Sistema flexible de cerca en múltiples repositoris

TFG – Memòria

Projecte de final de grau

Autor: Ricardo Javier Soriano Chiva
Director: Xavier Burgués Illa
Quatrimestre de primavera 2019
Especialitat Enginyeria de Software

Resum

El projecte s'ha basat en realitzar un sistema de cerca que sigui flexible i que funcioni per múltiples repositoris. Els sistemes de cerca actuals no permeten adaptacions personalitzades de cara al proveïdor final del contingut ni permeten afegir dinàmicament nous continguts. S'ha realitzat un sistema que permet solucionar aquestes problemàtiques des de la indexació de nous continguts de forma dinàmica fins la explotació d'aquests continguts mitjançant un *front-end* web i un servei que permet a l'usuari final, adaptar les cerques a realitzar sobre el contingut ofert.

Resumen

El proyecto se ha basado en realizar un sistema de búsqueda que sea flexible y que funcione para múltiples repositorios. Los sistemas de búsqueda actuales no permiten adaptaciones personalizadas de cara al proveedor final del contenido ni permiten añadir dinámicamente nuevos contenidos. Se ha realizado un sistema que permite solucionar estas problemáticas desde la indexación de nuevos contenidos de forma dinámica hasta la explotación de estos contenidos mediante un *front-end web* y un servicio que permite al usuario final, adaptar las búsquedas a realizar sobre el contenido ofrecido.

Abstract

The project has been based on making a search system that is flexible and that works for multiple repositories. Current search systems do not allow personalized adaptations for the final supplier of the content neither allow adding new content dynamically. A system has been created to solve these problems from the indexing of new contents dynamically to the exploitation of these contents through a *front-end* and a web service that allows the end user to adapt the searches to perform about the offered content.

Índex de continguts

Índex de figures

1. Introducció

1.1 Formulació del problema

Sovint ens trobem que tant en empreses com entitats tals que la universitat o les biblioteques públiques, disposes d'un gran volum de documents repartits en diferents repositoris tals que servidors, intranets, etc. Cosa que dificulta en gran mesura l'obtenció d'un document concret o amb un determinat contingut requerint als usuaris un gran esforç i una inversió de temps força gran en alguns casos.

D'altra banda, també suposa un gran problema el mantenir actualitzat un sistema de documents de gran mesura, sobretot si s'incorporen nous continguts de forma sovint, com pugui ser el registre d'una empresa de periodisme, que hauria d'incorporar les noves noticies o diaris a la seva base de dades principal cada dia.

Actualment existeixen softwares que et permeten *crawlejar* diversos tipus de repositoris per obtenir els continguts dels diferents documents. Existeixen també moltes eines que permeten fer cerques sobre continguts indexats en bases de dades o altres índexs de continguts, però tots aquests softwares estan enfocats a un tipus molt concret de repositoris. Per tant, existeix la problemàtica de centralitzar i mantenir continguts de diferents repositoris en un mateix Software.

Així doncs, en aquest projecte ens centrarem en centralitzar un buscador que permeti buscar en múltiples repositoris i que a més permeti la flexibilitat de filtratge de les cerques en funció de l'àmbit en que vagi destinat gràcies a l'explotació dels continguts mitjançant un servei web.

1.2 Contextualització

Sovint ens trobem amb la problemàtica de buscar documents que poden residir en múltiples repositoris i que buscar-los en les múltiples plataformes suposa un sobreesforç important. Els cercadors més actuals s'han enfocat més a la red (internet) que en el dia a dia de moltes entitats (biblioteques, diaris, asseguradores, editorials, bancs...) tot i disposar d'un gran nombre de documents.

En aquest projecte es tractarà amb múltiples repositoris; problemàtica que suposa la dispersió de documents. Hi ha diverses solucions per tractar amb el problema de centralitzar els continguts dels documents en un únic índex sobre el que poder realitzar cerques. D'altra banda també s'abordarà el problema de gestionar cerques sobre diferents criteris en funció de l'usuari final.

Les solucions en aquestes problemàtiques es basen en tecnologies actuals com ara *Elasticsearch*, *ManifoldCF*, i serveis web basats en *Java* i *html5* per el frontal que ens permetrà visualitzar els resultats de les cerques.

1.3 Actors implicats (*Stakeholders*)

A continuació s'especifiquen tots els actors i els seus rols dins del projecte que desenvoluparem.

1.3.1 Programador, dissenyador i tester

Degut a que el projecte està encarregat a una sola persona, seré jo mateix qui realitzarà les tasques de dissenyador (o arquitecte), programador i tester.

1.3.2 Director del projecte

Xavier Burgués Illa és el tutor d'aquest projecte. Ell s'encarregarà d'assumir el rol de director i, per tant de supervisar que s'està portant a terme la planificació prèvia del projecte per tal de seguir el calendari establert i per tal de complir els objectius establers. A més, com a director també pot ajudar i guiar el projecte per tal de complir les fites definides i el projecte final.

1.3.3 Empreses o entitats públiques

Les o empreses o entitats públiques (com ara biblioteques) que disposen d'un gran volum de documents poden estar interessades en les conclusions d'aquest projecte, ja que els permetria donar un millor servei de cara als usuaris finals, ja siguin empleats de l'empresa o persones que vulguin disposar d'un contingut d'un llibre sense saber com trobar-lo.

1.3.4 Usuaris

Seran els principals beneficiaris del sistema. Disposaran d'un sistema que els permetrà buscar continguts o documents mitjançant un sistema centralitzat sobre únicament la documentació de la que disposi l'empresa o entitat final que hagi aportat els documents.
Existiran dos tipus d'usuaris. Els administradors del sistema, és a dir qui tindrà muntat tot el sistema sobre el qual es permetrà buscar i l'usuari final que serà qui podrà accedir a aquests continguts.

1.4 Objectiu

Aquest projecte es basa en el muntatge d'un sistema de cerca flexible sobre múltiples repositoris mitjançant un *Crawler* per obtenir contingut, una base de dades i motor de cerca a on indexar-lo, un servei web per explotar el contingut i un *front-end* web per visualitzar les cerques.

El problema principal és que sovint, les empreses o entitats disposen de molta documentació totalment dispersa en múltiples repositoris, servidors o webs (en cas de intranets) dificultant força la cerca d'alguna documentació concreta.

Actualment existeix una tecnologia que permet extreure contingut de diferents tipus de repositoris, tals com sistemes de fitxers, webs... Mitjançant aquesta tecnologia es pretén obtenir la flexibilitat sobre múltiples repositoris que suposa un dels problemes actuals.

Així doncs en aquest projecte ens centrarem en solucionar aquest objectiu que ajudarà a la millora de gestió del sector documental a les diferents entitats.

2. Estat de l'art

Aquesta part té com objectiu informar sobre l'estat de l'art de les diferents parts del projecte. A continuació, és llisten les dues parts de les que està format i una introducció als sistemes actuals.

2.1 Sistemes actuals de cerca

Actualment el sistema de cerca més usat entre els usuaris, es el cercador de *Google* indiscutiblement. De cara a empreses privades i d'altres entitats, *Google* oferia un sistema intern de cerca personalitzada anomenat *Google Search Appliance* (*GSA*), fins desembre del 2018 que *Google* va deixar de donar servei a aquest sistema. Existeixen diversos software que permeten realitzar cerques, però tots ells enfocats o especialitzats en un sol ús; Ja sigui web (com els softwares: *Doofinder*, *Atomz*, *PicoSearch*...) o d'altres tipus de repositoris com sistemes de fitxers (com ara els softwares: *ScanFS*, el propi cercador de *Windows*,...). Així doncs no existeix un software que permeti realitzar cerques sobre múltiples repositoris alhora i estigui a l'abast de tothom.

D'altra banda, els softwares existents gairebé no permeten modularització ni personalització de cara a l'enfocament que se'ls vulgui donar. Per exemple, en tots els casos mencionats anteriorment, cap dels softwares de cerca permet filtrar continguts específics a unes "regles de negoci". Tots ells son cercadors genèrics i permeten poca personalització. Per tant, si volguéssim afegir regles concretes com ara "cerques que continguin sinònims sobre la cerca realitzada" no ho podríem realitzar d'aquesta manera. Un altre exemple seria en un cas concret sobre una universitat, si tota la intranet i documents de caràcter educatiu estesin a l'abast de la cerca, no es podrien realitzar regles de caràcter concret com ara "materials o webs internes d'una assignatura o departament".

Durant l'apartat "Formulació del problema" i "Actors implicats" havíem parlat d'exemples d'entitats com ara les biblioteques. A dia d'avui les biblioteques poden disposar de softwares que permeten guardar títols dels documents, permeten afegir manualment una breu descripció, permeten també mantenir un registre de si estan o no prestats, escriure notes, i etiquetar o catalogar els diferents document. Però no permeten buscar sobre els propis continguts dels documents. Alguns exemples d'aquests softwares per biblioteques serien: *Libib, LibraryThing, Bibliotecas XL, Book Collector, BookDB2…*

Així doncs, en aquest projecte ens centrarem en dos parts per enfocar aquesta problemàtica. Per un costat, la ingesta de continguts i per altra l'explotació d'aquests.

2.2 Ingesta i emmagatzematge de continguts sobre els que realitzar les cerques

Per poder buscar continguts, aquests primer s'han de recol·lectar i emmagatzemar a algun lloc. Per portar a terme aquesta tasca, existeixen diversos softwares de recol·lecció (*crawlers*) i diverses maneres d'emmagatzemar documents. Habitualment per emmagatzemar documents s'utilitzen bases de dades relacionals. Però les bases

de dades relacionals tenen limitacions i, sobretot no tenen facilitat per incorporar noves dades a les tables (actualitzacions d'índexs, modificacions de tables, claus externes entre tables, etc.).

Existeixen diversos softwares utilitzats per l'emmagatzematge de documents. Com a bases de dades relacionals, les més usades a dia d'avui son *MySql*, *PostgreSql*, *Oracle*, *DB2*.

De cara a les bases de dades no relacionals, les més usades a dia d'avui son *MongoDB*, *Couchbase*, *Elasticsearch*, *Solr*.

Sorl i *Elasticsearch* són a més a més motors de cerca força potents basats en *Lucene*.

Tant *Solr* com *Elasticsearch* són *opensource*. Motiu pel qual s'han acabat utilitzant en múltiples projectes d'empreses privades per guardar dades i explotar-les com si fos una simple base de dades. A més de ser *opensource*, tant *Solr* com *Elasticsearch* son sistemes distribuïts, cosa que els fa fàcilment escalables i adaptables.

Elasticsearch ofereix el seu servei mitjançant crides *HTTP* en format *JSON*, i els seus resultats també els retorna en format *JSON*. Facilitant integració amb serveis *REST*.

De cara als recol·lectors de documents (o crawlers), els principals recol·lectors per enllaços web son *GoogleBot* (de Google), *Slurp Bot* (de *Yahoo*), *DuckDuckBot*, *Alexa Crawler* (*d'Amazon*). D'altra banda els més usats per sistemes de fitxers son *Diskover, ManifoldCF,...*

GoogleBot és el *crawler* que utilitza Google per "rastrejar" pàgines web. A part d'indexar els *HTML* també és capaç d'extreure la informació de fitxers *PDF*, *PS*, *XLS*, *DOC* y d'altres.

Diskover és *opensource.* Únicament permet crawlejar sistemes de fitxers i disposa de compatibilitat amb Elasticsearch.

ManifoldCF també és *opensource.* Pertany a *Apache* i disposa d'una amplitud de connectors a diferents tipus de repositoris: *Alfresco, CMIS, DropBox, Email, File System, Google Drive, HDFS, Windows Shares, JDBC, Jira, Web, Solr, Elasticsearch, Documentum, SharePoint, FileNet… ManifoldCF* al igual que GoogleBot també és capaç d'extreure els continguts de diferents tipus de fitxers mitjançant el seu component incorporat *Tika* (també *d'Apache*).

2.3 Sistema de cerca sobre els continguts

Aquesta part tracta sobre l'explotació dels continguts que s'hagin indexat i les regles a aplicar sobre les cerques. Actualment els *softwares* de cerca que ofereixen empreses privades, no permeten tal nivell de personalització; però d'altra banda la idea de fer un frontal *web* i un servei *Java* per tractar aquestes problemàtiques tampoc suposa una novetat.

La majoria de tecnologies actuals opten per l'arquitectura en forma de servei per evitar problemes a l'hora d'enfocar-los a multi plataformes com ara webs, aplicacions mòbils, tablets i d'altres frontals; al desvincular-los de la plataforma estalvia problemes i permet més canviabilitat. D'altra banda, un portal web permet accedir-hi des de

qualsevol lloc físic sempre que es disposi de connexió a internet. A més un frontal web sempre pot limitar el seu accés a una intranet si es vol limitar el seu accés a un grup més reduït d'usuaris com puguin ser alumnes d'una universitat, empleats d'una empresa privada, usuaris d'una biblioteca concreta. Ja que un frontal web s'adreça a un servidor que pot ser privat pertanyent a l'entitat final que vulgui oferir aquest cercador.

A l'apartat anterior hem parlat d'alguns sistemes d'emmagatzematge que suposen també un motor de cerca força potent. *Elasticsearch* i *Solr*.

El sistema d'emmagatzematge i motor de cerca *Elasticsearch*, no és una tecnologia novedosa. *Elasticsearch* va néixer al 2010 amb la seva primera versió. A dia d'avui ha anat evolucionant fins la versió 6.6. *Elasticsearch* guarda les dades introduïdes de manera similar a una base de dades no relacional. El seu motor de cerca està basat en *Lucene* (*d'Apache*); l'arquitectura de la qual està basada en el concepte de "document".

Solr també està basat en *lucene*. A nivell de prestacions és totalment comparable a *Elasticsearch*. *Solar* porta al mercat des del 2004.

2.4 Conclusions

Així dons, podem concloure que actualment no hi ha una eina que solucioni totes les problemàtiques abordades en aquest projecte, però sí hi ha diferents eines que solucionen les diferents problemàtiques per separat.

Per tant, no podem dir que sigui un projecte totalment innovador, però si exclusiu (sense competència directe).

Podem concloure també que si les diferents eines per separat poden solucionar els diferents problemes, al muntar un projecte que les utilitzi en comú, podrem aconseguir l'objectiu establert.

3. Definició de l'Abast

Volem crear un sistema flexible de cerca sobre múltiples repositoris.

Per realitzar aquest sistema, primer necessitarem muntar una estructura que permeti obtenir, guardar i gestionar els continguts sobre els que es vol poder realitzar cerques. El model que s'ha pensat per portar-ho a terme es basa en el motor de cerca *Elasticsearch*. *Elasticsearch* ens permetrà guardar documents i a l'hora serà el pilar central del cercador de continguts.

La primera part del projecte doncs es basarà en recol·lectar i indexar els documents al motor de cerca *Elasticsearch*. Per portar a terme aquesta tasca, s'ha decidit utilitzar el *crawler ManifoldCF* per poder recol·lectar els documents i enviar-los al *Elasticsearch*, ja que *ManifoldCF* disposa de connectors d'entrada cap a múltiples repositoris i, sobretot, disposa de connectors de sortida compatibles amb *Elasticsearch*.

Per la segona part del projecte s'ha de desenvolupar un sistema que permeti explotar les dades recopilades al *Elasticsearch*. Per fer això s'ha plantejat utilitzar un servei desenvolupat en *Java*. La tecnologia basada en serveis, permet la desvinculació del *hardware* que el vulgui utilitzar per accedir al servei (tablets, mòbils, pcs, ...).

