingBuilder);
    createIndexRequestBuilder.execute().actionGet();
}
```

Para el mapeo de los datos se necesita un documento json, se ha utilizado el helper que provee elasticsearch para crear objetos JSON: XContentBuilder.

Al campo `_id` que es el identificador del documento se mapea con el id del documento, si no se realizará así elasticsearch crearía automáticamente un id. El significado del resto de valores se ha explicado anteriormente.

```
public XContentBuilder createMapping(String documentType) throws IOException {
    XContentBuilder mappingBuilder =
    jsonBuilder().startObject().startObject(documentType)
        .startObject("_id")
        .field("path", "id")
        .endObject()
        .startObject("properties")
        .startObject("id")
        .field("type", "long")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("postTypeId")
        .field("type", "long")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("parentId")
        .field("type", "long")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("creationDate")
        .field("type", "date")
        .field("format", "yyy-MM-dd HH:mm:ss.SSS")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("score")
        .field("type", "long")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("ownerUserId")
        .field("type", "long")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("lastActivityDate")
        .field("type", "date")
        .field("format", "yyy-MM-dd HH:mm:ss.SSS")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("viewCount")
        .field("type", "long")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("body")
        .field("type", "string")
        .field("index", "no")
    }
```

```

        .endObject()
        .startObject("favoriteCount")
        .field("type", "long")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("answerCount")
        .field("type", "long")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("commentCount")
        .field("type", "long")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("acceptedAnswerId")
        .field("type", "long")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("gradoDificultad1")
        .field("type", "double")
        .field("index", "not_analyzed")
        .endObject()
        .startObject("tags")
        .field("type", "string")
        .field("store", "yes")
        .field("index", "not_analyzed")
        .endObject()
        .endObject();

    return mappingBuilder;
}

```

Para indexar un documento, simplemente se le pasa el índice, el tipo y el documento que deberá ser un String json.

```

public void addDocument(String content) {
    addDocument(index, type, content);
}

public void addDocument(String indexName, String type, String content) {
    IndexResponse response = null;

    IndexRequestBuilder requestBuilder = client.prepareIndex(indexName, type)
        .setSource(content);

    response = requestBuilder.execute().actionGet();
}

```

## JsonUtils:

Clase para el tratamiento de las líneas tipo xml.

Básicamente es un solo método que se encarga de convertir el String de tipo xml en un objeto json, este objeto es un objeto de la clase Question.

Para esto se utiliza el api de java para el manejo de documentos DOM a través del uso de clases como Element, DocumentBuilder, etc.

En los posts el body es la información de la web y en el fichero xml viene con formato html, se limpia completamente este formato para obtener texto plano.

### SparkUtils:

Esta clase se usa para el procesamiento de los datos con spark.

Contiene un método principal

```
public JavaRDD processData(JavaSparkContext sc, String inputFile) {
```

Se detalla paso a paso que operaciones se están realizando en este método:

1. Se lee del HDFS el fichero a tratar (previamente se ha subido a HDFS), la ruta del fichero se obtiene de las properties properties

```
JavaRDD<String> lines = sc.textFile(inputFile);
```

2. Se filtran las líneas para procesar sólo las que son de tipo row, se vio anteriormente que en el fichero pueden venir otras etiquetas de xml pero sólo nos interesarían estas.

```
JavaRDD<String> linesSinPosts = lines.filter(createFunctionFilter());
```

```
public static Function<String, Boolean> createFunctionFilter(){
    return new Function<String, Boolean>() {
        @Override
        public Boolean call(String s) throws Exception {
            return (s.contains("row"));
        }
    };
}
```

3. Se convierte la línea de tipo xml a un objeto tipo Question (como se vio anteriormente este objeto representa nuestras preguntas y respuestas). Se convierte en una tupla donde el primer parámetro es el id y el segundo el objeto. Se filtran los post que no son de tipo 1 ni 2 (no son ni preguntas ni respuestas).

```
JavaPairRDD<String, Question> rowWithkey = linesSinPosts.mapToPair(
    new PairFunction<String, String, Question>() {
        public Tuple2<String, Question> call(String s) {
            Question q = JsonUtil.convertStringToObject(s);
            return new Tuple2<>(q.getIdString(), q);
        }
    }).filter(new Function<Tuple2<String, Question>, Boolean>() {
    @Override
```

```

    public Boolean call(Tuple2<String, Question> s) throws Exception {
        return (s._2.getPostTypeId() == 1 || s._2.getPostTypeId() == 2);
    }
});

```

#### 4. Se desea obtener la fecha de creación de la respuesta acertada

- a. `mapToPair`: por lo que se genera un par (id\_pregunta-id\_respuesta, fecha\_creación de respesta).
  - i. Si es una pregunta:
    1. Con respuesta acertada: se devuelve el (id\_pregunta-id\_resp\_acertada, "")
    2. Sin respuesta acertada ("","")
  - ii. Si es respuesta: (id\_parentId-id\_respuesta, fecha\_creacion), el id\_parentId es el id de la pregunta al que corresponde esa respuesta.
- b. `reduceByKey`: Devuelve la clave del par con un --1 (es un valor que utilizaremos para saber que esa pregunta si tenemos que procesarla), este valor solo se va a poner en los pares que se agrupen es decir cuando una pregunta tenga respuesta correcta.
- c. `filter`: Se eliminan los pares vacios ("","") que se hayan podido generar anteriormente.
- d. `filter`: Se eliminan los paras que no tengan -1.
- e. `mapToPair`: Se devuelve el id de la pregunta con la fecha de la respuesta acertada.

```

JavaPairRDD<String, String> respCorrectasFechas = linesSinPosts.mapToPair(
    new PairFunction<String, String, String>() {
        public Tuple2<String, String> call(String s) {
            Question q = JsonUtil.convertStringToObject(s);
            if (q.getPostTypeId() == 1) {
                if (q.getAcceptedAnswerId() != 0) {
                    return new Tuple2<>(q.getIdString() + "--" +
q.getAcceptedAnswerIdString(), " ");
                } else {
                    return new Tuple2<String, String>("", "");
                }
            } else {
                return new Tuple2<>(q.getParentIdString() + "--" +
q.getIdString(), q.getCreationDateString());
            }
        }
    }).reduceByKey(
    new Function2<String, String, String>() {
        public String call(String value0, String value1) {

```

```

        if (value1.equalsIgnoreCase(" "))
            return value0 + "--1";
        else
            return value1 + "--1";
    }

    ;
}

).filter(new Function() {
    public Boolean call(Tuple2<String, String> value1) throws Exception
    {
        return !(value1._1.equalsIgnoreCase("")) &&
        value1._2.equalsIgnoreCase("");
    }
})

).filter(new Function() {
    public Boolean call(Tuple2<String, String> value1) throws Exception
    {
        return value1._2.indexOf("--1") > 0;
    }
})

).mapToPair(
    new PairFunction() {
        public Tuple2<String, String> call(Tuple2<String, String> t) {
            return new Tuple2<>(t._1.substring(0, t._1.indexOf("--")), t._2.substring(0,
            t._2.lastIndexOf("--")-1));
        }
    }
);

```

5. Una el RRD inicial con las respuestas y preguntas con el de las preguntas con la fecha adecuada, por eso devuelve un optional String.

```

JavaPairRDD<String, Tuple2<Question, Optional<String>>> rowWithkeyAndDate =
rowWithkey.leftOuterJoin(respCorrectasFechas);

```

6. Finalmente se obtienen las preguntas y en el atributo gradodificultad1 el valor que le corresponde.

- filter:** Si es de tipo=1 pero no tiene respuesta acertada se elimina.
- Map:** Para las preguntas calcula el grado de dificultad basándose en el algoritmo explicado anteriormente y tomando como fecha de referencia del fichero la fecha que está en el properties, con las respuestas no realizaría ningún calculo.

```

JavaRDD pregsAnswers= rowWithkeyAndDate
    .filter(new Function

```



```

    )

    .map(new Function

```

## Tester:

Está clase tiene test unitarios (es una pequeña muestra simplemente).

Para realizar los test unitarios las transformaciones realizadas sobre los RDD deberán estar en métodos, por lo que normalmente es necesaria una refactorización del código para ello.

Se utiliza la librería holdenkarau.

## Start:

Main del batch. Tiene la unión de todas las piezas para conseguir el procesado y almacenamiento de los datos del fichero Posts. Lo que destaca de este main es que para el almacenamiento de los datos en elasticsearch se utiliza `foreachPartition` esto es un foreach para el trozo de RDD que tenga cada partición.

```

public static void main(String[] args) {

    try {
        log.debug("Iniciando...");
        c.loadConstants();

        /*Conexion elasticsearch mapeo y indice*/
        elasticSearchService = new ElasticSearchUtils();
        elasticSearchService.openConexionBBDD(c.nameElasticSearchCluster,
        c.hostElasticSearchCluster, c.portElasticSearchCluster);
        elasticSearchService.createIndex();
        elasticSearchService.closeConexionBBDD();

        JavaSparkContext sc = new JavaSparkContext(new
        SparkConf().setMaster(c.masterSparkCluster).setAppName("PFM"));

        sparkUtil=new SparkUtil();
        JavaRDD data = sparkUtil.processData(sc, c.inputFile);

        data.foreachPartition(new VoidFunction<Iterator<String>>() {
            public void call(Iterator<String> s) {
                ElasticSearchUtils es = new ElasticSearchUtils();
                while (s.hasNext()) {
                    es.addDocument(s.next());
                }
                es.closeConexionBBDD();
            }
        });

    } catch (Exception e){
        if(e instanceof BusinessException)
            log.error(e.getMessage());
        else
            e.printStackTrace();
        System.exit(1);
    }
}

```

### Parte Web

Se desea obtener un fichero pdf que sea un ejemplo de examen. Para la creación de este fichero se deberá de poder elegir el número de preguntas de cada tipo así como el grado de dificultad.

### **Estructura del proyecto web**

La parte web es la encargada de la creación de los exámenes y además ofrece dos gráficos de visualización basados en D3 para poder analizar la información a partir de la cual se obtienen los exámenes.

Se ha creado un proyecto maven siguiendo un patrón de arquitectura modelo-vista-controlador con spring (framework para el desarrollo de aplicaciones de código

abierto para la plataforma Java), para la interacción con elasticsearch se ha utilizado spring-data-elasticsearch (submodulo del framework spring).

Para el control de versiones se está utilizando git con bitbucket:

[https://coesta@bitbucket.org/coesta/pfm\\_rest.git](https://coesta@bitbucket.org/coesta/pfm_rest.git)

Con maven realizamos la gestión de dependencias indicándolas en el fichero pom.xml, las dependencias del proyecto son:

```
<properties>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  <java.version>1.8</java.version>
  <spring.version>4.1.5.RELEASE</spring.version>
  <es.version>1.2.2.RELEASE</es.version>
</properties>

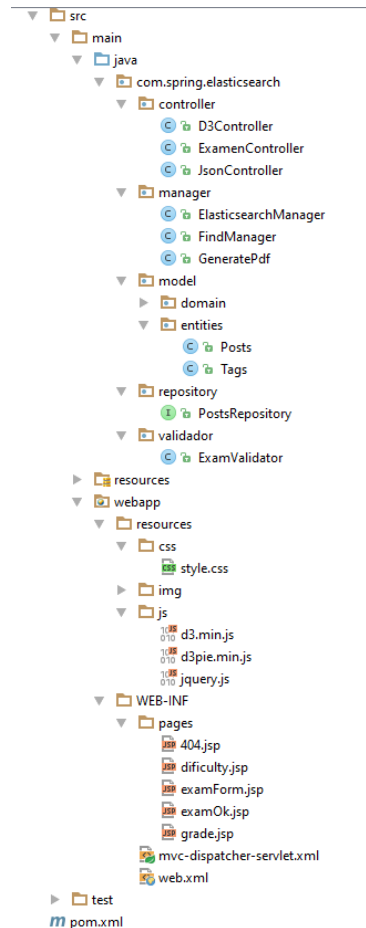
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-core</artifactId>
    <version>${spring.version}</version>
  </dependency>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>${spring.version}</version>
  </dependency>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-webmvc</artifactId>
    <version>${spring.version}</version>
  </dependency>
  <dependency>
    <groupId>org.springframework.data</groupId>
    <artifactId>spring-data-elasticsearch</artifactId>
    <version>${es.version}</version>
    <exclusions>
      <exclusion>
        <groupId>org.elasticsearch</groupId>
        <artifactId>elasticsearch</artifactId>
      </exclusion>
    </exclusions>
  </dependency>
  <dependency>
    <groupId>org.elasticsearch</groupId>
    <artifactId>elasticsearch</artifactId>
    <version>1.3.2</version>
  </dependency>
  <dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>jstl</artifactId>
    <version>1.1.2</version>
  </dependency>
  <dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>servlet-api</artifactId>
    <version>2.5</version>
  </dependency>
  <dependency>
    <groupId>>taglibs</groupId>
    <artifactId>standard</artifactId>
    <version>1.1.2</version>
  </dependency>
</dependencies>
```

```

<groupId>com.itextpdf</groupId>
<artifactId>itextpdf</artifactId>
<version>5.5.6</version>
</dependency>
</dependencies>

```

La estructura del proyecto es la siguiente:



Los ficheros de configuración son el web.xml y mvc-dispatcher-servlet.xml básicamente indican que todas las peticiones de tipo .htm sean las tratadas por spring, donde se encuentran las jsp de la vista, los controladores, etc. Es decir, son los ficheros de configuración de spring.

Destacar que es aquí donde se indica la conexión con elasticsearch:

```
<bean name="elasticsearchTemplate"
class="org.springframework.data.elasticsearch.core.ElasticsearchTemplate">
  <constructor-arg name="client" ref="client"/>
</bean>
```

```
<elasticsearch:transport-client id="client" cluster-nodes="10.211.55.101:9300,
someip:9300"/>
```

Como se ha indicado en se utiliza un patrón MVC, con lo que se desacopla la parte de la vista de la lógica de negocio y los almacenes de datos.

- **Vista:** Muestra la información a través de una interfaz de usuario. Pueden existir múltiples vistas del modelo. Cada vista tiene asociado un componente controlador con el que interactúa.

En este proyecto se compone de una parte para la creación de exámenes, FormExam.jsp, con sus estilos y las imágenes. Dando como resultado:

Y por otra parte esta la visualización de D3, esta parte consta de dos jsp:

La primera nos permite introducir los tags para obtener el número de preguntas que existen por cada tags.

Contiene un formulario para poder introducir los tags que se desean aparecer en la visualización.

```
<form:form method="POST" action="grade.htm" commandName="examen">
<:forEach items="${examen.questions}" var="question" varStatus="uStatus">
  <div class="pregunta">
    <div id='tags' class='atributo'>
      <label for="tags" class='etiqueta'>Tags:</label>
      <form:input class="textarea" path="questions[${uStatus.index}].tags"
        title="Tags a buscar separados por comas"/>
    </div>
  </div>
</forEach>
```

```
<input type="submit" name="accion" value="buscar" title="buscar"/>

<form:errors path="*" cssClass="errorblock2" element="div"/>
```

Y la visualización:

```
var tagsLong = ${tagsLong};
var color = '#a6d1e1';
var tagsString = ${tagsString};
if (typeof tagsString !== 'undefined' && tagsString.length > 0) {

    var grid = d3.range(25).map(function (i) {
        return {'x1': 0, 'y1': 0, 'x2': 0, 'y2': 480};
    });

    var xscale = d3.scale.linear()
        .domain([0, Math.max.apply(Math, tagsLong)])
        .range([0, 722]);

    var yscale = d3.scale.linear()
        .domain([0, tagsString.length])
        .range([0, 480]);

    var colorScale = d3.scale.quantize()
        .domain([0, tagsString.length])
        .range(color);
```

La segunda nos muestra un gráfico con el tanto por ciento de preguntas que hay para un rango de dificultad:

```
</div>
<div id="pieChart"></div>
<script>
    var data = [];
    var url="${home}/mvc_FormExamen/string.json"
    d3.json(url, function(json) {
        $.each(json, function(d,i) {
            data.push({
                label: i.label,
                value: i.value
            })
        })
    })

    var pie = new d3pie("pieChart", {
        "header": {
            "title": {
                "text": "Grados de dificultad",
                "fontSize": 22,
                "font": "verdana"
            },
        },
        "size": {
            "canvasHeight": 400,
            "canvasWidth": 590
        },
        "data": {
            "content": data
        },
        "labels": {
            "outer": {
                "pieDistance": 32
            }
        }
    });
});
</script>
```

Es necesaria la librería de jquery y de d3, así como estilos.

- **Controlador:** Reciben las entradas de las vistas. Los eventos (acciones del usuario) son traducidos a solicitudes de servicio para el modelo o la vista.

Existe un controlador por jsp.

### ExamController:

Controlador específico para la parte de creación de exámenes de la aplicación. Se encarga de mostrar la vista e identificar las acciones que desea realizar el usuario para pasárselas a las clases de negocio.

```
@RequestMapping(value = "/examen.htm",method = RequestMethod.POST)
public String processSubmit(
    @RequestParam(value = "accion") String accion, @ModelAttribute("examen")
    Exam ex,
    BindingResult result, SessionStatus status, HttpServletRequest request,
    HttpServletResponse response) {

    if (accion.equalsIgnoreCase(ANADIR)) {
        ex.getQuestions().add(new Question());
        return "examForm";
    }else if(accion.startsWith(CONSULTAR)){
        int numberFind = Integer.valueOf(accion.substring(accion.indexOf("_")
+ 1, accion.length()));
        examValidator.validateQuestion(ex.getQuestions().get(numberFind),
result,numberFind);
        if(!result.hasErrors()) {
            status.setComplete();
            boolean
encontrado=fm.searchExistFind(ex.getQuestions().get(numberFind), numberFind);
            if (encontrado)
                ex.setResult("Existen posts con los criterios buscados en la
fila " + (numberFind + 1));
            else
                ex.setResult("No existen posts con los criterios buscados en
la fila " + (numberFind + 1));
            return "examForm";
        }else if(accion.startsWith(ELIMINAR)){
            if(ex.getQuestions().size()>1) {
                int numeroeliminar =
Integer.valueOf(accion.substring(accion.indexOf("_") + 1, accion.length()));
                ex.getQuestions().remove(numeroeliminar);
            }else
                result.addError(new ObjectError("", "No se puede eliminar si solo
hay un elemento"));
            return "examForm";
        }else{
            examValidator.validate(ex, result);
            if(!result.hasErrors()) {
                if(!fm.generatePDF(ex,response)) {
                    result.addError(new ObjectError("", "Revise las preguntas
pedidas porque para alguna de ellas no se encuentras datos"));
                    return "examForm";
                }
            }
            return "examKo";
        }
    }
}
```

```

        }else {
            return "examForm";
        }
    }
}

```

### D3Controller:

Controlador encargado de la vista para la obtención del número de preguntas existentes para un tags.

### JsonController:

Controlador para la vista que muestra los porcentajes de los rangos para los grados de dificultad.

Aquí existen dos métodos el método que se encarga de gestionar las peticiones hacia la página y el método que devuelve un objeto json que necesitará la vista.