A més a més, el servei web hauria de permetre incorporar regles de negoci per poder adaptar les cerques a realitzar en funció de l'Entitat final que vulgui oferir el sistema de cerca.

Finalment s'hauria de realitzar un *front-end* web que ens permeti visualitzar els continguts buscats d'una manera visualment còmoda per als usuaris finals.

En aquest cas s'ha plantejat desenvolupar un frontal web que permeti accedir al servei desenvolupat en *Java* i permeti mostrar les dades d'una forma "amigable" a un usuari final.

Es realitzarà un prototip que funcioni raonablement bé per 3 usuaris concurrents. En el prototip que es desenvoluparà, només es tindrà en compte que el *front-end* es visualitzi correctament des d'una torre i un portàtil. La resta de dispositius comentats estaran fora de l'abast (mòbils, tablets...).

Un cop definit el sistema i tenint en compte les eines utilitzades, així com el temps emprat per poder realitzar el sistema, ens podem trobar amb tres grans limitacions.

3.1 Limitacions de ManifoldCF

ManifoldCF és un producte en constant desenvolupament. Actualment disposa de 21 connectors a repositoris. Això significa que encara hi ha molts tipus de repositoris als que no pot arribar. D'altra banda el component integrat *Tika*, encarregat de extreure continguts dels documents *crawlejats*, encara no és capaç de reconèixer tots els formats de documents/fitxers existents actualment. Per tant, es mirarà d'arribar a tants tipus de fitxers i repositoris com es pugui, tenint en compte la limitació temporal (descrita a sota) i els límits que ens ofereix aquesta eina.

3.2 Limitacions de Calendari

Un dels problemes que ens trobarem és el curt termini per a la realització del projecte, 4 mesos, ens força a planificar un calendari molt ajustat, havent de realitzar reunions quinzenals amb el director del projecte per tal de encaminar-ho i portar-lo a terme de la millor manera possible.

4. Requisits

Els requisits són aquelles condicions que ha de satisfer el programa que desenvoluparem. Els hem dividit depenent de si són de qualitat, rendiment, funcionals o de seguretat, manteniment i documentació. A continuació es presenten aquests requisits juntament amb la seva descripció i els criteris de satisfacció (com podrem comprovar que el criteri es compleix un cop estigui acabat el projecte). Aquests requisits són el resultat de les diferents fases de definició del problema, explicat a l'apartat de Metodologia.

4.1 Requisits de qualitat

Els requisits de qualitat són aquelles condicions que requereixen que l'eina o la seva execució tingui unes certes propietats de qualitat.

Disseny simple, intuïtiu i fàcil d'utilitzar

Descripció: Els usuaris han de poder realitzar les funcionalitats principals sense haver de seguir massa passos. L'ús de l'eina ha de ser intuïtiva, no ha de ser necessari molt de temps d'aprenentatge. Tampoc ha de ser necessària una formació prèvia per a l'ús de l'eina.

Criteri de satisfacció: Per manca de temps no podrem seguir els criteris d'usabilitat de *Jakob Nielsen*, per el que se li sol·licitarà a un usuari i al cap de projecte que facin servir l'eina i que facin un parell de cerques sense explicar-los com han de fer-ho.

El llenguatge escrit ha de ser comprensible, clar i concís

Descripció: Com l'eina serà utilitzada per usuaris de diverses edats i titulacions acadèmiques, els textos de tota la part web desenvolupada per nosaltres i les parts desenvolupades per terceres persones (*ManifoldCF* i *Elasticsearch*) han de tenir un llenguatge comprensiu i concret.

Criteri de satisfacció: Es sol·licitarà a un usuari que validi els diversos textos per comprovar si son comprensibles, clars i concisos.

Sense faltes d'ortografia

Descripció: Tot i que no hi hagi massa text a la part desenvolupada per nosaltres i els diferents softwares utilitzats (*ManifoldCF* i *Elasticsearch*) son de terceres persones, tot text nostre no pot contenir faltes ortogràfiques.

Criteri de satisfacció: Es revisaran amb un corrector tots els textos tant de les eines com de la web.

La web s'ha de veure correctament des de qualsevol equip

Descripció: El disseny s'ha de veure de forma correcta amb la configuració per defecte dels monitors, tant els dels portàtils com torres.

Criteri de satisfacció: És provarà el frontal web en una torre i en un portàtil que no siguin amb els que s'hagi desenvolupat.

El disseny s'ha de veure correctament des del navegador Google Chrome

Descripció: El navegador recomanat per utilitzar el nostre frontal web, és *Google Chrome*, ja que és el més utilitzat pels usuaris. La nostra aplicació ha de ser accessible almenys des d'ell.

Criteri de satisfacció: Abans de posar l'eina a l'abast es provarà la web a un entorn de desenvolupament. A l'entorn de desenvolupament estarà instal·lada la versió més actual del moment del navegador *Google Chrome*[1] i es solucionaran possibles problemes que sorgeixin.

4.2 Requisits de rendiment

A causa de la naturalesa del projecte necessitem una eina amb un rendiment alt, no tant per la recol·lecció e indexació de documents sinó més aviat en la velocitat de resposta de les cerques realitzades a través del frontal web; a causa que s'espera que l'eina escali considerablement en el futur. Per aquest motiu s'ha separat els requisits de rendiment dels de qualitat, per donar-los un èmfasi més gran.

L'eina donarà resposta ràpidament a les peticions dels usuaris

Descripció: Per tal que els diferents usuaris es sentin còmodes al frontal web a l'hora de buscar documents i continguts, aquest ha de respondre ràpidament a les seves peticions. Això implica que s'ha de minimitzar el temps de resposta de les diferents cerques continguts i de l'aplicació web.

Criteri de satisfacció: *Elasticsearch* permet calcular el temps de resposta de cada una de les peticions de cerca que es realitzen sobre la plataforma; a més amb el servei *Java* també es pot calcular el temps total de processat de les dades i fins abans de retornar les respostes de cerca al frontal web. Per tant, podrem comprovar el temps que triguen les peticions en processar-se i realitzar-se. Es faran proves en un entorn acotat amb tres usuaris concurrents per comprovar que el temps de resposta de la web no excedeixi els 0,5 segons en donar resposta. En aquest prototip no s'establirà un sistema de monitorització com seria recomanat en un sistema real.

Ha de poder resoldre peticions simultànies

Descripció: Tot i que es tracti d'un prototip hem d'assegurar-nos que puguin haver almenys tres usuaris simultàniament treballant amb l'eina.

[1] https://www.muycomputer.com/2018/05/03/chrome-dominando-navegadores

Criteri de satisfacció: Es realitzaran proves d'estrès amb tres usuaris i es controlarà que els temps de resposta no excedeixin dels requerits.

4.3 Requisits funcionals

Els requisits funcionals són aquelles condicions que requereixen que l'eina compleixi una determinada funcionalitat. Per a aquests requisits no es defineixen uns criteris de satisfacció ja que la descripció és suficientment detallada i es pot verificar fàcilment si hi ha aquesta funcionalitat o no.

Crawlejar documents

Descripció: Es requereix que l'eina permeti rastrejar documents de múltiples repositoris.

Indexar documents

Descripció: Es requereix que l'eina permeti guardar els documents rastrejats.

Buscar documents

Descripció: Es requereix que l'eina permeti buscar sobre els continguts dels documents indexats. També es requereix que es puguin aplicar regles de negoci sobre les cerques.

Visualitzar documents

Descripció: Es requereix que l'eina permeti visualitzar els documents sobre els que s'ha realitzat la cerca.

4.4 Requisits de seguretat, mantenibilitat i documentació

En aquest punt es tractaran els requisits del projecte relacionats amb la seguretat, el manteniment o la documentació.

Seguretat en l'eina

Descripció: Es requereix algun sistema que permeti recuperar el codi de l'eina així com els softwares utilitzats (*ManifoldCF* i *Elasticsearch*) per poder recuperar-la en cas d'avaria o atac.

Criteri de satisfacció: Al realitzar el desenvolupament del servei web i del front-end, amb un repositori *GIT* ja comptem amb un historial dels canvis que es vagin fent.

Accés al contingut

Donat que aquest treball està pensat en un entorn de consulta oberta i no en un entorn de treball intern a una organització, no cal posar proteccions d'accés al contingut.

Documentació per futurs desenvolupadors de l'eina

Descripció: Es requereix un document que detalli el funcionament de l'eina. També s'afegirà una guia d'instal·lació i configuració de l'entorn. Aquest informe anirà destinat als futurs desenvolupadors de l'eina.

Criteri de satisfacció: Queda cobert amb aquest informe.

5. Metodologia i rigor

En aquest apartat s'especifica quina serà la metodologia utilitzada al llarg de tot el projecte.

Per poder realitzar el projecte i desenvolupar-lo segons el calendari establert, s'han realitzat reunions setmanals per poder solucionar tots els possibles dubtes i problemes que han anat sorgint i poder encaminar el projecte de la manera preestablerta.

5.1 Mètodes de treball

Per a desenvolupar el sistema es s'utilitzaran diferents eines de treball.

Per un costat es prepararà la base des d'on es faran les cerques. Aquesta base serà *Elasticsearch*. *Elasticsearch* té com a dependència el *crawler* de continguts. Sense aquest *crawler*, *l'Elasticsearch* estaria totalment buit. El *crawler* serà portat per *ManifoldCF*. *ManifoldCF* s'encarregarà de connectar als diferents repositoris amb els connectors propis que porta incorporats i serà el que obtindrà els documents. Mitjançant *tika*, incorporat també a dins de *ManifoldCF* s'obtindran els continguts dels documents que seran indexats a *Elasticsearch* a través dels connectors de sortida de *ManifoldCF.*

D'altra banda es prepararà el cercador. Aquest cercador s'encarregarà d'accedir a *Elasticsearch* i obtenir el contingut indexat. El cercador consta de dues parts. La primera serà desenvolupar el servei que sigui capaç de connectar amb *Elasticsearch*. Aquest servei serà desenvolupat amb *Java*. Aquest servei té dependència amb *Elasticsearch* ja que requereix connectar-se per obtenir les dades que es volen buscar. La segona part consta d'un frontal web que accedeixi al servei Java mitjançant una interfície "amigable". Aquest *front-end* té dependència amb el servei *Java* ja que si el servei no està aixecat i disponible, aquest no pot accedir a ell. El *front-end* es desenvoluparà amb *JSP* i *CSS*.

Finalment es prepararà el sistema per incloure "regles de negoci". Aquestes regles de negoci aniran al *front-end* i es passaran al servei *Java* per poder "personalitzar les cerques".

Tot aquest procés serà portat a terme utilitzant una metodologia de treball amb cicles curts de dues setmanes, d'aquesta manera es garantirà una visió mes real de l'estat en que es troba el projecte i si continua o no dins del calendari marcat a l'inici del projecte.

5.2 Eines de seguiment

Mitjançant l'eina *Git* i el seu control de versions podrem anar guardant totes les modificacions realitzades sobre el servei web i el frontal, per així poder portar un control de l'evolució del projecte durant cada reunió quinzenal. De cara a *Elasticsearch*, només es requerirà un control de versions de les *templates,* el qual també es pujarà a git per mantenir un control de versions*. ManifoldCF* d'altra banda, no requereix ser modificat durant el projecte donat que és una eina de tercers (*Apache*) i ens serveix tal qual ve proporcionada d'origen.

5.3 Mètode de validació

Per tal de poder validar l'evolució del projecte es realitzaran proves amb dos pc's. El primer servirà com a servidor; i es comprovarà que s'obtinguin i indexin els continguts. El segon pc servirà com a client. Des del segon pc es comprovarà que es pot accedir al frontal web i que des del frontal es poden realitzar cerques sobre el contingut a través del servei desenvolupat i desplegat al primer pc.

6. Justificació de les tecnologies

Durant l'estudi de l'*Estat e l'Art* vam veure que hi havia diferents tecnologies que ens permetien portar a terme el projecte. En aquest punt justificarem les tecnologies que finalment s'utilitzaran per realitzar el prototip.

Crawler

Degut a que *GoogleBot* és el *crawler* més usat per "rastrejar" web, el vam estudiar com a primera opció; però el vam descartar degut a que només servia per webs.

Diskover, un altre dels *crawlers* estudiats durant l'Estat de l'Art, resultava que era només per sistemes de fitxers, per tant també el vam descartar.

Llavors va ser fàcil decidir-se per *ManifoldCF* que era l'altre eina proposada durant l'Estat de l'Art, ja que aquest disposa d'una gran amplitud de connectors a diferents tipus de repositoris i a més disposa d'un extractor de text (també el té *GoogleBot*).

Indexador/Motor de cerca

A diferència del *crawler*, amb el motor de cerca si vam tenir dificultats per escollir-ne un d'entre tots els que havíem vist durant l'estudi de l'Estat de l'Art.

Tenint en compte que el motor de cerca no només havia d'emmagatzemar documents amb diverses metadades, sinó que també ens havia de facilitar la cerca sobre aquests, es va optar per descartar les bases de dades relacionals, ja que aquestes, quan hi ha un gran volum de documents, incrementa força el temps de cerca sobre aquests. A part que tampoc sabíem en primera instància quines metadades acabaríem extraient. Per tant tampoc podríem definir una tabla tancada.

Amb aquesta premissa vam descartar *MySql*, *PostgreSql*, *Oracle*, *DB2*... ja que totes aquestes son bases de dades relacionals.

Així dons la decisió més difícil estava entre *Elasticsearch* i *Solr*. Ambdós basats en *Lucene* i igual de potents en quan a cercadors. Fent recerca per internet vam veure que *Elasticsearch* era el més utilitzat com a motor de cerca [2] i, a més a més, *ManifoldCF* disposa d'un connector a *Elasticsearch* ja inclòs. Per aquests motius finalment, vam decantar-nos per aquest.

Servei de cerca

[2] https://www.hiberus.com/crecemos-contigo/nosql-y-los-motores-de-busqueda-apache-solr-vs-elasticsearch/

De cara a la implementació del servei, es va decidir realitzar-lo amb *Java*. Això és deu al coneixement previ que teníem d'aquest llenguatge de programació degut per un costat al les classes cursades a la *UPC* i d'altra a l'experiència laboral a on hem incrementat el coneixement d'aquest llenguatge.

Front-end

Per realitzar el *front-end* es va decidir muntar una pàgina web amb *Html*. Posat que el servei serà *Java*, el *front-end* també portarà codi *JSP* per poder realitzar l'acoplament de manera més fàcil i eficient.

7. Planificació

En aquesta part parlarem sobre la distribució temporal del projecte, descrivint la duració de cada part d'aquest.
La duració aproximada del projecte és de 3 mesos i mig (115 dies). Com ja es va comentar a la definició del abast, durant la duració del treball poden sorgir diferents complicacions que afectin a la planificació inicial.

7.1 Pla inicial de projecte

A continuació es mostra una taula amb les fases i la duració de cada fase del projecte.