```

@RequestMapping(value = "/string.json", method = RequestMethod.GET, produces =
"application/json")
@ResponseBody
public String bar() {

```

Para trabajar con elasticsearch se necesita definir las entidades de dominio y una clase repositorio que soporte las operaciones CRUD. Las entidades de dominio para los datos de elasticsearch son objetos POJO: Posts y Tags.

- **Modelo:** Encapsula los datos y las funcionalidades. El modelo es independiente de cualquier representación de salida y/o comportamiento del controlador o la vista.

### ExamValidador:

Se encarga de las validaciones de los datos introducidos en las jsp. Desde el controlador se envía el objeto y los métodos de esta clase valida que contenga datos correctos.



**ElasticsearchManager:**

Tiene la lógica de negocio de las consultas contra elasticsearch. Es la capa por encima del repositorio de elasticsearch que realiza el procesamiento de los datos para realizar las consultas necesarias y obtener los resultados pedidos.

```
@Service
@EnableElasticsearchRepositories(basePackages = "com.spring.elasticsearch.repository")
public class ElasticsearchManager {
    @Autowired
    private PostsRepository repository;
```

**PostsRepository:**

Interfaz de acceso a los datos contenidos en elasticsearch, como se comentó antes spring a través de spring-data-elasticsearch provee de varias interfaces de las que se debe heredar y que contienen los métodos necesarios para el acceso a base de datos. Spring, por su parte, se encarga de implementar esa interfaz.

La interface hereda de la interfaz elasticsearchRepository. En ella se le indica el tipo del objeto DTO base y el tipo del índice.

```
@Repository
public interface PostsRepository extends ElasticsearchRepository<Posts, String> {
```

La interfaz ElasticsearchRepository proporciona los métodos básicos de acceso a datos, pero se pueden añadir métodos de acceso específicos con queries más complejas gracias a la anotación @Query. Con esta anotación, se permite diseñar queries más complejas (con joins, consultas con varios parámetros, etc.).

```
@Query("{ \"terms\" : { \"tags\" : [ \"?0\", \"?1\" ], \"minimum_should_match\" : 2 }}")
Page<Posts> findByTagsUsingCustomQuery(String tags1, String tags2, Pageable pageable);
```

**GeneratePDF:**

Clase encargada de generar el documento PDF que contiene el examen. Utiliza PDFWriter.

**FindManager:**

Union entre ElasticsearchManager y GeneratePdf, se encarga de realizar consultas a elasticsearch y procesar el resultado realizando las transformaciones entre objetos de negocio y objetos dto, tratando sus datos.

```
public boolean generatePDF(Exam ex, HttpServletResponse response) {
    int i=0;
    for (Question q : ex.getQuestions()) {
        if(!searchExistFind( q, i))
            return false;
        i++;
    }
    List<ExamResponse> list = new ArrayList<ExamResponse>();
    for (Question q : ex.getQuestions()) {
        List<ExamResponse> list1 = new ArrayList<ExamResponse>();
        int numberOfQuestion = Integer.parseInt(q.getNumber());
        double difficultyOfQuestion = Double.parseDouble(q.getDifficulty());
        double difficultyOfQuestionto = Double.parseDouble(q.getDifficultyto());
        String[] tags = null;
        if (!q.getTags().trim().equalsIgnoreCase("")) {
            tags = q.getTags().split(",");
        }
        if (tags == null || tags.length == 0) {
            list1 =
esManager.findAndReturnByDificultyBetween(difficultyOfQuestion,difficultyOfQuestion
to, numberOfQuestion);

        } else {
            list1 =
esManager.findAndReturnByTagsInAndGradoDificultadlBetween(tags,
difficultyOfQuestion, difficultyOfQuestionto,numberOfQuestion);
        }
        list = ListUtils.union(list1, list);
    }
    generatePdf.doPdf(list,response);
    return true;
}
```

Objetos de negocio y entidades

Como entidades tenemos Posts y tags son las clases de mapeo a elasticsearch, los datos guardados en elasticsearch se corresponden a estas clases. Básicamente son POJOs.

```
@Document(indexName = "posts", type = "row")
public class Posts {
```

Los objetos de negocio están formados por estas tres clases: Exam, Question y ExamResponse

Básicamente la clase Exam contiene una lista de cuestión (cada question se corresponde a una fila en la vista) y la clase examResponse tiene dos atributos de tipo string que se corresponden con las preguntas y las respuestas.

## Implantación en el cluster

El video con la implantación está disponible en la siguiente url:

<https://www.youtube.com/watch?v=BQ3GMWjYGUk&feature=youtu.be>

Para la implantación en el cluster se ha partido de un cluster en vagrant disponible en github:

<https://github.com/vangi/vagrant-hadoop-2.4.1-spark-1.0.1>

```
node1: HDFS NameNode + Spark Master
node2: YARN ResourceManager + JobHistoryServer + ProxyServer
node3: HDFS DataNode + YARN NodeManager + Spark Slave
node4: HDFS DataNode + YARN NodeManager + Spark Slave
```

Se han adaptado las versiones para que coincidieran con las versiones que se necesitaban, quedando de la siguiente manera: hadoop-2.4.1, spark-1.6.0, jdk-7u80-linux-x64, también ha sido necesaria la modificación de algún script para un tema de librerías.

Se inicializa el clúster: vagrant up

```
=> node1: Running provisioner: shell...
node1: Running: C:/Users/Tamara/AppData/Local/Temp/vagrant-shell20160708-9564-18qfnw0.sh
=> node1: setup java
=> node1: installing oracle jdk
=> node1: setting up java
=> node1: creating java environment variables
=> node1: Running provisioner: shell...
node1: Running: C:/Users/Tamara/AppData/Local/Temp/vagrant-shell20160708-9564-k1rluo.sh
=> node1: setup hadoop
=> node1: install hadoop from local file
=> node1: creating hadoop directories
=> node1: copying over hadoop configuration files
=> node1: creating hadoop environment variables
=> node1: Running provisioner: shell...
node1: Running: C:/Users/Tamara/AppData/Local/Temp/vagrant-shell20160708-9564-1a9hr7h.sh
=> node1: setup hadoop slaves
=> node1: modifying /usr/local/hadoop/etc/hadoop/slaves
=> node1: adding node3
=> node1: adding node4
=> node1: Running provisioner: shell...
node1: Running: C:/Users/Tamara/AppData/Local/Temp/vagrant-shell20160708-9564-1g48xtp.sh
=> node1: setup spark
=> node1: install spark from local file
=> node1: setup spark
=> node1: creating spark environment variables
=> node1: Running provisioner: shell...
node1: Running: C:/Users/Tamara/AppData/Local/Temp/vagrant-shell20160708-9564-kdof5x.sh
=> node1: setup hadoop slaves
=> node1: modifying spark slaves
=> node1: adding node3
=> node1: adding node4
```

Se levantan los 4 nodos

Ahora es necesario inicializar el cluster, hay que conectarse por ssh. Se ha utilizado el cliente de ssh putty.

```
sudo $HADOOP_PREFIX/bin/hdfs namenode -format myhadoop
```

Inicializamos HDFS en el nodo1:

```
sudo $HADOOP_PREFIX/sbin/hadoop-daemon.sh --config $HADOOP_CONF_DIR --script hdfs start namenode
sudo $HADOOP_PREFIX/sbin/hadoop-daemons.sh --config $HADOOP_CONF_DIR --script hdfs start datanode
```

En la siguiente url se puede comprobar que está disponible el entorno hdfs:

The screenshot shows the 'Datanode Information' page in the Hadoop web interface. It displays a table of nodes in operation and a section for decommissioning nodes.

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
node4 (10.211.55.104:50010)	2	In Service	7.28 GB	1.22 GB	3.74 GB	2.32 GB	19	1.22 GB (16.77%)	0	2.4.1
node3 (10.211.55.103:50010)	2	In Service	7.28 GB	1.22 GB	3.74 GB	2.32 GB	19	1.22 GB (16.77%)	0	2.4.1

Below the 'In operation' table, there is a 'Decommissioning' section with a table showing nodes being decommissioned, including their last contact, under replicated blocks, blocks with no live replicas, and under replicated blocks in files under construction.

En el nodo2 inicializamos YARN.

```
sudo $HADOOP_YARN_HOME/sbin/yarn-daemon.sh --config $HADOOP_CONF_DIR start resourcemanager
sudo $HADOOP_YARN_HOME/sbin/yarn-daemons.sh --config $HADOOP_CONF_DIR start nodemanager
sudo $HADOOP_YARN_HOME/sbin/yarn-daemon.sh start proxyserver --config $HADOOP_CONF_DIR
sudo $HADOOP_PREFIX/sbin/mr-jobhistory-daemon.sh start historyserver --config $HADOOP_CONF_DIR
```

The screenshot shows the 'Nodes of the cluster' page in the Hadoop web interface. It displays a table of cluster metrics and a detailed view of the nodes in the cluster.

Cluster Metrics													
Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes	
0	0	0	0	0	0 B	16 GB	0 B	2	0	0	0	0	

Below the cluster metrics, there is a table showing the details of the nodes in the cluster. The table includes columns for Rack, Node State, Node Address, Node HTTP Address, Last health-update, Health-report, Containers, Mem Used, Mem Avail, and Version.

Rack	Node State	Node Address	Node HTTP Address	Last health-update	Health-report	Containers	Mem Used	Mem Avail	Version
default-rack	RUNNING	node4:39581	node4:8041	8-Jul-2016 17:07:14	0	0 B	8 GB	8 GB	2.4.1
default-rack	RUNNING	node3:33605	node3:8042	8-Jul-2016 17:07:09	0	0 B	8 GB	8 GB	2.4.1

Se ha instalado elasticsearch posteriormente en dos de los nodos con los siguientes comandos:

```
#Se descarga wget ya que no está instalado por defecto
yum install -y wget
# Se descarga elasticsearch versión 1.3.2
wget
https://download.elasticsearch.org/elasticsearch/elasticsearch/elasticsearch-1.3.2.tar.gz
#Se descomprime
Sudo tar xzf elasticsearch-1.3.2.tar.gz
Sudo rm elasticsearch-1.3.2.tar.gz