Nombre	Duración	Inicio	Fin	Predecesoras	Recursos
⊟ **Gestió inicial del projecte**	40días	28/01/2019	22/03/2019		Cap de projecte
Contextualització	5días	28/01/2019	01/02/2019		Cap de projecte
Gestió econòmica i sostenibilitat	5días	04/02/2019	08/02/2019		Cap de projecte
Estat de l'art	15días	11/02/2019	01/03/2019		Cap de projecte
definició de l'abast	5días	25/02/2019	01/03/2019		Cap de projecte
Planificació temporal	5días	04/03/2019	08/03/2019		Cap de projecte
Requisits	10días	11/03/2019	22/03/2019		Cap de projecte
Metodologia i rigor	5días	18/03/2019	22/03/2019		Cap de projecte
⊟ **ManifoldCF**	10días	25/03/2019	05/04/2019	1	
Preparació de l'entorn	2días	25/03/2019	26/03/2019		Programador
Instalació i configuració	2días	25/03/2019	26/03/2019		Programador
Connectors entrada/sortida	2días	28/03/2019	29/03/2019	10,11	Programador
Jobs	5días	28/03/2019	03/04/2019	10,11	Programador
Proves unitàries	5días	01/04/2019	05/04/2019		Tester
⊟ **Elasticsearch**	10días	08/04/2019	19/04/2019	9	
Preparació de l'entorn	2días	08/04/2019	09/04/2019		Programador
Instalació i configuració	2días	08/04/2019	09/04/2019	13	Programador
Definició d'índex i templates	2días	11/04/2019	12/04/2019	16,17	Dissenyador
Proves unitàries	5días	15/04/2019	19/04/2019	18	Tester
Proves integrades amb ManifoldCF	2días	18/04/2019	19/04/2019	16,17	Tester
⊟ **Servei web**	10días	22/04/2019	03/05/2019	15	
Preparació de l'entorn	2días	22/04/2019	23/04/2019		Programador
Disseny	5días	22/04/2019	26/04/2019		Dissenyador
Implementació	5días	29/04/2019	03/05/2019	20,22,23	Programador
Proves unitàries	2días	02/05/2019	03/05/2019	20,22,23	Tester
⊟ **Front-end web**	10días	06/05/2019	17/05/2019	21	
Preparació de l'entorn	2días	06/05/2019	07/05/2019	24,25	Programador
Disseny	5días	06/05/2019	10/05/2019		Dissenyador
Implementació	5días	13/05/2019	17/05/2019	27,28	Programador
⊟ **Proves integrades completes**	15días	20/05/2019	07/06/2019	26	
Definició de les proves integrades	5días	20/05/2019	24/05/2019	29	Dissenyador
Realització de les proves integrades	10días	27/05/2019	07/06/2019	31	Tester
⊟ **Tancament del projecte**	15días	10/06/2019	28/06/2019		
Memòria	10días	10/06/2019	21/06/2019	32	Cap de projecte
Presentació Power Point	5días	24/06/2019	28/06/2019	34	Cap de projecte
Demo	5días	24/06/2019	28/06/2019	32	Cap de projecte
Presentació Final	1día	01/07/2019	01/07/2019	33	Cap de projecte

Figura 1. *Detall temporal de tasques de projecte*

Aquesta taula es correspon amb el diagrama de Gantt adjunt a l'annex d'aquest document.

El projecte serà portat a terme per només una persona, hauré d'assumir tots els rols presents al projecte: cap de projecte, dissenyador, programador i tester. Posat que no es disposa d'una completa disponibilitat d'horaris, ens impossibilitarà poder treballar a jornada completa (8h) així que cada dia calculat al calendari suposa mitja jornada (4h).

7.1.1 Gestió inicial del projecte

Aquesta part del projecte és la fase inicial que es correspon a la part de GEP, en aquesta part es va definir com es distribuiria tot el treball a realitzar. Aquest consta de 7 parts:

I. Contextualització
II. Gestió econòmica i sostenibilitat
III. Estat de l'art
IV. Abast del projecte
V. Planificació temporal
VI. Requisits
VII. Metodologia i rigor

7.1.2 ManifoldCF

Aquesta fase es basa en la instal·lació i configuració del recol·lector de documents (*crawler*) perquè sigui capaç d'accedir a múltiples repositoris i obtenir els documents que hi hagi, extraient els continguts d'aquests. Aquesta fase s'estructura en 5 parts detallades a continuació.

I. Preparació de l'entorn

En aquesta primera part es basa en preparar el sistema a on s'instal·larà el *software* necessari per la instal·lació del *ManifoldCF* i s'assegurarà que hi hagi suficient espai per tenir el *software* instal·lat i actiu. Inclou també l'obtenció del *software d'Apache*.

II. Instal·lació i configuració

Instal·lació i configuració de *ManifoldCF* per poder disposar de l *software* a nivell "productiu" per poder començar a desenvolupar el nostre projecte sobre aquest software de tercers.

III. Connectors entrada/sortida

ManifoldCF es basa en 3 parts. Connectors d'entrada (que connecten als repositoris a *crawlejar*), connectors de sortida (que connecten a la plataforma a on es volen guardar els elements i informació *crawlejada*) i transformadors, que s'encarreguen de transformar la informació obtinguda a l'entrada abans de guardar-la a la sortida. Aquesta part de la fase consta de la correcta creació i configuració dels connectors implicats al projecte.

IV. Jobs

Els *jobs* (o tasques) són els processos creats sobre *ManifoldCF* que s'encarreguen d'obtenir documents d'un repositoris concret, extreure la informació i guardar totes les dades a la sortida configurada (en el nostre cas, *Elasticsearch*). Es desenvoluparan

diversos *jobs* per tal de disposar de connexions a diversos tipus de repositoris, ja que forma part de la finalitat del projecte.

V. Proves unitàries

Realització de proves sobre els diferents components de *ManifoldCF* que s'utilitzaran per comprovar el correcte funcionament d'aquests abans d'integrar-los amb la resta de components del projecte (en el cas de *ManifoldCF*, la integració amb *Elasticsearch*).

7.1.3 Elasticsearch

La segona fase del projecte es basa en la instal·lació i configuració del indexador i motor de cerca *Elasticsearch*. Aquí es definirà el model de dades dels documents i es provarà que *ManifoldCF* i *Elasticsearch* estiguin ben connectats entre si. A continuació es detallen les 5 parts d'aquesta fase.

I. Preparació de l'entorn

Elasticsearch és un software de tercers. Aquest *software* requereix d'un conjunt d'eines prèviament instal·lades per al correcte funcionament d'aquest. En aquesta part de la fase es tractarà mantenir l'entorn en perfectes condicions per poder instal·lar i fer funcionar *l'Elasticsearch* sobre aquest.

II. Instal·lació i configuració

Instal·lació del *software Elasticsearch* i configuració de l'eina per poder començar a desenvolupar els nostres requisits sobre aquest.

III. Definició d'índex i templates

La informació que es guarda a *Elasticsearch*, queda registrada sobre els índexs que té aquest. Els índexs són com les "taules" d'una base de dades relacional. D'altre banda existeixen "*templates*", que venen a ser l'esquelet de mostra sobre el que es vol generar un nou índex. A la *template* és a on es defineixen els diferents elements i configuracions que tindrà l'índex abans de ser generat. En aquesta part de la fase ens centrarem en realitzar aquestes *templates* i generarem els índexs de dades.

IV. Proves unitàries

En aquesta part ens centrarem en comprovar que els índexs funcionen correctament i que es permet realitzar cerques sobre aquests mitjançant crides *curl*.

V. Proves integrades amb ManifoldCF

En la última part de la fase, ens centrarem en les proves integrades amb *ManifoldCF*. Aquestes proves ens donaran la certesa de que la indexació funciona correctament i que les dades que obté *ManifoldCF* queden correctament enregistrades a l'índex que haguem generat a *Elasticsearch*. Aquesta dependència és la més important del projecte, ja que suposa el "*core*" de la nostra eina.

7.1.4 Servei Web

En la tercera fase del projecte, es tracta de generar un servei web que permeti accedir a *Elasticsearch* per obtenir les dades que s'han guardat i es permeti fer cerques sobre aquestes dades. Consta de 5 parts.

I. Preparació de l'entorn

Desenvolupar aquesta part del projecte requereix disposar d'un conjunt d'eines prèvies. En aquesta part ens encarregarem d'obtenir i disposar de Java per picar el nostre servei, *IntelliJ* com a *front-end* per picar el codi i d'un *Tomcat* com a servidor sobre el que poder arrancar el servei un cop desenvolupat.

II. Instal·lació i configuració

En aquesta segona part de la fase, es tractarà de instal·lar i configurar totes les eines obtingudes en la part anterior (*Java*, *IntelliJ*, servidor *Tomcat*...). Es verificarà també el correcte funcionament d'aquestes eines abans de començar amb el disseny a desenvolupar.

III. Disseny

En aquesta part es definirà el disseny del servei a desenvolupar. Així com les connexions cap al motor de cerca i la inclusió de les regles de negoci que es vulguin aplicar posteriorment.

IV. Implementació

Implementació del codi Java amb les crides *HTTP* cap al motor de cerca *Elasticsearch*. Es seguirà el disseny prèviament establert a la part anterior d'aquesta mateixa fase.

V. Proves unitàries

Realització de petits tests per comprovar el correcte funcionament del servei web abans de ser integrat amb el motor de cerca *Elasticsearch*.

7.1.5 Front-end Web

En aquesta fase es tractarà de generar un *front-end* "amigable" per poder visualitzar les dades que es vulguin buscar a través del servei web de la fase anterior. Consta de 4 parts.

I. Preparació de l'entorn

Desenvolupar aquesta part del projecte requereix disposar d'un conjunt d'eines prèvies. En aquesta part ens encarregarem d'obtenir i disposar d'un compilador/verificador de codi *HTML* i per verificar el nostre frontal web.

II. Instal·lació i configuració

Instal·lació i configuració del compilador/verificador de codi *HTML* que s'utilitzarà per al desenvolupament d'aquesta part del projecte. S'instal·larà també *Google Chrome* als dos PCs que s'utilitzaran al projecte, un per verificacions durant el desenvolupament i l'altre sobre el que es realitzaran les proves finals.

III. Disseny

En aquesta part es realitzarà el disseny en que es basarà el *front-end* web a realitzar entrant en detall sobre els diferents components i les diferents vistes que suposarà aquest *front-end*.

IV. Implementació

Implementació del codi web que permetrà visualitzar les cerques que realitzi l'usuari sobre el motor de cerca. Durant el desenvolupament es comprovarà també la compatibilitat amb *Chrome* (requisit definit en el projecte per verificar la correcte visualització de la web).

7.1.6 Proves integrades completes

En aquesta fase es tracta de definir i realitzar un conjunt de proves integrades *end-to-end* de tot el sistema per verificar el correcte funcionament de tota la plataforma desenvolupada i corregir els diferents problemes o errors que puguin sorgir.

I. Definició de les proves integrades

Realització d'un pla de proves sobre el que s'exposaran tots els tests a realitzar pas a pas i els diferents resultats esperats de cadascun dels passos.

II. Realització de les proves integrades

Realització de totes les proves detallades en el pla de proves i correcció d'errors en cas de no passar correctament els tests que s'hagin definit.

7.1.7 Tancament del projecte

En aquesta fase es tracta com diu el seu nom, de tancar el projecte. Es a dir, acabar de redactar la memòria, preparar el material per la defensa final del projecte, etc.

I. Memòria

Realització del document final (memòria del TFG) informant i recopilant tota la informació corresponent a les diferents parts del treball de final de grau.

II. Presentació

Elaboració del document *Power Point* que s'utilitzarà en la presentació final davant el tribunal del projecte, que permeti exposar els diferents punts en que es basa el projecte per ajudar tant a la presentació com al tribunal al que s'exposi el TFG.

III. Demo

Preparació i realització d'un video com a demostració del projecte realitzat. On es pugui veure el funcionament d'aquest i les diferents parts en que es composa.

7.1.8 Presentació final

Aquesta és la última fase del projecte i correspon a la presentació presencial i defensa davant el tribunal de la Universitat Politècnica de Catalunya.

7.2 Planificació final

Tant amb el *Crawler* com amb *Elasticsearch* i el *front-end*, ens hem trobat en que la corba d'aprenentatge ha sigut més gran de l'esperada. Això ha fet que el temps que teníem planificat per desenvolupar les tasques corresponents a aquests punts del projecte s'endarrerissin.

Elasticsearch ha acabat tenint un conjunt de configuracions molt gran i una sintaxi pròpia per realitzar les consultes com a motor de cerca molt diferent a la sintaxi que coneixíem i hem tractat durant el pas per la universitat (*SQL*). Tot i que la documentació corresponent a aquesta tecnologia es molt complerta, hem hagut d'invertir molt temps en les proves i les diferents configuracions finals que s'han acabat establint per el nostre prototip.

En quan al *Crawler* de *ManifoldCF*, la part del *crawleig* de sistema de fitxers no ens ha suposat cap problema; però la part web ha acabat tenint més dificultats de les esperades, ja que en molts casos es requeria configuració de certificats dels que no disposàvem tot i que *ManifoldCF* permet l'ús d'aquests. Finalment vam acabar *crawlejant* una petita part de la *Wikipedia*, ja que no requeria l'ús de certificats ni usuaris de *login*. Tot i així, el gran volum de documents detectats al llarg de la web ha fet que s'endarrerís el procés de *Crawling* i acabéssim *crawlejant* un volum força petit degut a que el pc utilitzat no disposa d'una capacitat tan gran de memòria *RAM* ni d'una xarxa d'alta velocitat.
Al igual que *Elasticsearch*, la corba d'aprenentatge de l'eina ens ha suposat un problema sobre el calendari, ja que desconeixíem el funcionament intern d'aquesta i la documentació que hi ha disponible a la red es força escassa.

Com ja vam comentar a l'inici del projecte, un dels riscos era la limitació del calendari, ja que tenim unes dates prefixades sobre les que acabar el projecte i això ens podria fer que al llarg del temps que ens quedava arribéssim a descartar alguna part del desenvolupament del prototip. Finalment, tot i la contingència establerta els endarreriments ens han suposat uns canvis sobre els objectius finals.

7.2.1 Canvis sobre la planificació inicial i estat final

Durant els mesos de febrer i març es va realitzar la gestió inicial del projecte definint els punts a realitzar, la planificació, la gestió econòmica, l'estudi de l'estat de l'art i la definició dels diferents requisits que composen el nostre projecte. Tota aquesta part corresponia al lliurable de *GEP*.

Un cop acabada la gestió inicial, es va procedir a mitjans de març a realitzar les diferents tasques que suposava la part de *ManifoldCF* i el seu *crawlejat*. Donades les dificultats trobades per *crawlejar* algunes webs, finalment es va optar per la *wikipedia*. Donades les dimensions i el pc utilitzat, ens va suposar un problema de temps tot aquest procés i finalment es van emmagatzemar al voltant de 1500 pàgines web. En tot aquest procés i proves ens va suposar uns dies d'endarreriment sobre la planificació inicial.

El més d'abril es va dedicar a la configuració *d'Elasticsearch*. Vam veure que *Elasticsearch* oferia moltes més opcions de configuració de les esperades i que cada prova de configuració sobre els índexs suposava reindexar els continguts *crawlejats*. Això ens va fer endarrerir encara més la planificació inicial. Finalment a finals d'abril es va establir una configuració "final" amb documents de la *wikipedia* i documents obtinguts durant el transcurs de la universitat per el connector de sistema de fitxers, donant per acabades les proves integrades entre els dos components.

A principis de maig es va procedir a realitzar el servei web en *Java* com a cercador deixant com una segona fase la part d'introducció de regles de negoci. Cap a mitjans de maig es va començar a implementar una primera versió del frontal web per visualitzar els continguts que es permetien buscar actualment. Aquesta part ens va portar encara més endarreriment al que ja teníem acumulat durant els mesos anteriors ja que teníem un coneixement mínim de *Html* i cap coneixement sobre *JSP*. Per realitzar les peticions de cerca no ens va suposar molt problema, però a l'hora de rebre la resposta i pintar-la ens vam trobar en que la manera que s'ha de passar en *JSP* i les llibreries externes a incloure, no son trivials. Això ens va fer que haguéssim d'invertir més temps en aquesta part del prototip que vam haver de treure de la part de la incorporació de regles de negoci.

Finalment, a dia d'avui en quan al prototip, podem donar per acabades les parts corresponents a la indexació i a la cerca de documents des d'un *front-end*, però la "Fase 2" que suposava la millora del prototip afegint les regles de negoci, ha quedat parada únicament com a disseny.

7.2.2 Impacte sobre els objectius

La problemàtica anteriorment explicada ens va ha suposar un endarreriment sobre el calendari establert inicialment. Ja vam comentar que un dels riscos era la limitació per calendari. Però no vam tenir en compte la corba real d'aprenentatge que suposava el conjunt de softwares utilitzat en el projecte.

Aquest risc de projecte ens va suposar un canvi sobre la planificació inicial que finalment ens ha acabat obligant a prescindir de la part de regles de negoci del prototip generat, juntament amb un *front-end* més treballat. El *front-end* del que disposa el prototip actualment és força simple i senzillament permet llençar cerques i mostrar els resultats d'aquestes, però no permet altres funcionalitats que teníem en ment com és el selector d'idioma de la cerca, el *paginador* d'elements i el selector de metadades de la incorporació de les regles de negoci per fer filtres de cerca.

La contingència establerta al inici del projecte no va ser suficient donada la inexperiència de que disposàvem sobre els softwares utilitzats. Això ens ha fet que no només ens afectés al calendari inicial i sobre els objectius comentats, sinó que el temps que ens portaria acabar aquestes parts també implicaria un canvi sobre el temps de dedicació i, per tant, sobre pressupost del projecte.

El pressupost es veuria afectat a tots nivells de recursos humans, ja que caldrien hores de programadors que desenvolupessin la part afectada, testers que provessin els diferents objectius que no s'han portat a terme i cap de projecte que dirigís i documentés tots aquests objectius.

Tot i així, considerem que en un equip real amb recursos humans amb experiència s'hagués pogut portar a terme l'abast definit inicialment amb el calendari establert.

7.3 Recursos i requeriments

Equip de persones:

- Cap de projecte
- Dissenyador
- Programador
- Tester

Lloc de treball:

- Casa
- Universitat

Hardware:

- PC (x2)

Software:

- Windows 10
- IntelliJ
- Microsoft Office 2010
- ManifoldCF
- Elasticsearch

8. Gestió econòmica

En aquesta part és proporciona l'estimació del cost del projecte tenint en compte recursos humans, *hardware*, *software* i també els costos indirectes.

8.1 Sostenibilitat

En aquesta part parlarem sobre la sostenibilitat del projecte. I profunditzarem una mica sobre l'impacte que pot tenir sobre les àrees econòmica, social i ambiental.

A continuació es mostren les puntuacions de cadascuna de les àrees de sostenibilitat.

Puntuació sostenibilitat			
¿Sostenible?	**Econòmica**	**Social**	**Ambiental**
Planificació	9	8	9

Figura 2 *Puntuació sostenibilitat*

A continuació detallem cada punt de la sostenibilitat i justificarem el perquè de les puntuacions de cada àrea.

8.1.1 Econòmica

Tal com es justificarà a la part de "Gestió econòmica", és un projecte amb un cost relativament baix. Això és degut a que es requereix un treball sobre la realització del projecte, però un cop acabat, és totalment portable i no tindrà cost addicional algun. Tampoc cal un manteniment i té una llarga durada ja que és un sistema estable i força actual. El que pugui canviar és totalment temes de configuracions i filtres que es vulguin afegir, però per això s'ha millorat el sistema de configuració de les cerques, perquè siguin els propis usuaris qui els puguin anar adaptant a les necessitats del moment.

Aquest buscador no està enfocat a obtenir un benefici econòmic sinó més aviat social. Tot i que si s'explotés el cost entre els potencials clients, es podrien extreure beneficis.

Considerem que el cost del projecte s'ajusta molt a un cost real d'un projecte d'aquesta mena i que en cap moment s'han inflat els costos. Per tant considerem que té un preu totalment competitiu en cas que sorgissin competidors directes.

D'altra banda el seu cost és totalment assequible per qualsevol entitat. I molt interessant per la majoria (sinó totes) les institucions que tinguin un volum de documents/llibres o fins i tot una intranet que vulguin compartir a uns usuaris determinats.

8.1.2 Social

Aquest projecte beneficiarà als usuaris finals del buscador, ja que no hauran de destinar tant temps en buscar els fitxers o recursos en els diferents repositoris en que es puguin trobar, i també beneficiarà a qui ofereixi els elements sobre els que buscar ja que no haurà de destinar recursos per facilitar als usuaris finals la obtenció d'aquests documents.

Per exemple, si l'usuari final fos una biblioteca, beneficiaria per una part als alumnes o usuaris de la biblioteca que esperessin trobar algun contingut concret entre tots els recursos de la biblioteca i també als treballadors, ja que no haurien d'atendre a tants usuaris per ajudar-los a trobar els continguts que busquin.

Un altre clar exemple seria si l'usuari final fos una empresa que disposa de diverses intranets sobre les que té repartits molts documents. En aquest cas beneficiaria als usuaris ja que no haurien d'estar saltant entre les diverses pàgines d'intranet ja que disposarien d'un sistema centralitzat d'es d'on obtenir qualsevol dels documents d'aquesta intranet alliberant-los de la pèrdua de temps que suposa la recerca d'un element concret del que es desconeix l'origen.

En qualsevol cas, queda clar que aquesta eina suposa una millora en el temps dedicat a mantenir un sistema com pugui ser una biblioteca o una intranet i sobretot una millora de cara als usuaris que no hauran de dedicar tant temps a una tasca de recerca i podran dedicar el temps en altres tasques més productives.

8.1.3 Ambiental

Degut que els projectes de *software* tenen poca repercussió en el medi ambient, aquest projecte no te una gran impacte en l'àrea ambiental.
Tot i així, també afecta subtilment al medi ambient degut a la necessitat d'energia elèctrica i de paper, dos factors que afecten en el seu procés d'obtenció.
Considerem que l'impacte tot i no ser considerable, és força positiu degut a que no influeix gairebé gens sobre la part negativa. Té un baix consum elèctric, emet molt poc CO2, té una vida útil alta ja que el seu gestor de continguts permet incorporar tots els continguts que es desitgin sobre el sistema de cerca.

A més la incorporació d'un cercador a qualsevol empresa o entitat pública, permetrà reduir el consum de paper ja que no requereix d'un sistema de gestió i control en paper. Avui en dia i cada cop més, les millores tecnològiques ajuden a reduir el consum de productes materials (paper, boli…) i dels temps de realització de tasques i aquest projecte n'és un clar exemple.

8.1.4 Informe autoavaluació (sostenibilitat)

Actualment disposo d'experiència laboral amb projectes relacionats amb la sostenibilitat econòmica. Actualment la majoria d'empreses privades es centren en aquest aspecte de la sostenibilitat així que no es d'estranyar que als treballadors ens resulti familiar aquest aspecte de la sostenibilitat.

Habitualment les empreses privades disposen d'eines per mesurar els costos econòmics dels projectes que porten a terme i sovint la producció que suposaran

aquests. Les formules que porten aquestes eines solen ser concretes de les empreses, ja que solen incloure un marge de benefici que és totalment particular de cada organització.
Personalment no he gestionat cap projecte abans durant la meva vida laboral i per tant no he fet ús d'aquestes eines, però si he les he pogut veure i entendre com funcionen.
En el cas particular d'aquest projecte, vaig recórrer a fórmules genèriques que ja existien en l'actualitat (publicades entre la documentació de GEP) però sempre fent ús de la poca experiència de la que disposo. Tot i així en projectes d'aquests tipus sempre acaben sorgint problemes inesperats que ens acaben afectant a la sostenibilitat econòmica d'aquest.

Com a segon aspecte diria que la sostenibilitat social suposa la segona dimensió més familiar per mi, ja que en alguns projectes s'ha mirat més per els treballadors que per l'economia del projecte. Això es podria exemplificar amb projectes que suposen eines de treball millors a les que té una empresa per ajudar als treballadors a desenvolupar de manera més fàcil les seves tasques. No puc entrar en detall en projectes en els que he participat degut a clàusules de confidencialitat, però em resulten força habituals en el treball.

Finalment afegiria que sobre la sostenibilitat ambiental no m'hi he trobat durant la meva vida laboral ja que son poc habituals en les empreses privades a no ser que tinguin projectes amb entitats com ara ajuntaments o terceres empreses enfocades a millorar el medi ambient.

8.2 Pressupost recursos humans

El projecte serà portat a terme per només una persona, hauré d'assumir tots els rols presents al projecte: cap de projecte, dissenyador, programador i tester. Posat que no es disposa d'una completa disponibilitat d'horaris, ens impossibilitarà poder treballar a jornada completa (8h) així que cada dia calculat al calendari suposa mitja jornada (4h).

A continuació es detalla el nombre d'hores que es preveu que caldran per dur a terme totes les tasques de cada rol en el projecte i el valor econòmic de les hores dedicades.

Pressupost recursos humans			
Rol	**Hores**	**Preu/h**[3]	**Preu total**
Cap de projecte	280h	60€/h	16.800€
Dissenyador	68h	40€/h	2.720€
Programador	116h	35€/h	4.060€
Tester	96h	20€/h	1.920€
Total	**560h(*)**		**25.500€**

***Figura 3**. Pressupost recursos humans*

[3] Tots els salaris per hora estan extrets de l'empresa en la que treballo, *Everis,* la qual disposa d'una política de transparència de cara al públic i, com a tal, son consultables. No son els sous reals dels treballadors sinó el preu/hora que es cobra a client per projecte i rol.

(*) El total d'hores podem veure que no correspon directament al total de dies de la planificació del projecte (115 dies) a 4h de jornada laboral dedicades al dia (Si es fes el càlcul directe, el projecte hauria de tenir una durada de 460h (115 dies x 4h)). Això és degut a que com podem veure al diagrama de Gantt (adjunt a l'annexa), i com hem comentat abans, al no disposar d'un equip de persones i, al haver dates superposades, es treballarà amb dos rols i en més d'una tasca en algunes etapes del projecte. Aquest fet ens obligarà a dedicar més temps de jornada laboral a cada dia de superposició i ens obligarà a treballar en dos rols diferents per poder arribar a temps a la data definida com a final de projecte.

8.3 Pressupost hardware

Per a poder realitzar la implementació del projecte, seran necessaris certs elements *hardware* per tal de realitzar la implementació i proves del sistema que s'han d'incloure en el pressupost final ja que formen part d'aquest projecte.

Pressupost hardware				
Producte	**Preu**	**Unitats**	**Vida útil**	**Amortització**
PC 1 (Servidor)	1000€	1	4 anys	159,09€ (*)
PC 2 (Client)	1000€	1	4 anys	159,09€ (*)
Total	**2000€**			**≈** 320,00€

***Figura 4**. Pressupost hardware*

(*) L'amortització s'ha tingut en compte seguint la següent fórmula:

$$\frac{Preu\ de\ compra\ de\ l'equip(€)}{Vida\ útil\ equip\ anys\ (4) * Dies\ feiners\ any * (220) * Hores\ treball\ dia\ (4h)}\ x\ Hores\ d'ús\ equip\ TFG\ (h)$$

$$\boldsymbol{PC\ 1\ (Servidor)} \Rightarrow \frac{1000\ €}{4\ anys * 220\ Dies\ feiners * 4h}\ x\ (560h) =\ 159{,}09\ €$$

$$\boldsymbol{PC\ 2\ (Client)} \Rightarrow \frac{1000\ €}{4\ anys * 220\ Dies\ feiners * 4h}\ x\ \ (560h) = 159{,}09\ €$$

8.4 Pressupost software

A part del material de hardware, també es necessitarà cert software específic, de manera que puguem utilitzar el hardware especificat anteriorment i portar a terme totes les fases del projecte.

Pressupost software				
Producte	**Preu**	**Unitats**	**Vida útil**	**Amortització**
Windows 10	200€	1	3 anys	42,43€ (*)
Microsoft Office	140€	1	3 anys	29,70€ (*)
IntelliJ	0,00€	1	---	0,00€
ManifoldCF	0,00€	1	---	0,00€
Elasticsearch	0,00€	1	---	0,00€
Total	**340€**			**≈ 72,00€**

Figura 5. *Pressupost software*

(*) L'amortització s'ha tingut en compte seguint la següent fórmula:

$$\frac{Preu\ de\ compra\ de\ l'equip(€)}{Vida\ útil\ equip\ anys\ (4) * Dies\ feiners\ any * (220) * Hores\ treball\ dia\ (4h)}\ x\ Hores\ d'ús\ equip\ TFG\ (h)$$

$$\boldsymbol{Windows\ 10} \Rightarrow \frac{200\ €}{3\ anys * 220\ Dies\ feiners * 4h}\ x\ (560h) =\ 42{,}43\ €$$

$$\boldsymbol{Microsoft\ Office} \Rightarrow \frac{140\ €}{3\ anys * 220\ Dies\ feiners * 4h}\ x\ (560h) = 29{,}70\ €$$

8.5 Despeses indirectes

A part de les despeses en personal, *software* i *hardware* hem de tenir en compte altres despeses del projecte com poden ser l'ús de paper o l'electricitat.

Despeses indirectes			
Producte	**Preu**	**Unitats**	**Cost aproximat**
Electricitat	0,15€/kWh	560,0kW	84,00€
Paper	4,00€/pack	1 pack	4,00€
Total			**88,00€**

Figura 6*. Despeses indirectes*

8.6 Contingència

Donat que com s'ha comentat a l'abast del projecte, tenim limitacions de calendari, s'ha de tenir en compte una contingència per si s'arribés a donar el cas, ja que suposaria un sobre cost sobre el pressupost fixat a recursos humans i sobre el hardware.

Contingència	
Concepte	**Cost aproximat**
Contingència (*)	142,00€
Total	**25.980,00** €

Figura 7. *Contingència*

(*) La contingència s'ha tingut en compte en base a una desviació temporal de 10 dies (gairebé el 10% del temps del projecte) amb una ocurrència del 25% de probabilitat que això passi. Això suposaria un sobre cost sobre els recursos humans, un sobre cost sobre els recursos de hardware i un sobre cost sobre les despeses indirectes.

La formula utilitzada per treure el total de contingència és la següent:

$$\boldsymbol{Contingència} = Ocurrencia\ (\%)\ x\ \frac{Cost\ dels\ recursos\ del\ projecte\ (€)(*)}{Dies\ totals\ de\ projecte\ (d) * Hores\ treball\ dia\ (4h)}\ x\ desviació\ (d)$$

*El cost dels recursos del projecte son la suma dels costos de recursos humans, Software, Hardware y les despeses indirectes.

$$\boldsymbol{Contingència} = 25\%\ \frac{(25.500 +\ 320\ + 72 + 88)\ (€)}{115\ (d) *\ (4h)}\ x\ 10\ (d) =\ 142{,}00\ €$$

8.7 Pressupost total

Finalment, ajuntem tots els pressuposts en conjunt amb les despeses per poder calcular el cost total del projecte.