#Se cambia de directorio
Sudo mv elasticsearch-1.3.2 /opt
Sudo ln -s elasticsearch-1.3.2 elasticsearch
```

```
Transaction Test Succeeded
Running Transaction
  Installing : wget-1.12-8.el6.x86_64                1/1
  Verifying  : wget-1.12-8.el6.x86_64                1/1

Installed:
  wget.x86_64 0:1.12-8.el6

Complete!
[root@node1 vagrant]# wget https://download.elasticsearch.org/elasticsearch/elasticsearch/elasticsearch-1.3.2.tar.gz
--2016-07-08 19:11:46-- https://download.elasticsearch.org/elasticsearch/elasticsearch/elasticsearch-1.3.2.tar.gz
Resolving download.elasticsearch.org... 23.21.156.42, 23.23.137.104, 23.21.114.17, ...
Connecting to download.elasticsearch.org|23.21.156.42|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 27949096 (27M) [application/x-tar]
Saving to: "elasticsearch-1.3.2.tar.gz"

83% [=====>] 23,412,308 3.64M/s eta 2s
```

Se ha configurado el fichero en cada uno de los nodos para formar el cluster:

```
vi config/elasticsearch.yml
```

El fichero de configuración de elasticsearch queda de la siguiente manera:

Nodo1:

```
cluster.name: cluster_elasticsearch
```

```

node.master: true
node.data: true
index.number_of_shards: 2
index.number_of_replicas: 1
network.bind_host: 10.211.55.101
network.publish_host: 10.211.55.101
network.host: 10.211.55.101
transport.tcp.port: 9300
http.port: 9200
node.name: "nodo1"
discovery.zen.ping.timeout: 3s
discovery.zen.ping.unicast.hosts: ["10.211.55.101", "10.211.55.102"]

```

Nodo2:

```

cluster.name: cluster_elasticsearch
node.master: true
node.data: true
index.number_of_shards: 2
index.number_of_replicas: 1
network.bind_host: 10.211.55.102
network.publish_host: 10.211.55.102
network.host: 10.211.55.102
transport.tcp.port: 9300
http.port: 9200
node.name: "nodo2"
discovery.zen.ping.timeout: 3s
discovery.zen.ping.unicast.hosts: ["10.211.55.101", "10.211.55.102"]

```

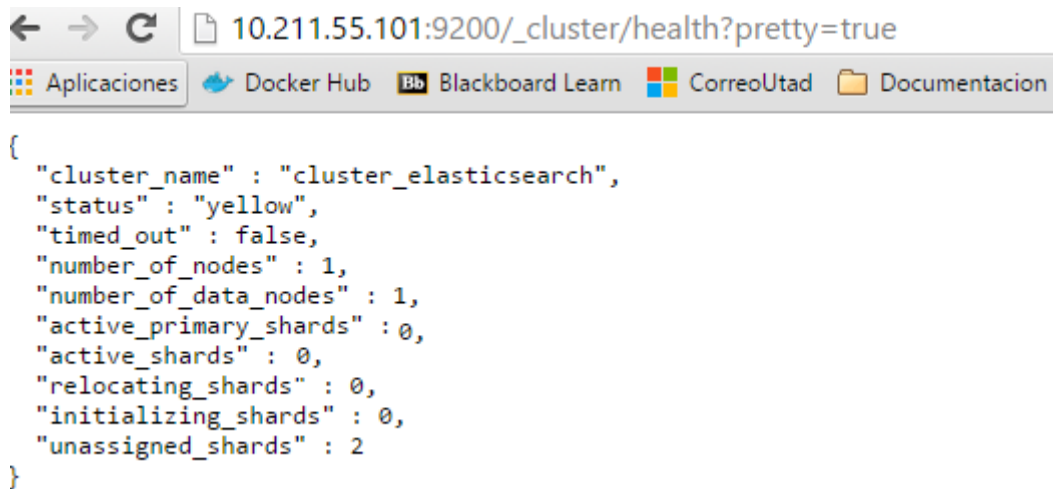
Se ha instalado también un plugin que es un servicio web para interactuar con el cluster elasticsearch:

```
sudo -E bin/plugin --install mobz/elasticsearch-head
```

Se levanta en el primer nodo:

```
sudo -E bin/elasticsearch
```

Aparece solo un nodo:

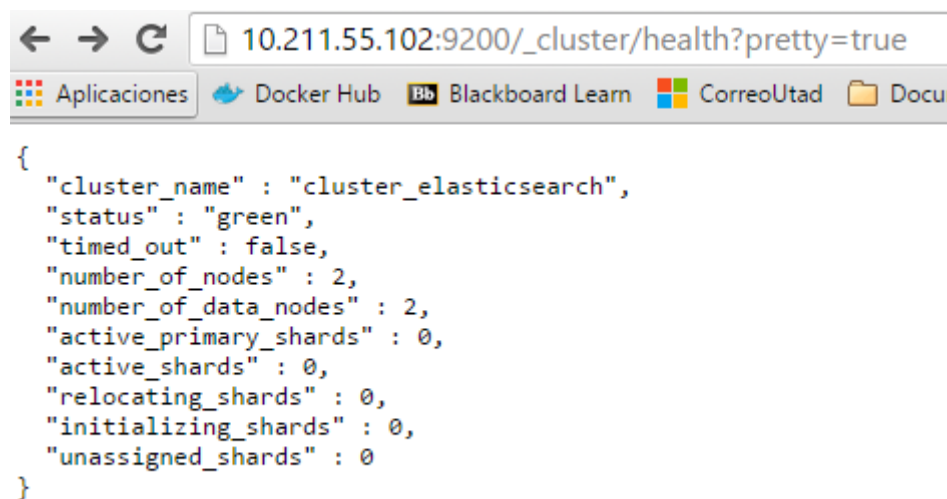


```

{
  "cluster_name" : "cluster_elasticsearch",
  "status" : "yellow",
  "timed_out" : false,
  "number_of_nodes" : 1,
  "number_of_data_nodes" : 1,
  "active_primary_shards" : 0,
  "active_shards" : 0,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 2
}

```

Se levanta en el segundo nodo:



```

{
  "cluster_name" : "cluster_elasticsearch",
  "status" : "green",
  "timed_out" : false,
  "number_of_nodes" : 2,
  "number_of_data_nodes" : 2,
  "active_primary_shards" : 0,
  "active_shards" : 0,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 0
}

```



Con lo que tenemos:

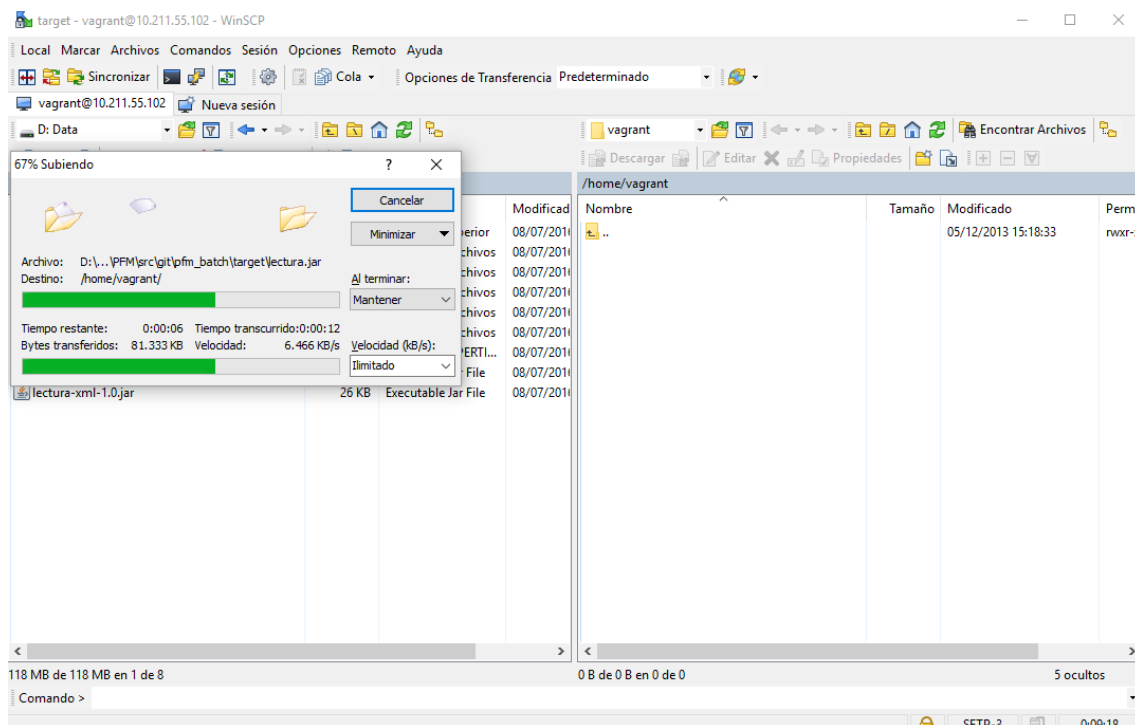