Pressupost projecte	
Concepte	**Cost aproximat**
Recursos humans	25.500,00€
Hardware	320,00€
Software	72,00€
Costos indirectes	88,00€
Pressupost projecte	**25.980,00** €

Figura 8. *Pressupost projecte*

Tenint en compte el pressupost dedicat a contingència i possibles imprevistos, el total pujaria a:

Pressupost Final	
Concepte	**Cost aproximat**
Pressupost Projecte	25.980,00 €
Imprevistos (5% sobre el total)	1299,00 € (*)
Contingència	142,00 €
Total	**27.421,00** €

Figura 9. *Pressupost final*

(*) Els imprevistos són un fons de reserva que suposa un augment del 5% sobre el cost total del projecte per assumir una possible desviació temporal i que aquesta desviació no suposi un augment no controlat del pressupost final.

8.8 Control de gestió y desviacions

Com ja hem comentat, un dels principals problemes que pot portar el projecte i que hem de controlar, ja que podria afectar al pressupost final, es la desviació temporal respecte a la planificació realitzada, ja que un increment temporal sobre el calendari ens incrementaria el cost de recursos humans i el de despeses indirectes.
Per controlar això hem establert la planificació amb el diagrama de Gantt i es pretén complir rigorosament aquesta dedicació mitjançant reunions quinzenals on es verifiqui que cada tasca realitzada s'hagi finalitzat correctament.

Assumirem aquest cost de desviació temporal sense carregar-lo al projecte, ja tenim previst un lleuger increment que pot dedicar-se al cost dels recursos humans de 1299,00€ (5% del cost del projecte) fixat com a imprevistos, a més del ja comentat, fons de contingència dedicat exclusivament a aquesta possible desviació.

També ens podem trobar amb el problema de mal funcionament del hardware, en

aquest cas hauríem d'afegir els costos de les reparacions o en cas extrems, adquirir nou hardware.

La possibilitat que passi això és bastant baixa i per solucionar aquest possible imprevist s'ha decidit que en cas que algun dels pc's doni problemes s'utilitzaria un pc propi i no es carregarà el cost addicional que suposaria un nou element de hardware al projecte.

9. Arquitectura

En aquest punt mostrarem l'arquitectura que composa el nostre sistema flexible de cerca en múltiples repositoris, començant per l'arquitectura global del sistema amb els diferents components i baixant una mica al detall al conjunt de components quer composa el *Crawler* i el servei de cerca.

9.1 Arquitectura del projecte

El projecte a desenvolupar està format per els següents components:

- **Repositoris d'entrada**: Repositoris a *crawlejar*. A la imatge es mostren el llistat de connectors d'entrada que porta *ManifoldCF* ja incorporats (s'ha marcat en groc els dos repositòris *crawlejats* al prototip).

- **Crawler**: *Crawler* de documents format per *ManifoldCF*. *ManifoldCF* inclou l'extractor de text i metadades *Tika*. El *crawler* està desplegat sobre un servidor *Jetty* integrat (*Embedded*).
- **indexador/Motor de cerca**: Indexador de documents i motor de cerca *Elasticsearch*. Aquest component s'encarrega d'emmagatzemar els documents i gestionar les cerques.

- **Servei**: Servei desenvolupat en *Java*. Aquest servei s'encarrega de rebre un conjunt d'informació amb el que llençarà les cerques sobre *Elasticsearch*. El servei està desplegat sobre un servidor *Tomcat*.

- **Front-end**: Frontal web. Aquest frontal és el component des que els usuaris finals podran enviar el text que volen buscar i veure els resultats. Sobre el prototip s'ha desenvolupat en *Html* i *JSP* i està desplegat sobre el mateix *Tomcat* que el servei, però la idea d'aquest component és que l'entitat client que vulgui muntar aquest servei de cerca es generi el seu propi frontal amb els logos de l'entitat.

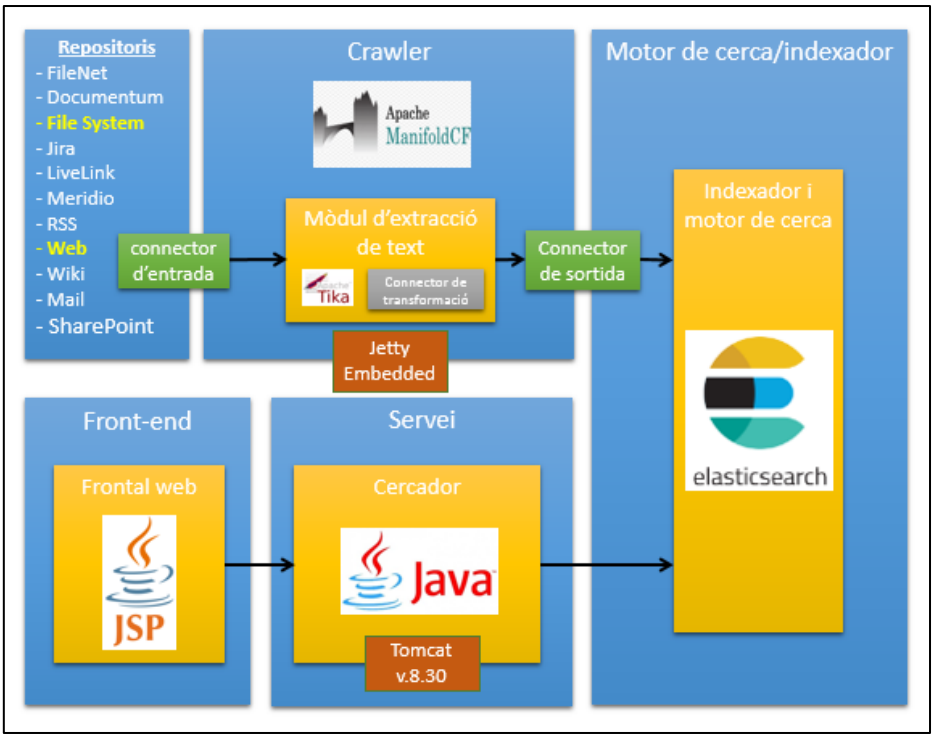


Figura 10. *Diagrama de components*

9.2 Arquitectura del Crawler

El sistema de *crawleig* disposa d'un conjunt de components que formen la seva arquitectura.

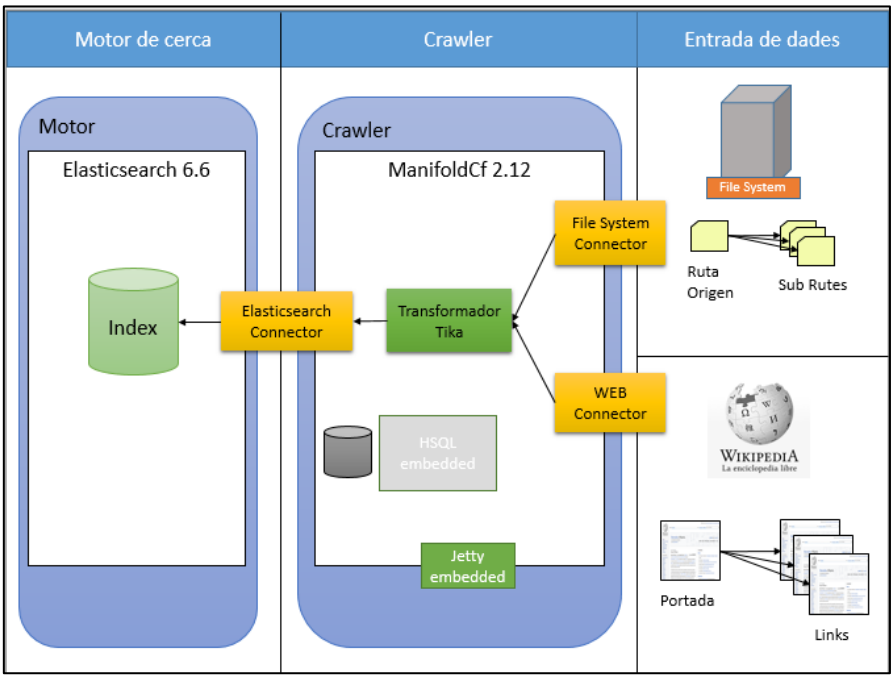


Figura 11. *Diagrama components del Crawler*

El sistema de *crawlejat* es llança des de *ManifoldCF*. *ManifoldCF* llença una conjunt de connexions sobre els repositoris d'entrada mitjançant el connector d'entrada corresponent al tipus de repositori a crawlejar.

Acte seguit, es comença el *crawleig*. Independentment de cada tipus de repositori, *ManifoldCF* detecta des del punt d'origen establert, tots els subconjunts a *crawlejar* i estableix l'arbre de *crawleig*. *ManifoldCF* disposa d'una base de dades interna (*HSQL*) a on controla tot aquest procés i en cas de no ser el primer cop que es llanci aquest procés, és capaç de detectar si un document ha sigut modificat o es manté igual que l'anterior cop que es va *crawlejar*. En cas de ser igual, *ManifoldCF* salta el procés d'extracció i indexació i continua el *crawleig* amb la resta de documents.

La base de dades *HSQL* ve d'entrada configurada per *ManifoldCF*, però aquest Software permet connectar amb altres bases de dades com ara *MySql* o *PostgreSql* entre altres. Només cal modificar el fitxer de configuració amb la ruta del *driver* i els paràmetres de connexió i autenticació de la base de dades a la que es vol connectar.

Un cop *ManifoldCF* obté un document per el connector d'entrada, aquest document passa per el transformador de *Tika*. Aquest extreu el contingut del fitxer i el passa de binari (extret de origen a text pla). *Tika* també extreu un conjunt de metadades que varien en funció del tipus de document obtingut.

Un cop extret el text i les metadades, *ManifoldCF* mou el document al connector de sortida que s'hagi definit. En el cas d'aquest projecte aquest connector de sortida correspon al connector *d'Elasticsearch*. Aquest connector genera una crida *HTTP Put* amb un missatge *JSON* amb el document i les metadades extretes durant el procés anterior i l'envia a *Elasticsearch* a on poden passar dos casos, que el document existeixi al índex configurat al connector de sortida o que sigui nou. Si un document és nou, aquest s'indexa i passa a formar part dels documents guardats a *Elasticsearch*. En cas que ja existís, el document es sobreescriu amb la nova versió *crawlejada*.

D'entrada *ManifoldCF* arranca sobre un *Jetty* intern (*embedded*). Però aquest es podria extreure i generar un paquet *Jar* o *War* que podria ser desplegat sobre altres servidors.

9.3 Arquitectura del Servei de cerca

El sistema de cerca que s'ha dissenyat, consta de tres blocs. El front-end, encarregat de visualitzar les cerques, el servei de cerca, on cau tota la lògica de la cerca i el motor de cerca que és a on estan els documents i qui realitza la cerca establerta per el servei sobre aquests documents.

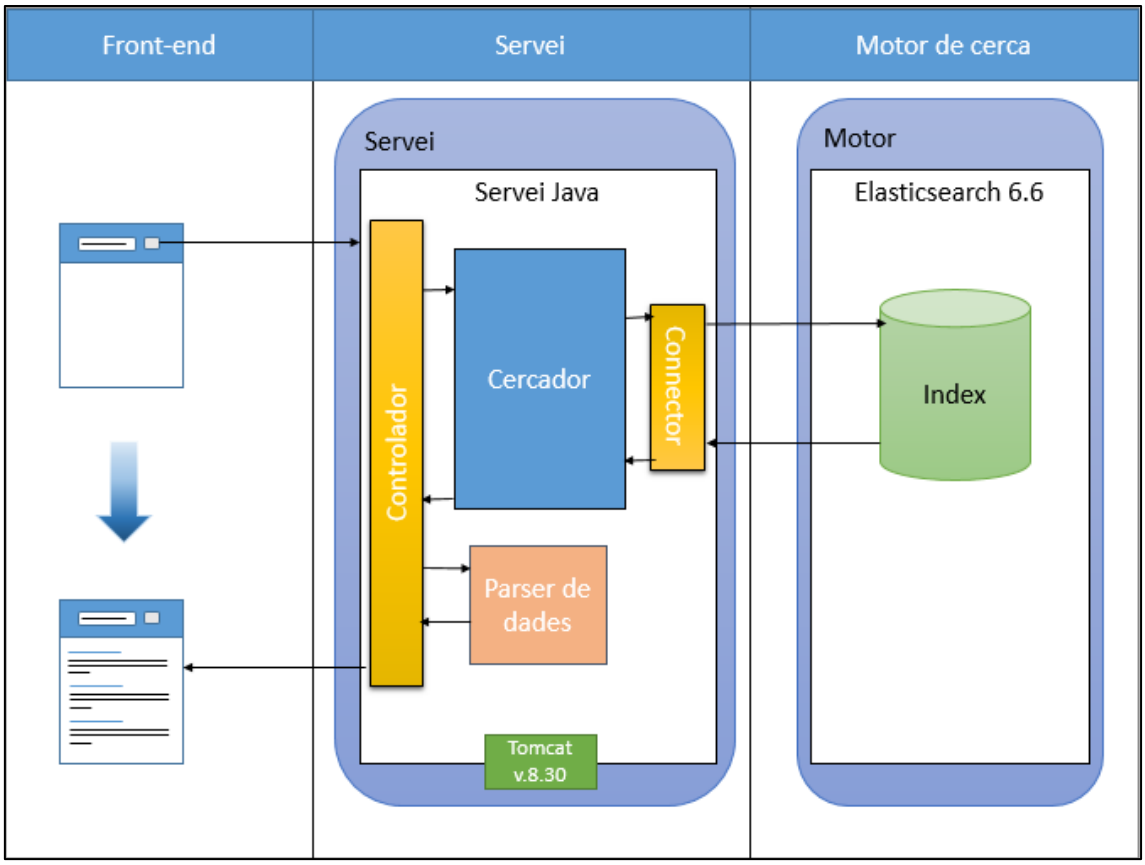


Figura 12. *Diagrama components del Crawler*

El *front-end* està format per una pàgina web. Aquesta web disposa d'un camp de text a on es pot escriure el text a buscar i un botó que llença el procés de cerca.

Un cop s'obtenen els resultats de la cerca, aquests es mostren en forma de llistat mostrant per cada resultat un títol que porta el *link* al document, un breu text a on es mostra on s'ha trobat el text buscat (es marca en negreta el text exacte) i un camp que et mostra la puntuació de la cerca (merament informatiu).

Aquest *front-end* llença peticions *Http* Post al servei de cerca dissenyat en Java i aquest retorna les dades trobades en un format *Html*.

D'altra banda, el Servei de Cerca està format per una aplicació Java que té desplegat un *endpoint* en el seu controlador, que espera peticions *Http* Post. Aquest servei rep les peticions amb un conjunt de camps esperats (per més detall, veure la part tècnica del servei de cerca en aquest mateix document), extreu caràcters conflictius i muntar la "*query*" de cerca en format JSON. Un cop muntada la *query*, llença una petició *Http* Post sobre el motor de cerca amb la "*query*" a través d'un connector d'entrada. Un cop rep la resposta del motor de cerca, aquesta arriba en format JSON, la retorna al controlador, i aquest l'envia al component "*parser*", que s'encarrega de descompondre en elements la cerca i muntar la resposta esperada per el *front-end*. Un cop muntada la resposta, el *Parser* la retorna al controlador i aquest l'envia al *front-end*.