```

{
  "cluster_name": "cluster_elasticsearch",
  "nodes": {
    "Au6B_gY3RI27uZnaHRJFNA": {
      "timestamp": 1467998271333,
      "name": "Lionheart",
      "transport_address": "inet[/10.211.55.101:9300]",
      "host": "node1",
      "ip": [ "inet[/10.211.55.101:9300]", "NONE" ],
      "attributes": { "master": "true" },
      "indices": { "docs": { "count": 0, "deleted": 0 }, "store": { "size_in_bytes": 0, "thr
      "os": { "timestamp": 1467998271333, "uptime_in_millis": 1794, "load_average":
      "process": { "timestamp": 1467998271333, "open_file_descriptors": 129, "cpu":
      "jvm": { "timestamp": 1467998271334, "uptime_in_millis": 396853, "mem": { "t
      "thread_pool": { "percolate": { "threads": 0, "queue": 0, "active": 0, "rejected":
      "network": { "tcp": { "active_opens": 61, "passive_opens": 82, "curr_estab": 59
      "fs": { "timestamp": 1467998271335, "total": { "total_in_bytes": 7818289152, "t
      "transport": { "server_open": 26, "rx_count": 610, "rx_size_in_bytes": 72250, "b
      "http": { "current_open": 6, "total_opened": 13 },
      "fielddata_breaker": { "maximum_size_in_bytes": 639015321, "maximum_size"
    },
    "eyWBY0iiQImfV1j8CRpAYQ": {
      "timestamp": 1467998271078,
      "name": "Zarek",
      "transport_address": "inet[/10.211.55.102:9300]",
      "host": "node2",
      "ip": [ "inet[/10.211.55.102:9300]", "NONE" ],
      "attributes": { "master": "true" },
      "indices": { "docs": { "count": 0, "deleted": 0 }, "store": { "size_in_bytes": 0, "thr
      "os": { "timestamp": 1467998271078, "uptime_in_millis": 1827, "load_average":
      "process": { "timestamp": 1467998271079, "open_file_descriptors": 124, "cpu":
      "jvm": { "timestamp": 1467998271079, "uptime_in_millis": 288000, "mem": { "t
      "thread_pool": {
        "percolate": {
          "threads": 0,
          "queue": 0,
          "active": 0,

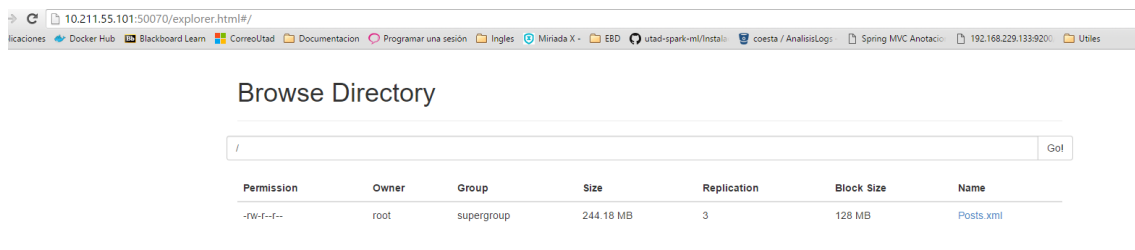
```

Para subir el fichero y el programa al cluster usamos el cliente winscp



En el nodo1: `hadoop fs -put Posts.xml /`

```
[root@node1 vagrant]# hadoop fs -put Posts.xml /
Java HotSpot(TM) 64-Bit Server VM warning: You have loaded library /usr/local/hadoop-2.4.1/lib/native/libhadoop.so.1.0.0 which might have disabled stack guard.
The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c <libfile>',
or link it with '-z noexecstack'.
16/07/08 19:01:50 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
[root@node1 vagrant]#
```



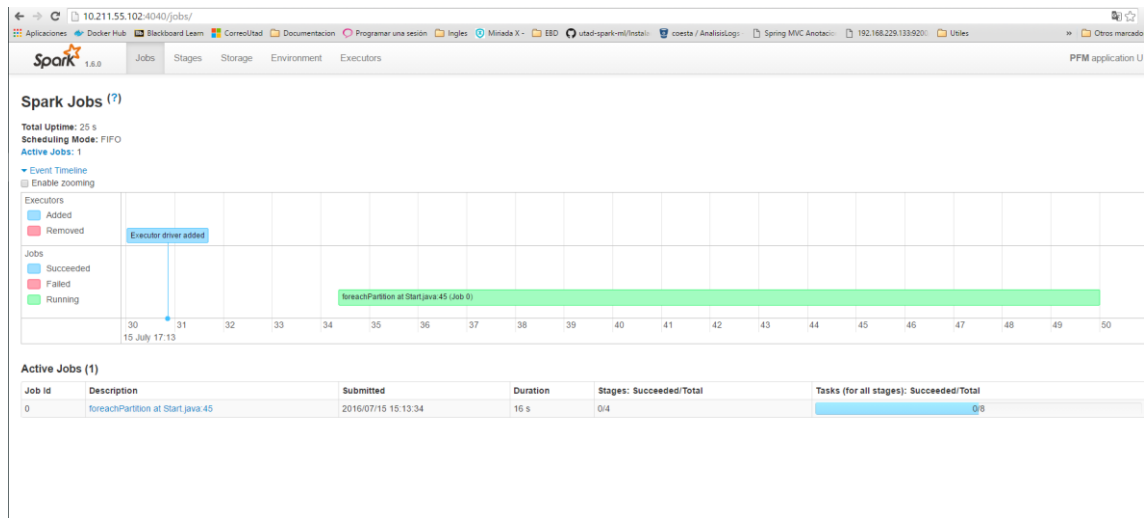
Si lo ejecutamos en modo client en el cluster:

```
$SPARK_HOME/bin/spark-submit --class Start \
  --master yarn \
  --executor-cores 2 \
  --num-executors 10 \
  /home/vagrant/lectura.jar
```

Los parámetros estarían en el fichero config.properties y serían:

```
cluster.elasticsearch.name=cluster_elasticsearch
cluster.elasticsearch.host=10.211.55.102
cluster.elasticsearch.port=9300
cluster.spark.master=local[5]
inputFile = hdfs://node1/Posts.xml
DAY_FILE = 2016-03-06 04:00:36.443
```

Se ejecuta:



Y obtenemos:

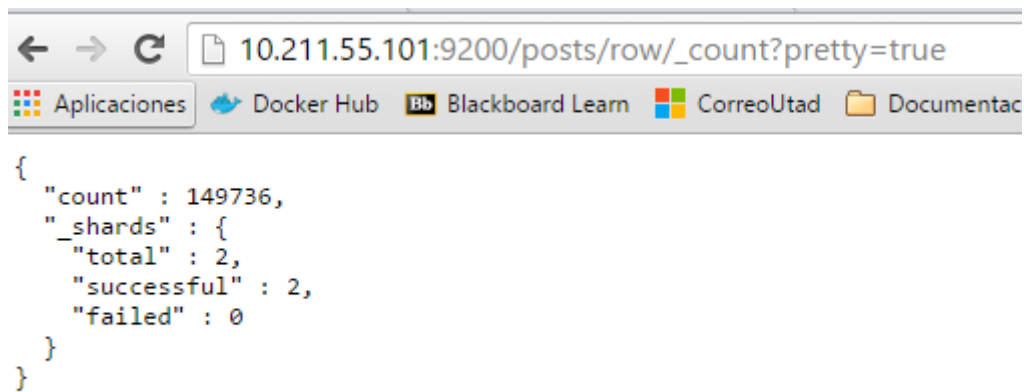
**Elasticsearch** <http://10.211.55.101:9200/> [Connect](#) **cluster\_elasticsearch** [Index](#)

Overview Indices Browser [Structured Query](#) [Any Request](#) [Refresh](#)

Browser

Searched 2 of 2 shards, 2458 hits, 0.728 seconds


	_index	_type	_id	_score	_id	postType	parentid	creationDate	score	ownerUserId	lastActivityDate	viewCount	body
posts	posts	row	306221	1	306221	2	306218	2015-12-30 21:21:03.673	13	17853	2015-12-30 21:23:44.423	0	Well, you're not going to get executed for it. The complaint in th
posts	posts	row	66031	1	66031	2	66027	2011-04-07 15:50:40.323	7	12436	2011-04-07 17:14:02.687	0	IANA, (I Am Not A Lawyer), but CPA 1998 states Encrypt any p
posts	posts	row	272209	1	272209	2	272206	2015-02-05 11:06:37.310	8	22685	2015-02-05 11:06:37.310	0	Yes Yes (It doesn't have to be dynamically linked. In detail: What
posts	posts	row	153454	1	153454	2	153448	2012-06-19 16:35:13.300	2	611	2013-09-21 12:02:38.003	0	Are you using Java 7? If so then it has native support for Zip arc
posts	posts	row	62736	1	62736	2	62727	2009-02-06 12:27:10.753	4	0	2009-02-06 12:27:10.753	0	Only if your team's core business is programming languages. I've
posts	posts	row	118952	1	118952	2	118940	2011-11-10 11:44:59.373	3	7167	2011-11-10 11:44:59.373	0	RESTful FizzBuzz Give them a computer, full access to the Intern
posts	posts	row	102917	1	102917	2	102889	2011-08-22 10:02:02.377	3	34985	2011-08-22 10:02:02.377	0	Look also at how much memory your program uses, not just its
posts	posts	row	225856	1	225856	2	225833	2014-01-28 23:55:21.133	1	116521	2014-01-28 23:55:21.133	0	An effective and efficient solution to this problem will most likely
posts	posts	row	10137	1	10137	2	9219	2010-10-07 09:51:56.123	0	0	2010-10-07 09:51:56.123	0	I will prefer those in the middle. Like the poster rightly point
posts	posts	row	114006	1	114006	2	114002	2011-10-12 20:43:11.433	24	14277	2011-10-12 20:43:11.433	0	Google is a tool, and you shouldn't feel bad about using it to you
posts	posts	row	173399	1	173399	2	173396	2012-11-02 18:26:15.817	9	31280	2012-11-02 20:44:23.063	0	Merriam-Webster definition of designate suggests: 1: to attack t
posts	posts	row	255066	1	255066	2	255006	2014-09-01 20:10:37.850	1	60023	2014-09-01 20:10:37.850	0	Given that: Both of the original STUN authentication schemes (v
posts	posts	row	139056	1	139056	2	139052	2012-03-10 04:16:16.010	16	15389	2012-03-10 04:16:16.010	0	Think of the List as an array and the Dictionary as a hash table.
posts	posts	row	296742	1	296742	2	296709	2015-09-08 17:41:07.327	6	60223	2015-09-08 17:41:07.327	0	The OS (unless it's a very low level embedded system or somet
posts	posts	row	57488	1	57488	2	56985	2011-03-13 07:40:25.910	0	8656	2011-03-13 07:40:25.910	0	Another deciding factor can be how the defect relates to your la
posts	posts	row	181666	1	181666	2	181661	2013-01-06 10:05:50.423	5	5099	2013-01-06 10:05:50.423	0	Sometimes, inheritance can be used for code re-use. Especially i
posts	posts	row	70629	1	70629	2	70628	2011-04-23 03:33:18.057	3	10097	2011-04-23 03:33:18.057	0	Because you can call a method and not assign its result.
posts	posts	row	147035	1	147035	2	146990	2012-05-03 07:30:48.617	1	36589	2012-05-03 07:30:48.617	0	In addition to what others have posted, I've always found drTV
posts	posts	row	10768	1	10768	2	3645	2010-10-09 20:57:03.113	1	4327	2010-10-09 20:57:03.113	0	'It depends' would be my short answer to this. It isn't necessari
posts	posts	row	161901	1	161901	2	161811	2012-08-22 18:48:41.720	3	11200	2012-08-22 18:48:41.720	0	It depends on why you want to pick the project back up. If you
posts	posts	row	36474	1	36474	2	36350	2011-01-13 17:41:43.043	0	1973	2011-01-13 17:41:43.043	0	I figured I could check that one off our Joel test so... I made a b
posts	posts	row	305765	1	305765	2	305761	2015-12-23 19:51:44.240	1	208679	2015-12-23 19:51:44.240	0	I believe what you're asking is based off of a person's personal
posts	posts	row	207744	1	207744	2	207710	2013-08-09 12:58:36.560	28	28175	2013-08-09 15:36:57.607	0	A comment that tells you why explains the reasoning behind the
posts	posts	row	150118	1	150118	2	150115	2012-05-25 01:52:14.893	3	44705	2012-05-25 01:52:14.893	0	Consider this example: let's say you want to provide a descripto
posts	posts	row	278628	1	278628	2	278624	2015-02-20 08:46:56.910	3	22685	2015-02-20 08:46:56.910	0	Use an existing one – don't bother reinventing the wheel when
posts	posts	row	120643	1	120643	2	120591	2011-11-19 14:23:11.903	1	2439	2011-11-19 14:23:11.903	0	Without going into specifics... but you might check out Semant
posts	posts	row	276966	1	276966	2	276950	2015-03-21 16:13:12.033	1	13181	2015-03-21 16:13:12.033	0	Before data binding was a popular method in UI libraries, the
posts	posts	row	288547	1	288547	2	288541	2015-07-02 15:37:35.243	1	28696	2015-07-02 19:57:33.957	0	Since a data mapper should have a single responsibility, it shou
posts	posts	row	241072	1	241072	2	241052	2014-05-25 18:46:31.260	1	4628	2014-05-25 18:46:31.260	0	First, you have to understand what the time complexity of a pro
posts	posts	row	36128	1	36128	2	36108	2011-01-12 20:32:20.010	0	5834	2011-01-12 20:32:20.010	0	How are they reinforced for attending? This is central to answer
posts	posts	row	94980	1	94980	2	94969	2011-07-21 20:02:46.407	0	25600	2011-07-21 20:02:46.407	0	Find an inductive logic game like Zendo and play it in your spare
posts	posts	row	78603	1	78603	2	78175	2011-05-23 20:00:50.610	1	3631	2011-05-23 20:06:39.490	0	Just dive in! Considering you already know how to program and
posts	posts	row	122216	1	122216	2	122209	2011-11-29 08:12:21.123	199	31090	2011-11-29 08:12:21.123	0	Object-oriented code is not by definition cleaner or conversely
posts	posts	row	260495	1	260495	1	0	2014-10-21 00:51:10.450	-1	63917	2014-10-21 15:37:30.753	181	What should be taken into account when choosing between des
posts	posts	row	152763	1	152763	1	0	2012-06-13 18:36:35.743	2	56726	2012-06-13 19:07:21.600	5482	I'm looking for a way to have a website remembering sensitive dat
posts	posts	row	96030	1	96030	1	0	2011-07-26 09:48:07.903	11	31418	2011-08-11 13:57:01.083	294	I'm sure you all have heard managers saying that "we need an i
posts	posts	row	234115	1	234115	2	234112	2014-03-29 00:01:57.077	1	98712	2014-03-29 01:25:53.387	0	The whole point of open source is that you don't have to reinve
posts	posts	row	199892	1	199892	2	199894	2013-05-30 00:01:42.697	10	9113	2013-06-06 06:11:50.180	0	So, from your more or less clarifying comments, I got it this wa



```
{
  "count" : 149736,
  "_shards" : {
    "total" : 2,
    "successful" : 2,
    "failed" : 0
  }
}
```

También se puede ejecutar en modo cluster:

```
$SPARK_HOME/bin/spark-submit --class Start \
  --master yarn-cluster \
  --deploy-mode cluster \
  --executor-cores 1 \
  --num-executors 2 \
  /home/vagrant/lectura.jar
```



Logged in as: dcraho

- Cluster
- About
- Nodes
- Applications
- NEW
- SAVING
- SUBMITTED
- ACCEPTED
- RUNNING
- FINISHED
- FAILED
- KILLED
- Scheduler
- Tools

**Application Overview**

User: vagrant

Name: Start

Application Type: SPARK

Application Tags:

State: FINISHED

FinalStatus: SUCCEEDED

Started: 10-Jul-2016 09:34:04

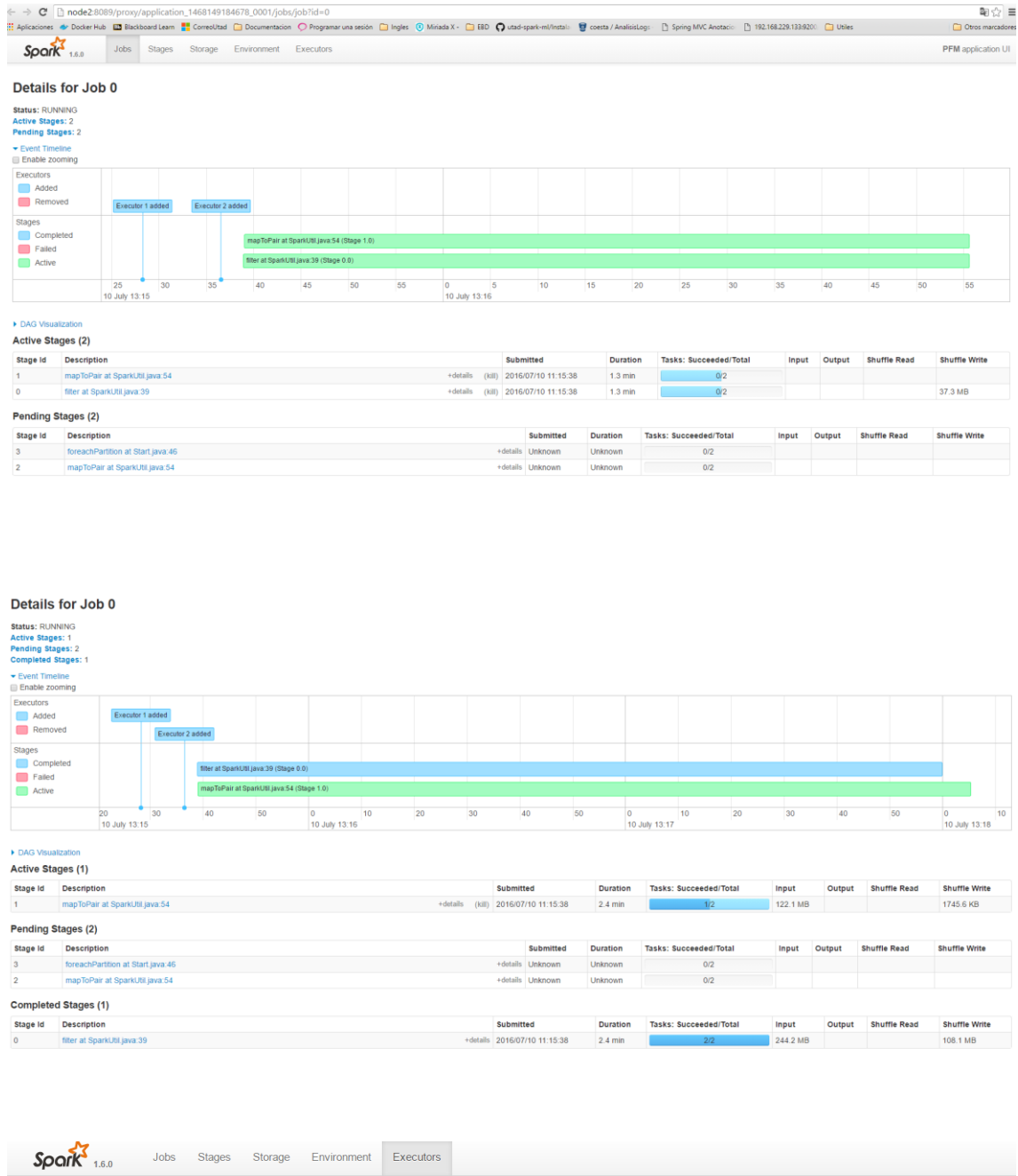
Elapsed: 3mins, 20sec

Tracking URL: History

Diagnostics: Shutdown hook called before final status was reported.

ApplicationMaster		Start Time	Node	Logs
Attempt Number				
1		10-Jul-2016 09:34:04	node3.8042	logs

Con lo que podríamos ver en las siguientes direcciones el proceso:



### Executors (3)

Memory: 0.0 B Used (1552.2 MB Total)  
Disk: 0.0 B Used

Executor ID	Address	RDD Blocks	Storage Memory	Disk Used	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time	Input	Shuffle Read	Shuffle Write	Logs	Thread Dump
1	node4:36863	0	0.0 B / 517.4 MB	0.0 B	1	0	1	2	1.6 m	122.1 MB	0.0 B	1745.6 KB	stdout stderr	Thread Dump
2	node3:42820	0	0.0 B / 517.4 MB	0.0 B	1	0	1	2	1.5 m	122.1 MB	0.0 B	55.1 MB	stdout stderr	Thread Dump
driver	10.211.55.104:56463	0	0.0 B / 517.4 MB	0.0 B	0	0	0	0	0 ms	0.0 B	0.0 B	0.0 B	stderr stdout	Thread Dump

(En este modo no hemos logrado que se conecte a elasticsearch con el cluster que se tiene, no tiene visibilidad sobre elasticsearch).

Como ya tenemos procesados los datos podemos realizar los análisis de los mismos desde kibana.

Primero se tiene que instalar.