L'aplicatiu Java està desplegat sobre un servidor *Tomcat v8.30.*

Finalment, el Motor de cerca està format per *Elasticsearch*. Aquest conté un índex de dades amb els documents i les metadades. Està desplegat com a servei i disposa d'un *endpoint* d'entrada sobre l'índex específic de dades que espera peticions *Http* Post amb un cos de missatge *JSON* amb una *query* formada correctament amb la sintaxi *d'Elasticsearch*.

9.4 Disseny tècnic Servei de cerca

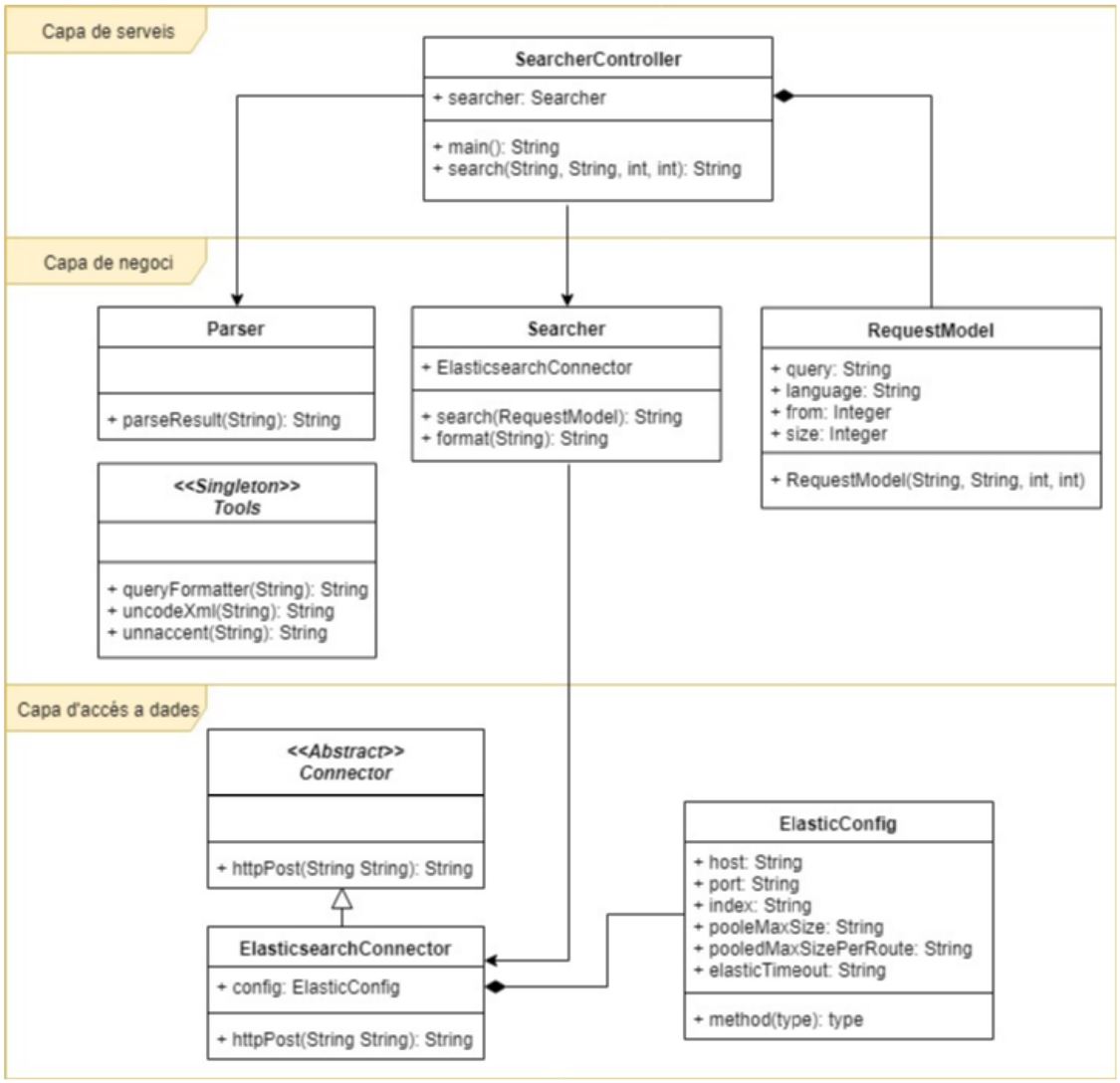


***Figura 13**. Disseny tècnic del servei de cerca*

De cara al disseny tècnic del servei de cerca, s'ha optat per un model d'Arquitectura per capes. Aquest disposa de tres capes:

- Capa de serveis: Encarregada de publicar *l'endpoint* i gestionar l'entrada i sortida de dades.
- Capa de negoci: Encarregada de gestionar tota la part referent a negoci (la *query* a utilitzar, el filtrat de caràcters específics, el model d'enviament de dades...).
- Capa d'accés a dades: encarregada de gestionar les connexions externes amb el motor de cerca.

En aquest disseny s'han usat els següents patrons de disseny:

Patró Singleton

En enginyeria de programari, *Singleton* o instància única és un patró de disseny que permet restringir la creació d'objectes pertanyents a una classe o el valor d'un tipus a un únic objecte. La seva intenció consisteix a garantir que una classe només tingui una instància i proporcionar un punt d'accés global a ella.[4]

En el nostre projecte l'hem aplicat sobre la classe que conté un conjunt d'eines de tractament de dades tals com filtrat de caràcters amb accents, codi *Xml*, etc.

Patró Bridge

El patró *Bridge* és una tècnica usada en programació per desacoblar una abstracció de la seva implementació, de manera que ambdues puguin ser modificades independentment sense necessitat d'alterar per això l'altra.[5]

Això permet realitzar canvis sense impacte sobre els clients. En el nostre cas l'hem aplicat sobre el connector al motor de cerca ja que això permetrà poder modificar el motor de cerca sense impactar sobre el servei realitzat. Només s'hauria d'implementar el nou connector.

[4] https://es.wikipedia.org/wiki/Singleton
[5] https://en.wikipedia.org/wiki/Bridge_pattern

10. Tecnologies

Un cop definida l'arquitectura i els diferents components que composen el nostre sistema flexible de cerca sobre múltiples repositoris, en aquest punt explicarem tota la part tècnica dels diferents components que formen la nostra arquitectura.

10.1 ManifoldCF

ManifoldCF és una eina de *crawleig*. En aquest punt explicarem cóm l'hem configurat sobre el prototip realitzat i què s'hauria de tenir en compte de cara a qualsevol usuari final.
Primer de tot definirem els components bàsics que suposen muntar el *crawler.* Un procés de *crawleig* ve definit per:

- Una entrada a repositori
- (Opcional) Un o més transformadors
- Una o més sortides de dades

Els processos es defineixen com *Jobs*, nom que s'utilitzarà a partir d'aquest punt. Cada *job* disposa únicament d'una entrada a un repositori. Per tant si es volen recol·lectar dades de més d'un repositori (independentment del tipus) s'han de crear *jobs* per cadascun d'ells. En el prototip realitzat, s'han generat connectors d'entrada per Sistema de Fitxers i per Web.

ManifoldCF disposa d'un gran nombre de connectors d'entrada:

Connector interoperability table			
System	**Server Platform**	**Client Version**	**Server Version**
Alfresco	Various	Tested using the Alfresco Web Services Client 4.0.b	Tested with Alfresco 2.x, 3.x and 4.x
CMIS	Various	CMIS 1.0	CMIS 1.0
DropBox	Various	1.5.3	N/A
Email	Various	Javamail 1.4	N/A
File System	Win/*NIX	N/A	N/A
Google Drive	Various	v2-rev64-1.14.1-beta	N/A
HDFS	Various	2.2.0	1.1.2
Windows Shares	Win, Samba, NetApp, other NAS systems	N/A	N/A
JDBC	Various	Supports JDBC V2, V3, V4; tested with Oracle 10, JTDS 1.2, PostgreSQL 9.1, MySQL 5.5 drivers	Various
Jira	Various	N/A	5.0-6.1
RSS	N/A	N/A	Atom, RSS 2.0, others
Web	N/A	N/A	HTML Version 1.0, 1.1, 2.0, Atom, RSS 2.0, others
Wiki	N/A	N/A	Wiki version 1.8 and above
LiveLink (OpenText)	Win	LAPI 9.7.1, 10.2.0	Tested with 9.2.0 - 10.2.0
Solr	N/A	N/A	Tested with Solr 1.4,

			3.6.2, 4.0.0, 4.1.0, 4.2.0, 4.3.0, 4.5.1
OpenSearchServer	N/A	N/A	Tested with OpenSearchServer 1.2.1, 1.2.2, 1.2.3, 1.3, 1.4, 1.5.x
ElasticSearch	N/A	N/A	Tested with ElasticSearch 1.0, 1.1, 1.2, 1.3
Documentum (EMC)	Win, RedHat	Tested with DFC 5.3 SP5	Tested against 5.3, 6.0, and 6.5 servers
SharePoint (MSFT)	Win	N/A	Tested with SharePoint 2003 (2.0), 2007 (3.0), 2010 (4.0), 2013 (5.0)
Meridio (Autonomy)	Win	N/A	Tested with Meridio 4.1, 5.0
FileNet (IBM)	Win, RedHat	Tested with P8 V4.1, V4.5	Tested with P8 V4.1, V4.5

***Figura 14**. Tabla de connectors de ManifoldCF*

Per més informació sobre els connectors, veure:
https://manifoldcf.apache.org/release/release-1.10/en_US/included-connectors.html

Els transformadors són totalment opcionals. D'entrada *ManifoldCF* ja incorpora un conjunt de transformadors. Els transformadors permeten realitzar modificacions o afegits sobre les dades obtingudes mitjançant el connector d'entrada. En el nostre prototip i de cara a futurs usuaris de l'eina, s'hauria d'utilitzar el transformador "*Tika content extractor*". Aquest transformador incorpora *Tika*, una eina *d'Apache* (mateix fabricant de *ManifoldCF*) que permet extreure continguts i metadades dels fitxers obtinguts amb el connector d'entrada. Aquest component ens incorpora la conversió del text obtingut en binari a text pla que podrem emmagatzemar per després realitzar les cerques. També incorpora tot i que al prototip no s'ha explotat, un conjunt de metadades extres en funció del tipus de document *crawlejat* (per exemple, autor del document, data d'última modificació, ...).

De cara a la sortida de dades, *ManifoldCF* incorpora un conjunt força ampli de connectors. Entre ells s'ha escollit el connector *d'Elasticsearch* per enviar les dades directament al motor de cerca. Incorpora altre connectors a altres tipus de bases de dades o fins i tot el connector a *null* per no enviar la sortida enlloc (s'ha utilitzat sobre el prototip per realitzar les proves unitàries mentre no es disposava d'un *Elasticsearch* degudament configurat). La incorporació de múltiples sortides podria permetre a un usuari duplicar les dades obtingudes sobre diferents repositoris de sortida o sobre dos índexs *d'Elasticsearch* diferents per mantenir un *backup* de les dades sense haver de realitzar explícitament una còpia de seguretat.

10.1.1 Connectors

En aquest punt explicarem les configuracions a tenir en compte sobre els diferents connectors necessaris per muntar un Job a ManifoldCF. El detall pas a pas de com crear cadascun dels diferents connectors del nostre prototip i punts a tenir en compte de cara als futurs usuaris de l'eina es pot trobar a l'Annex.

10.1.2 Connectors d'entrada

Els connectors d'entrada tenen en comú un conjunt de pestanyes de configuració bàsica i després en funció de cada tipus de connector unes pestanyes per la configuració avançada. Separarem per tipus de connector les dades necessàries a tenir en compte.

Connector d'entrada a sistema de fitxers:

- Nom del connector
- Tipus de connector (En aquest cas, "*FileSystem*")
- Número de connexions màxim cap al repositori

Connector d'entrada web:

- Nom del connector
- Tipus de connector (En aquest cas, "*Web*")
- Número de connexions màxim cap al repositori
- Email. Es opcional, serveix per configurar posteriorment notificacions
- Robots. Algunes webs disposen d'aquestes etiquetes per controlar el comportament dels *Crawlers*
- Ampli de banda màxim
- Credencials d'accés a les diferents webs a *crawlejar*
- Certificats en cas de seguretat SSL
- *Proxy* en cas de disposar d'un *Proxy* de sortida

Un cop acabada la configuració només cal prémer el botó "*Guardar*" per guardar el connector i poder utilitzar-lo en els *Jobs* posteriorment.

10.1.3 Connectors de transformació

Els connectors de transformació son aquells que un cop extrets els documents mitjançant una entrada, permeten realitzar modificacions sobre aquests o sobre les seves metadades. En el cas del nostre prototip hem utilitzat el connector de transformació de *Tika*, que ens permet extreure el contingut dels documents i permet també extreure més metadades en funció del tipus de document detectat. Les dades necessàries a emplenar son les següents.

Connector de transformació *Tika*

- Nom del connector
- Tipus de connector (En aquest cas, "*Tika content extractor*")
- Número de connexions màxim

- Configuracions addicionals que, en el cas del prototip s'ha deixat en blanc ja que no requereix d'aquestes

Un cop omplertes aquestes dades només cal prémer el botó "*Guardar*" per guardar el connector i poder-lo utilitzar posteriorment en els *Jobs*.

10.1.4 Crear connector de sortida Elasticsearch

ManifoldCF disposa de un ampli conjunt de connectors de sortida. En el nostre projecte es va escollir el motor de cerca *Elasticsearch.* A continuació veurem les dades necessàries per configurar aquest tipus de connector.

Connector de sortida a *Elasticsearch*

- Nom del connector
- Tipus de connector (En aquest cas, "*Elasticsearch*")
- Número de connexions màxim
- Servidor on es troba *Elasticsearch*
- Paràmetres (nom de l'índex de destí, mapejat de paràmetres bàsics...)

Finalment, un cop acabada la configuració només cal prémer el botó "*Guardar*" per guardar el connector i poder utilitzar-lo en els *Jobs* posteriorment.