```
cd /var/www/html
sudo wget https://download.elasticsearch.org/kibana/kibana/kibana-3.1.2.tar.gz
sudo tar -zxvf /tmp/kibana-3.1.2.tar.gz
sudo ln -s /var/www/html/kibana-3.1.2/ /var/www/html/kibana3
sudo mkdir /var/www/html/kibana3/log
```

```
[vagrant@node1 ~]$ cd /tmp
[vagrant@node1 tmp]$ sudo wget https://download.elasticsearch.org/kibana/kibana/kibana-3.1.2.tar.gz
--2016-07-08 22:07:53-- https://download.elasticsearch.org/kibana/kibana/kibana-3.1.2.tar.gz
Resolving download.elasticsearch.org... 23.23.137.104, 23.21.114.17, 23.21.156.42, ...
Connecting to download.elasticsearch.org|23.23.137.104|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1074306 (1.0M) [application/octet-stream]
Saving to: "kibana-3.1.2.tar.gz"

100%[=====>] 1,074,306 1.24M/s in 0.8s

2016-07-08 22:07:54 (1.24 MB/s) - "kibana-3.1.2.tar.gz" saved [1074306/1074306]
```

Ahora instalamos el servidor.

```
sudo yum -y install httpd
sudo /etc/init.d/httpd start
```

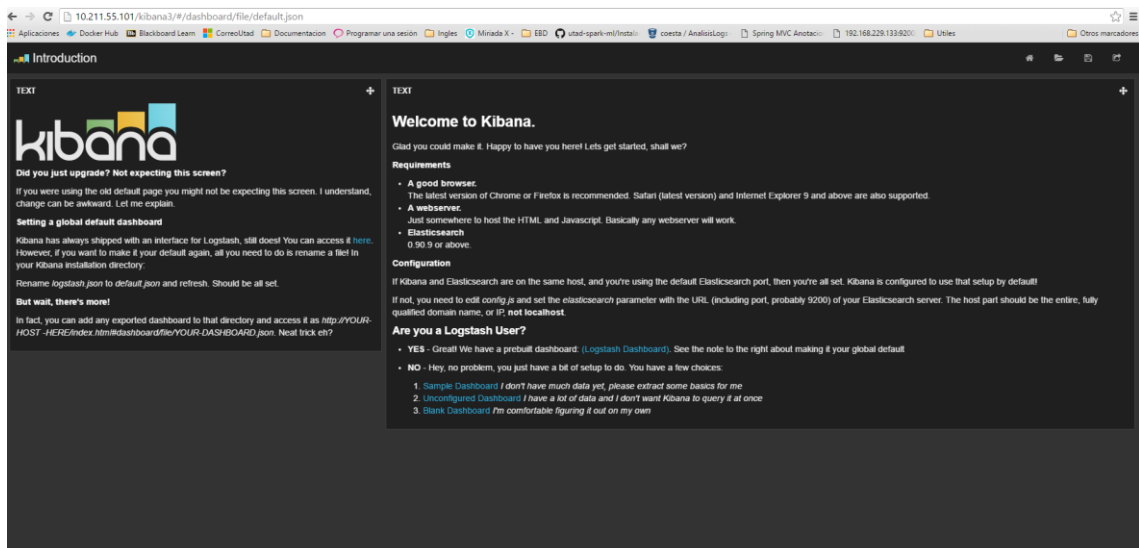
```

=====
Package                Arch             Version           Repository      Size
=====
Installing:
httpd                   x86_64           2.2.15-53.el6.centos    base            833
Installing for dependencies:
apr-util-ldap           x86_64           1.3.9-3.el6_0.1         base            15
httpd-tools             x86_64           2.2.15-53.el6.centos    base            78
mailcap                 noarch           2.1.31-2.el6            base            27
=====
Transaction Summary
=====
Install      4 Package(s)

Total download size: 954 k
Installed size: 3.2 M
Downloading Packages:
(1/4): apr-util-ldap-1.3.9-3.el6_0.1.x86_64.rpm | 15 kB  00:04
(2/4): httpd-2.2.15-53.el6.centos.x86_64.rpm | 833 kB 00:00
(3/4): httpd-tools-2.2.15-53.el6.centos.x86_64.rpm | 78 kB 00:00