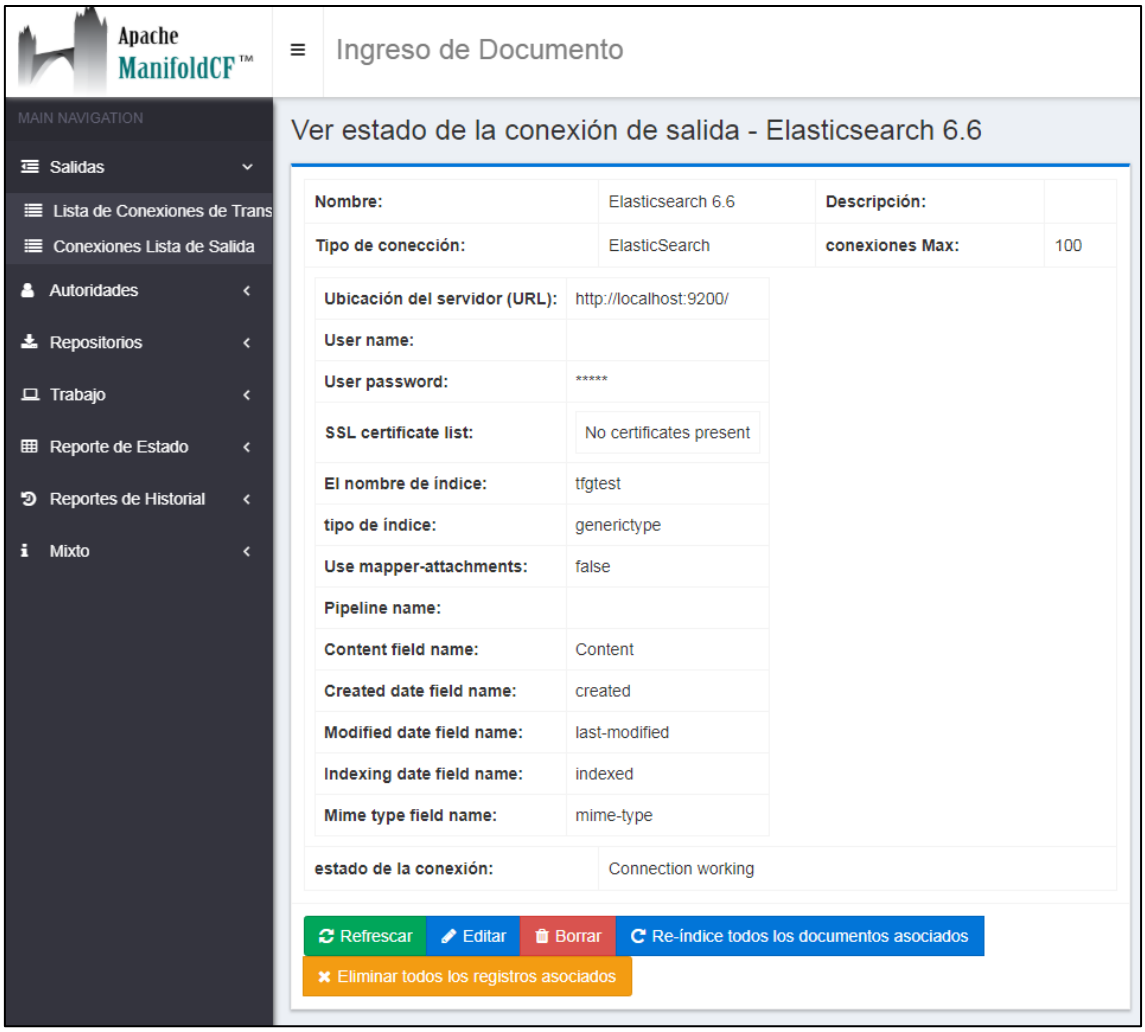


***Figura 15**. Pantalla de resum general del connector de sortida*

A aquesta pantalla "resum" podem veure l'estat de la connexió. Si ha pogut establir connexió amb *l'Elasticsearch* veurem que surt "*Connection working*".

Per més detall del procés de creació dels diferents connectors, veure Annex.

10.1.5 Crear Jobs

Un cop creats els diferents connectors, per poder utilitzar-los cal crear un Job que generi tot el procés pas a pas i configurar-lo per poder *Crawlejar* els documents desitjats. Les dades necessàries per generar aquest procés son les següents:

Procés a sistema de fitxers

- Nom del *Job*
- Connexions. Procés a seguir a través dels connectors generats anteriorment.
- Programació. Aquí es pot automatitzar el procés.
- Filtres de salt. Profunditat de *Crawleig* a la que es vol limitar.
- *Paths* dels repositoris. Tants com es desitgin *Crawlejar*.

Finalment cal prémer el botó “*Guardar*” per guardar el “*job*”.

Procés Web

- Nom del *Job*
- Connexions. Procés a seguir a través dels connectors generats anteriorment.
- Programació. Aquí es pot automatitzar el procés.
- Filtres de salt. Profunditat de *Crawleig* a la que es vol limitar.
- *Urls* de les webs a *Crawlejar*.
- Sessions obertes JSP/ASP/PHP...
- Mapejat de *Urls*
- Llistat d’inclusions
- Llistat d’exclusions
- *Token* de seguretat
- Mapejat de metadades
- Excloure excepcions de *Tika*
-

Finalment cal prémer el botó “*Guardar*” per guardar el “*job*”.

Per més detall del procés de creació dels *jobs*, veure Annex.

10.1.6 Llençar un job

Un cop creats els *jobs*, per poder llançar-los de manera manual (en cas de no haver-los programat per auto executar-se), cal anar des del menú principal a “*Trabajo*” → “*Estado y Gestión de Trabajos*” i allà veurem el llistat de *jobs* que s’hagin generat:

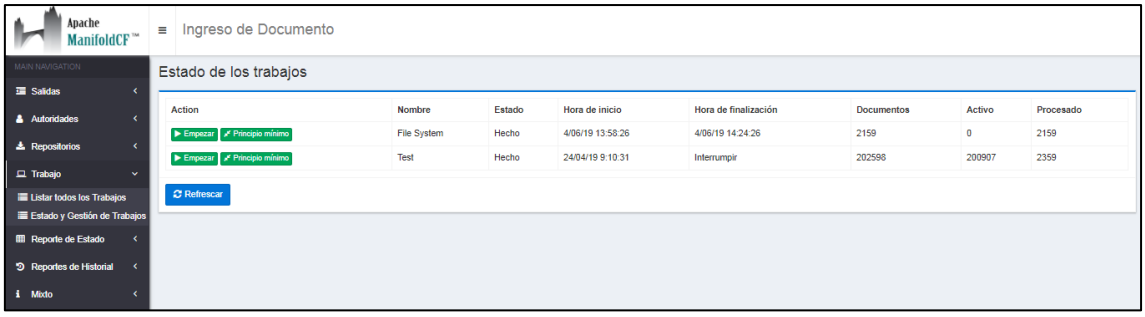


***Figura 16**. Pantalla d'estat dels jobs*

Per llançar el procés hi ha dos botons: “*Empezar*” i “*Principio mínimo*”. “*Empezar*” llença el procés utilitzant tots els recursos possibles de la màquina. “*Principio mínimo*” llença el procés utilitzant els recursos mínims per poder mantenir el procés.

A la dreta podrem veure quants documents ha detectat, quants links estan actius en cua, pendents de ser processats, i quants ha processat (indistintament de si els ha exclòs o inserit a l’índex).

10.2 Elasticsearch

Elasticsearch és l'eina escollida com a motor de cerca. En aquest punt explicarem com s'ha configurat aquest motor sobre el nostre prototip i diferents punts a tenir en compte de cara a qualsevol usuari final.

Les dades s'emmagatzemen en índex a dins *Elasticsearch*. Aquestes dades es reparteixen en *shards* a dins l'índex.

Un índex es pot generar manualment o automàticament. Quan s'incorpora un nou element a un índex, si aquest no existeix es genera automàticament amb les configuracions genèriques. És recomanable (i necessari en el nostre cas) definir unes configuracions més concretes. Per això definim una *template* amb aquestes configuracions. Una *template* és com un esquelet que aplicarà les configuracions definides quan es generi manual o automàticament un índex que compleixi amb el nom estipulat a dita *template*.

A continuació veurem les diferents parts que composen la *template* definida per el prototip explicant en detall des diferents configuracions escollides. La *template* sencera es pot trobar a l'Annex.

10.2.1 Configuració del índex

El primer fragment de la template correspon a la configuració de l'índex. Aquesta configuració consta dels següents parametres principals:

"**order**": Elasticsearch permet aplicar més d'una template a la creació d'un index. Aquest paràmetre defineix l'ordre en que s'aplicarien aquestes templates.

"**version**": Versió de la template en cas de voler sobreescriure-la.

"**template**": Nom de l'índex al que aplicarà la template quan es vagi a generar l'índex. Aquest paràmetre inclou expressions regulars com ara "*" per definir un conjunt de noms. Per exemple, "tfgtest*" inclouria tfgtest2, tfgtestexemple...

"**settings**": Subconjunt de configuracions que s'aplicaran a l'index.

"**index.refresh_interval**": Interval de refresc de l'índex. Internament *Elasticsearch* té dos buffers. Un on carrega les dades (escriptura) i l'altre on mostra a l'usuari les dades emmagatzemades. Aquest interval serveix per actualitzar les dades que mostra a l'usuari amb les noves dades introduïdes.

"**number_of_shards**": Numero de *shards*. *Elasticsearch* divideix la estructura de dades en *shards*. Els *shards* tenen un node màster i tants "esclaus" com la resta de

shards. El node màster és el que se'n carrega de gestionar la resta. Si el node màster cau, s'assigna automàticament un altre node com a màster. Això permet que el sistema tingui més robustesa i no caigui tan fàcilment.

"**number_of_replicas**": Número de rèpliques de les dades de cada *shard*. Cada *shard* replica les dades cap a altres *shards* tants com indiqui el paràmetre. Això permet que si cau un *shard*, les dades que tenia aquest continuen sent accessibles.

Configuració aplicada mitjançant la *template* del nostre prototip:

```
"order": 0,
"version": 0,
"template": "tfgtest",
"settings": {
   "index.refresh_interval": "1s",
   "number_of_shards": 3,
   "number_of_replicas": 1,
}
```

***Figura 17**. Configuració de l'índex d'Elasticsearch*

10.2.2 Filtres:

Els filtres son les configuracions que afectaran sobre les dades indexades. Cada tipus de filtre afecta de forma diferent un cop aplicat mitjançant un analitzador. A continuació explicarem què fan cadascun dels filtres definits a la nostre *template* utilitzada en el prototip.

"**char_filter**": Aquest filtre aplica sobre caràcters. També sobre paraules. Permet gestionar sinònims en cas que es vulguin establir i permet definir els caràcters especials que formen part del repertori reservat *d'Elasticsearch* com, per exemple, els símbols "+", "-", "=", etc. Per tal de poder utilitzar-los a les cerques, és necessari *mapejar* la codificació d'aquests caràcters com si fossin una paraula utilitzat aquest filtre com si fossin sinònims de la paraula assignada a cada caràcter.

"**english_stop**": *Stopwords* de l'idioma Anglès. Les *Stopwords* son les paraules que no es vol que es tinguin en compte a l'hora de buscar, tals com determinants, articles...

"**english_stemmer**": *Stemmer* en Anglès. Els *stemmers* son les arrels de les paraules per poder obtenir les diferents conjugacions al buscar. Per exemple en anglès, la paraula "*runner*", tindria l'arrel "*run*", per tant si un usuari busqués "*running*" també obtindria els resultats de "*run*", "*runner*"...

"**english_possessive_stemmer"**: Al igual que l'anterior, també permet obtenir les arrels de les paraules en Anglès, a diferencia de l'anterior, aquest permet obtenir les arrels dels possessius.

"**spanish_stop**": *Stopwords* de l'idioma Castellà. Les *Stopwords* son les paraules que no es vol que es tinguin en compte a l'hora de buscar, tals com determinants, articles...

"**spanish_stemmer**": Permet obtenir les arrels de les paraules en Castellà.

"**filter_shingle**": Un filtre que construeix *n-grames* a partir d'una seqüència de *tokens* (paraules en el nostre cas). Això permet buscar combinacions de les diferents paraules que composen una frase buscada.

"**my_ascii_folding**": Filtra els caràcters més enllà del 126 de la taula *Ascii*. Això permet que no faci distincions amb paraules que porten o no porten accents, "ç", etc.

A continuació mostrem el fragment de la *template* del nostre prototip corresponent a la configuració dels filtres comentats.

```
{
   "analysis": {
     "char_filter": {
       "basic_mapping": {
         "type": "mapping",
         "mappings": [
           "\\u0023 => hashtag",
           "\\u00A9 => copyright",
           "\\u002B => simbolomas",
           "\\u002A => simbolomultiplicacion",
           "\\u002D => simboloresta",
           "\\u002F => simbolodivision",
           "\\u003D => simboloigual",
           "\\u0040 => simboloarroba",
           "\\u0026 => simboloampersan",
           "\\u0028 => parentesisizquierda",
           "\\u0029 => parentesisderecha",
           "\\u005B => corcheteizquierdo",
           "\\u005D => corchetederecho",
           "\\u00BF => interroganteinicio",
           "\\u003F => interrogantefinal",
           "\\u00A1 => admiracioninicio",
           "\\u0021 => admiracionfinal",
           "\\u003C => menorque",
           "\\u003E => mayorque",
           "\\u007C => lineavertical",
           "\\u0025 => tantoporciento"
         ]
       }
     },
     "filter": {
       "english_stop": {
         "type": "stop",
         "stopwords": "_english_"
       },
       "english_stemmer": {
         "type": "stemmer",
         "language": "english"
       },
       "english_possessive_stemmer": {
```

```
        "type": "stemmer",
        "language": "possessive_english"
      },
      "spanish_stop": {
        "type": "stop",
        "stopwords": "_spanish_"
      },
      "spanish_stemmer": {
        "type": "stemmer",
        "language": "light_spanish"
      },
      "filter_shingle": {
        "type": "shingle",
        "min_shingle_size": 2,
        "max_shingle_size": 4,
        "filler_token": ""
      },
      "my_ascii_folding": {
        "type": "asciifolding",
        "preserve_original": "true"
      }
    }
```

***Figura 18**. Tabla de filtres d'Elasticsearch*

Per més informació sobre els diferents filtres que inclou Elasticsearch, veure els links:

- https://www.elastic.co/guide/en/elasticsearch/reference/6.6/analysis.html
- https://www.elastic.co/guide/en/elasticsearch/reference/6.6/analysis-tokenfilters.html
- https://www.elastic.co/guide/en/elasticsearch/reference/6.6/analysis-custom-analyzer.html

10.2.3 Analitzadors:

Els analitzadors són les configuracions que s'aplicaran als camps que es defineixin com "analitzats". Els analitzadors inclouen els filtres que s'han definit al punt anterior en funció del que es vulgui utilitzar en cada analitzador. Es poden tenir tants analitzadors com es vulgui i no cal que s'apliquin tots els definits sinó que l'usuari escollirà posteriorment quins camps vol analitzar concretament i amb quins analitzadors en específic.

Es important tenir en compte que els filtres aplicats a dins els analitzadors s'apliquen en l'ordre establert, això implica que en cas de conflicte sempre es tindrà en compte l'ordre establert.

Per exemple, si hi ha el filtre de caràcters (*char_filter*) que agafa sinònims d'una paraula i un filtre de *stopwords*, que filtra un conjunt de paraules com si no existissin a l'hora de realitzar la cerca (determinants, articles...). Si ens trobéssim el cas que entre els sinònims es definís una paraula composta com ara "Universitat Politècnica de Catalunya" amb un sinònim "UPC" i s'ha afegit el filtre de les *stopwords* primer; mai

agafaria el sinònim definit, ja que aquest inclou “de” que és definit com una *stopword* en castellà. Per tant internament *Elasticsearch* interpretaria la paraula composta com “Universitat Politècnica Catalunya” que no és exactament la mateixa cadena definida en el sinònim.

Per això és molt important tenir en compte l’ordre dels filtres afegits a l’analitzador.