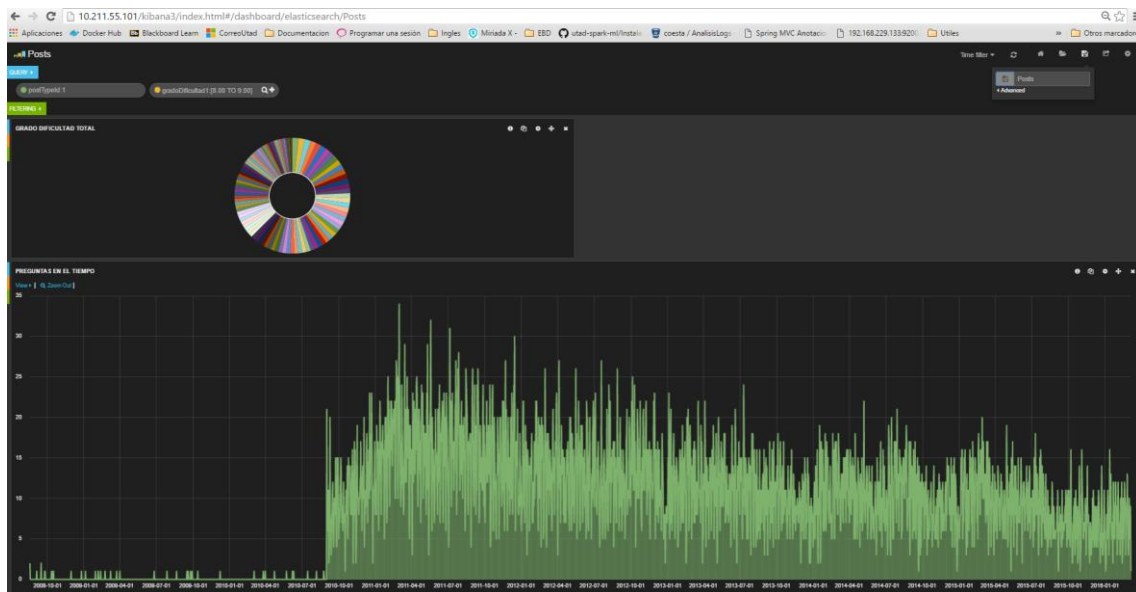
```

Accediendo a la url vemos la página web de entrada de kibana3



Y se pueden realizar multiples consultas:

En este caso en la primera gráfica se pueden ver los grados de dificultad de las respuestas (es un grafico de tipo donuts que muestra el campo gradoDificultad1 y este filtrado por postTypeId:1) y en el segundo gráfico se observa la cantidad de preguntas realizadas por fechas (gráfico de tipo histograma cuya x son las fechas de creación creaationDate y la y muestra el número de preguntas realizadas).



Ahora interactuando con la web contra elasticsearch se pueden obtener los exámenes según los requisitos informados por el usuario (este apartado se explica con más detalle en el manual de usuario).

The screenshot shows a web form titled "Crear examen" (Create exam) under the "Análisis Datos" (Data Analysis) section. The form includes input fields for "Dificultad" (Difficulty) ranging from 0.00 to 9.99, "Numero de preguntas" (Number of questions) set to 0, and a "Tags" field. There is a "Resposta a la consulta" (Response to the query) button and a "crear examen" (Create exam) button at the bottom.



## Manual de usuario

El usuario deberá subir a HDFS e fichero de Posts obtenido de stackoverflow, y lanzar el proyecto batch en el cluster.

Ahora ya puede consultar los datos desde el proyecto web, que deberá haber sido previamente desplegado en un servidor web.

La página principal es la de generación de exámenes, aunque podrá moverse entre las diferentes páginas a través del menú superior.

The screenshot shows a web application interface for creating an exam. At the top, there is a navigation bar with two tabs: 'Crear examen' (Create exam) and 'Análisis Datos' (Data Analysis). The 'Crear examen' tab is active. Below the navigation bar, there is a sidebar with two options: 'Preguntas por tag' (Questions by tag) and 'Grados de dificultad' (Degrees of difficulty). The main content area is titled 'Respuesta a la consulta' (Response to the query). It contains a form with the following fields: 'Dificultad: 0.00 a: 9.99' (Difficulty: 0.00 to 9.99), 'Numero de preguntas: 0' (Number of questions: 0), and 'Tags: ' (Tags: ). There is a trash icon and a magnifying glass icon next to the tags field. Below the form, there is a blue button labeled 'crear examen' (create exam).

## Creación de exámenes

El usuario dispone de la siguiente pantalla para la generación de exámenes:

Respuesta a la consulta


Dificultad: 0.00 a: 9.99 Numero de preguntas: 0 Tags:

En esta pantalla puede realizar cuatro tipos de acciones: Añadir un nuevo grupo de preguntas, eliminar un grupo de preguntas, consultar si existen datos para ese grupo de preguntas o generar el examen.



Los parámetros a informar son



- Grado de dificultad: Va de 0.00 a 9.99
- Numero de preguntas: El número de preguntas que se desean obtener con estos criterios, tiene que ser mayor que cero.
- Tags: Los tags por los que se desea filtrar, tienen que ir separados por comas. Este valor es opcional.



Si no se cumplen los requisitos se mostrará un error al consultar los datos o intentar generar un examen.


**AÑADIR PREGUNTAS:** Simplemente debe pulsar el botón , al pulsarlo aparecerá una nueva fila que se corresponde con un nuevo grupo de preguntas.

Resposta a la consulta


Dificultad: 0.01 a: 0.02 Numero de preguntas: 1 Tags: java  

Dificultad: 1.00 a: 1.02 Numero de preguntas: 5 Tags: sap  


Dificultad: 0.00 a: 9.99 Numero de preguntas: 0 Tags:  




crear examen

**BORRAR PREGUNTAS:** El botón de la papelera  permite realizar esta acción. (Si sólo tenemos un grupo de preguntas no permitirá borrar y mostrará un mensaje de error).

Resposta a la consulta

Dificultad: 0.01 a: 0.02 Numero de preguntas: 1 Tags: java  



No se puede eliminar si solo hay un elemento

crear examen

**CONSULTAR PREGUNTAS:** Boton de la lupa. Al pulsarlo lanza una consulta con los parámetros informados por el cliente en dicha fila contra elasticsearch y devuelve un mensaje indicando si se han encontrado preguntas con los parámetros deseados. Alidará también que los datos introducidos sean correctos.

Dificultad:  a:

Numero de preguntas:

Tags:

Dificultad:  a:

Numero de preguntas:

Tags:

Respuesta a la consulta

Existen posts con los criterios buscados en la fila 2

crear examen

GENERAR EXAMEN: Genera un pdf con los requisitos de todos los grupos de preguntas informados. Primero comprueba que existan resultados para todos los grupos de preguntas y si no informa al usuario que debe de cambiarlos.

My company is in the financial sector and it is using PHP as programming language. I am a PHP developer myself. I am leading a big project started from almost scratch. I can see how PHP is not the best candidate for building robust platforms. I want to convince my company to gradually switch to Java (which I have experience with). I was trying to find as many supporting arguments as possible. Can you help with this? So far I have found these: Most of the competitors are using Java (anyway not PHP) Most financial companies use Java rather than PHP On average, Java developers are better prepared (on average!) The compilation process catches a lot of problems before the software runs in production Strong typing makes everything more robust as contracts between interfaces is well defined Any other points I am missing? Thanks!

First you need to convince yourself. You have to exclude personal preferences from your reasoning. You may also try to change word Java to C++ (C#) in your list of arguments - most of them will stay valid but you should explain why C++ (C#) is not a good choice. If after all manipulations you find that the real reason is personal preference - you can set up your own start up or join competitors. UPDATE: This is marketing problem - you found number of benefits and disadvantages of switching to Java. You should choose those that are important for stakeholders. Better be honest because later you would be blamed if something goes wrong. That is why you should remove all subjective arguments. Personal preferences are important but they are not convincing. Think in terms of investment and return from it. It's not just features of programming languages - you should provide business plan with details of how you would do PHP to Java transition.

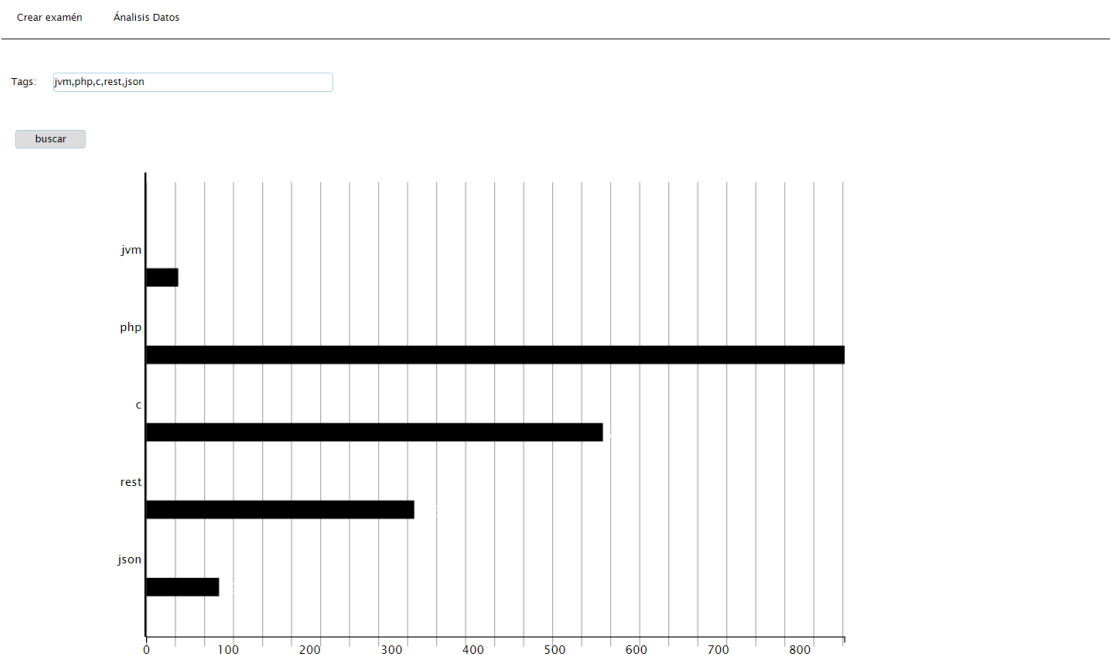
Java code is easier to unit test. This benefit might sound simple enough, but since your existing system is written in PHP, you may have other points to sell in order to make it sound appealing. I know that there have been improvements in the area of testing PHP code using frameworks such as PHPUnit, but I am skeptical about the ability to automate these tests (after reading up on the latest PHPUnit documentation). As far as I know, there is still no true way to utilize continuous integration as there are no actual builds, only subversion (or other VCS) tags to roll back to in the event of discovered defects. The last PHP project on which I worked (2 years ago), required the developers to come up with our own, project specific solution to run unit tests, which were expected to be manually run before requesting that code be deployed (meaning, plain text files moved) from our development server to our staging server. (Note use of the word 'expected', rather than 'required'.) Much room for human error. Hope that helps!

I don't see anything about the financial impact in your question/arguments. You HAVE TO talk about this as well when presenting your idea and your arguments. This is a business and money is the only thing managers and stakeholders care about. As long as it's profitable, managers and stakeholders don't care if you code in C#, Java, PHP, Javascript or even COBOL (may God have mercy on your soul if you deal with this one). This is obviously going to have a negative impact on all employees productivity and is going to cost the company a lot of money short-term and medium-term even though you mentioned a gradual switch to Java. But, if you really strongly think that this will result in a PROFIT on the long run, then this should be your primary argument when making this suggestion. Otherwise, I kind of doubt you'll succeed. Oh, and make sure you have some proof about WHY this would be profitable on the long run. Just saying that it will doesn't make it true. Good luck :).

## Visualizacion

### Número de preguntas por tags

En está pantalla se mostrará un campo donde el usuario puede introducir el numero de tags que desee separados por comas (si introduce caracteres extraños le mostrará un error). Cuando pulse el botón buscar se creará el gráfico con los valores recogidos de elasticsearch sobre esos tags.

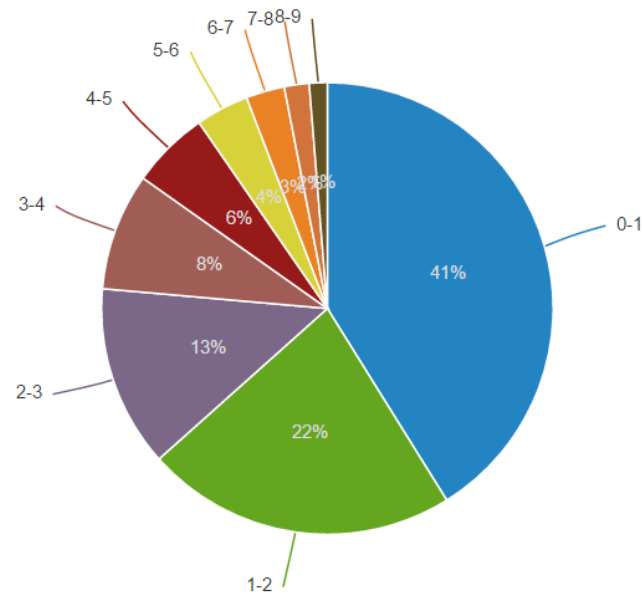


## Grados de dificultad percentage

El gráfico muestra el porcentaje en el que se dividen las preguntas entre los distintos grados de dificultad

[Crear examen](#)[Análisis Datos](#)

### Grados de dificultad



## REFERENCIAS

### *web*

<http://spark.apache.org/>

<http://docs.spring.io/spring-data/elasticsearch/docs/current/reference/html/>

<https://www.elastic.co>

[https://www.cloudera.com/documentation/enterprise/5-4-x/topics/cdh\\_ig\\_running\\_spark\\_on\\_yarn.html](https://www.cloudera.com/documentation/enterprise/5-4-x/topics/cdh_ig_running_spark_on_yarn.html)

<https://hadoop.apache.org/docs/r2.7.1/hadoop-yarn/hadoop-yarn-site/YARN.html>

<https://github.com/holdenk>

<https://d3js.org/>

<https://bl.ocks.org>

### *Libros*

Elasticsearch The definitive guide - O'REILLY

Learning Spark - O'REILLY