En el nostre prototip s’han definit tres analitzadors. Un en anglès, un en castellà i un “bàsic” que no inclou filtres per idioma. A continuació es mostren els analitzadors definits a la *template*:

```
"analyzer": {
      "english": {
        "type": "custom",
        "tokenizer": "standard",
        "char_filter": ["basic_mapping"],
        "filter": [
          "lowercase",
          "my_ascii_folding",
          "english_stemmer",
          "english_stop",
          "english_possessive_stemmer",
          "filter_shingle"
        ]
      },
      "spanish": {
        "type": "custom",
        "tokenizer": "standard",
        "char_filter": ["basic_mapping"],
        "filter": [
          "lowercase",
          "my_ascii_folding",
          "spanish_stemmer",
          "spanish_stop",
          "filter_shingle"
        ]
      },
      "basic": {
        "tokenizer": "standard",
        "char_filter": ["basic_mapping"],
        "filter": [
          "lowercase",
          "my_ascii_folding",
          "filter_shingle"
        ]
      }
    }
}
```

***Figura 19**. Tabla d'analitzadors d'Elasticsearch*

10.2.4 Configuració de camps:

Un cop definits els filtres i els analitzadors, a continuació mostrem les metadades esperades que s'extrauran durant el procediment de *crawleig*, juntament amb els analitzadors que els hem assignat.

Les metadades que s'espera extreure son:

"**Content**": Contingut dels documents i les webs.

"**Keywords**": Paraules clau. Només aplica en el cas de webs, els fitxers no tenen aquesta metadada.

"**title**": Títol de les webs obtingudes.

"**dc:title**": Títol que s'ha pogut extreure dels documents *crawlejats* per *File System*.

Analitzadors aplicats: S'ha triplicat el contingut de tots aquests camps. Al text original, s'ha aplicat l'analitzador bàsic. S'ha guardat un subcamp "*cast*" amb el text analitzat amb els filtres en Castellà. S'ha guardat un segon subcamp "*eng*" amb el text analitzat amb els filtres en Anglès.

```
"Content": {
   "type": "text",
   "analyzer": "basic",
   "fields": {
      "cast": {
         "type": "text",
         "analyzer": "spanish"
      },
      "eng": {
          "type": "text",
          "analyzer": "english"
      }
   }
},
"Keywords": {
   "type": "text",
   "analyzer": "basic",
   "fields": {
      "cast": {
         "type": "text",
         "analyzer": "spanish"
      },
      "eng": {
         "type": "text",
         "analyzer": "english"
      }
   }
},
"dc:title": {
   "type": "text",
```

```
    "analyzer": "basic",
    "fields": {
        "cast": {
            "type": "text",
            "analyzer": "spanish"
        },
        "eng": {
            "type": "text",
            "analyzer": "english"
        }
    }
},
"title": {
    "type": "text",
    "analyzer": "basic",
    "fields": {
        "cast": {
            "type": "text",
            "analyzer": "spanish"
        },
        "eng": {
            "type": "text",
            "analyzer": "english"
        }
    }
}
```

***Figura 20**. Tabla de metadades d'Elasticsearch*

10.3 Servei de cerca

Com s'ha comentat a l'apartat d'arquitectura, aquest servei està desenvolupat com una aplicació *Java*. En aquest punt explicarem el detall tècnic que suposa aquesta tecnologia del projecte.

El servei de cerca està muntat sobre un *Tomcat* v8.30. El l'aplicatiu *Java* porta la llibreria de *SpringBoot* amb la que desplega un *endpoint* sobre el que rebrà les cerques.

Endpoint:
/search

Exemple de *endpoint* del nostre prototip muntat a local:
http://localhost:8080/buscador-1.0-SNAPSHOT/search

L'endpoint espera peticions *Http Post* amb els següents paràmetres:

Endpoint de cerca			
Nom	**Definició**	**Tipus**	**Valors acceptats**
query	Text que es vol buscar	String	Qualsevol caràcter
language	Llenguatge en el que es	String	es/en

	vol realitzar la cerca		(castellà/Anglès)
from	Primer element des del que es vol paginar	Integer	Valors positius incloent el 0
size	Número d'elements que es vol retornar	Integer	Valors positius

***Figura 21**. Tabla de paràmetres de l'endpoint de cerca*

Un cop es rep la petició aquesta s'envia al mòdul de cerca a on es validen les dades i es netegen en funció de:

- Accents
- Etiquetes *Xml*
- Formatejat de text en funció del motor de cerca

Els dos primers son genèrics per tots tipus de motors, ja que en concret, a *Elasticsearch* ja tenim "escapats" els accents, però podria no ser així en altres motors de cerca. El formatejat de text en canvi, està desenvolupat expressament per *Elasticsearch*, ja que extreu els "*" i espais a principi i final de text perquè no modifiquin els valors de cerca. El caràcter "*" és un caràcter reservat per *Elasticsearch* que s'utilitza com a *wildcard* (comodí) i en segons quin tipus de cerca pot donar error ja que no sempre l'accepta.

Un cop netejada l'entrada de dades, es genera la *query* que s'enviarà a *Elasticsearch*. Aquesta *query* està muntada en format *JSON* ja que *Elasticsearch* espera aquest tipus de format i els camps estan amb la sintaxi que requereix *Elasticsearch*.

La *query* que es genera és la següent:

```
{
  "_source":{
    "includes":["title", "resourceName", "_score", "DATA"]
  },
  "from": 0,
  "size": 10,
  "query":{
    "multi_match": {
      "query":  "text exemple de cerca",
      "fields":["Content.cast", "keywords.cast", "title.cast", "dc:title.cast"]
    }
  },
  "highlight": {
    "fields": {
      "Content.cast":{
        "fragment_size" : 200,
        "number_of_fragments": 2,
        "no_match_size": 400
      }
    }
  },
  "sort": [
    {
      "_score": {
```

```
        "order": "desc"
      }
    }
  ]
}
```

***Figura 22**. Query Elasticsearch*

Aquesta query pasa al connector *d'Elasticsearch* a on es genera la petició *HTTP Post* al *endpoint* que té desplegat *Elasticsearch* i es llança.

Endpoint Elasticsearch:
http://localhost:9200/tfgtest/_search

Sent *localhost* el *host* d'origen (en el cas del prototip el mateix pc local), 9200 el port a on es desplega *Elasticsearch* per defecte, *tfgtest* el nom del nostre índex de proves i *_search l'endpoint* de cerca sobre l'índex.

Elasticsearch retorna la resposta a la *query* també en format *JSON*. Aquesta resposta té la següent estructura:

```
{
  "took":46,
  "timed_out":false,
  "_shards":{
    "total":3,
    "successful":3,
    "skipped":0,
    "failed":0
  },
  "hits":{
    "total":1,
    "max_score":8.131701,
    "hits":[{
      "_index":"tfgtest",
      "_type":"generictype",
    "_id":"file:/C:/Users/rsorianc/Desktop/TFG/Uni/1º/Quatrimestre%200/CIA%20Q0/Treball/Schickard.ppt",
      "_score":8.131701,
      "_source":{
      "resourceName" : "Schickard.ppt",
      "highlight":{
        "Content.cast":[
          "Te haré en otra ocasión un diseño más cuidadoso de la máquina aritmética; en resumidas cuentas, mira lo siguiente: <em>aaa</em> son los botones de los cilindros verticales que llevan las cifras de la tabla de multiplicación"
        ]
      }
    }]
  }
}
```

***Figura 23**. Exemple de resposta d'Elasticsearch*

Un cop s'obté la resposta, aquesta conté algunes dades més de les que necessitem. Per tant, a l'hora de construir l'objecte de resposta del servei, primer es necessita passar per el *Parser* de dades.

El *Parser* de dades s'encarrega d'agafar els camps que es volen retornar i donar el format esperat per el *front-end*. Aquest format consta d'un llistat d'objectes que formen

cada element de la resposta de la cerca cap a l'usuari. Un cop *parsejades* les dades, aquest llistat es retorna al *front-end*.

10.4 Front-end

El *front-end* desenvolupat és un exemple de com hauria de ser aquest component. La idea del projecte és que cada client final es generi el seu propi *front-end* a mesura.

El prototip generat, s'ha fet amb *Html* i *JSP*. Aquest prototip s'ha inclòs a dins del servei *Java* i s'ha desplegat conjuntament sobre el mateix *Tomcat* 8.30. Però com ja hem comentat, això és el cas del prototip. Aquest component es pot extreure i generar amb altres llenguatges web com ara *PHP* i desplegar sobre un servidor extern.

El prototip dissenyat té el següent aspecte:

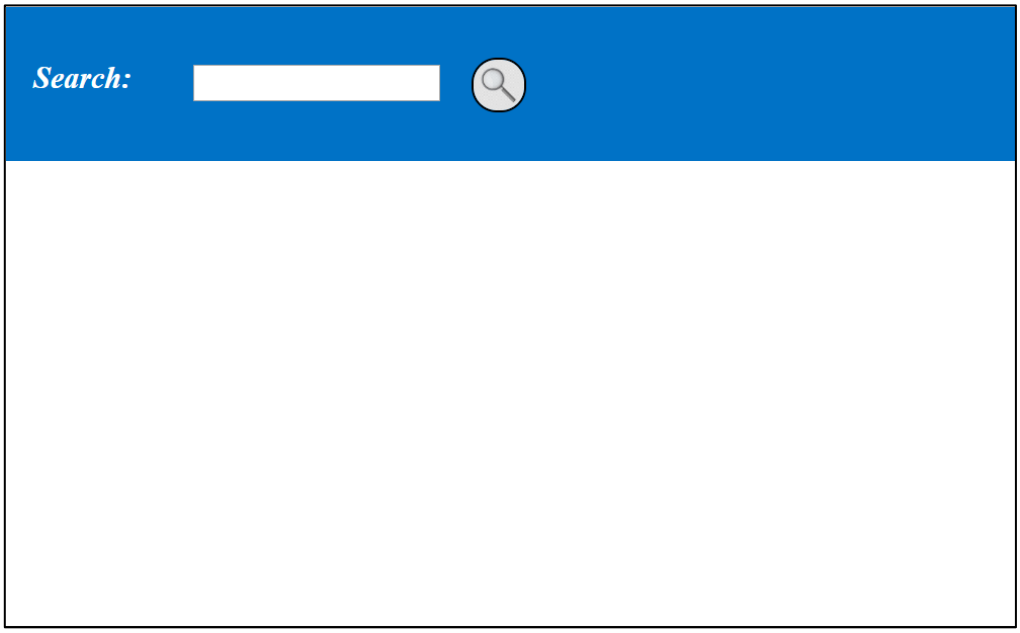


***Figura 24**. Pantalla principal del Cercador*

Internament el *front-end* té els paràmetres ocults:

Paràmetres front-end			
Nom	**Definició**	**Valors acceptats**	**Comentaris**
query	Text que es vol buscar	Qualsevol caràcter	
language	Llenguatge en el que es vol realitzar la cerca	es/en (castellà/Anglès)	En el prototip s'ha establert "es" per defecte. La idea seria incloure un selector d'idioma.
from	Primer element des del que es vol paginar	Valors positius incloent el 0	En el prototip s'ha establert valor 0 ja que no s'ha realitzat la paginació.
size	Número d'elements que es vol retornar	Valors positius	En el prototip s'ha establert valor 3000 per mostrar el màxim

			d'elements ja que no s'ha realitzat la paginació.

Figura 25. *Tabla de paràmetres del front-end*

En el projecte real el camp *from* s'hauria d'actualitzar un cop realitzada la cerca ja que s'hauria de paginar el resultat en funció d'un *size* més petit, per exemple, deu elements per pàgina.

S'hauria de definir també un selector d'idioma on es pogués escollir l'idioma en que es volgués realitzar la cerca i que aquest selector modifiqués el camp *language*.

Al realitzar la cerca l'usuari final escriurà el text en el camp de text i clicarà el botó en forma de lupa.

Internament el *front-end* genera una petició *Http Post* al servei de cerca enviant els camps comentats anteriorment a aquest servei.

Un cop s'obté la resposta del servei, el *front-end* pinta el llistat donant un format *Html* amigable.

Aquest format *Html* mostra els elements amb l'estructura següent:

- **Títol del document o web**. Títol clicable amb l'enllaç a la ruta del document o la web. En cas de document aquest es descarrega per poder ser visualitzat al PC del client.
- **Breu fragment de text.** Línies del document a on ha trobat el text buscat, marcant en negreta la part corresponent al text buscat.
- **Puntuació de la cerca**. Merament informatiu per ajudar visualment a veure quins resultats es consideren que s'adapten millor a la cerca. Aquesta puntuació la marca el motor de cerca *Elasticsearch*.

Exemple de cerca:

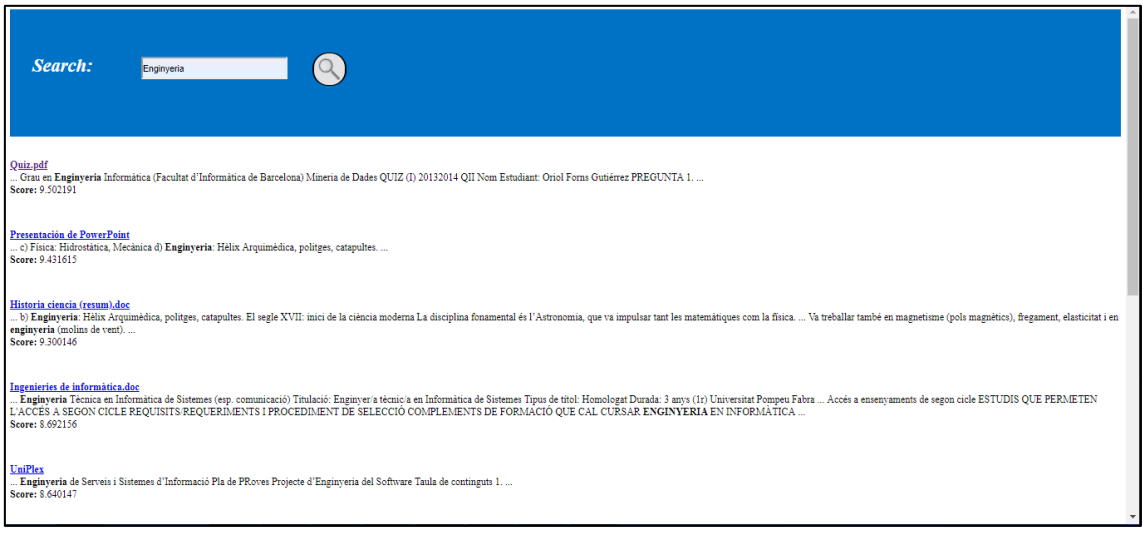


Figura 26*. Pantalla del Cercador amb resultats de cerca*

10.5 Regles de negoci

Aquesta part explica com s'introdueixen les regles de negoci. Què implica afegir-les i les modificacions que s'han de fer sobre el projecte i els documents per incloure-les.

10.5.1 Afegir metadades als documents

Els documents porten incorporades una sèrie de metadades en funció del tipus de document. Aquestes metadades acostumen a ser auto generades i normalment s'agafen el PC a on s'ha generat el document (Autor, data de creació, data de última modificació...). Cada tipus de document conté un conjunt de metadades diferent i en el cas de les webs d'entrada no les porten si no s'han escrit expressament.

Els documents permeten incloure noves metadades al llistat que ja porten per defecte. Per més detall i exemples veure Annex.

Un cop definides les metadades a afegir, a continuació veurem les modificacions necessàries per incloure-les a la resta de components i fer-les cercables.

10.5.2 Incloure noves metadades a Elasticsearch

Com ja s'ha comentat, el *Crawler* obté les metadades de forma automàtica. Això no les fa directament cercables. Com ja s'ha vist a la *template d'Elasticsearch*, aquesta porta les metadades amb els analitzadors incorporats. Així dons aquestes noves metadades que es vulguin afegir, s'han de definir a la *template* de l'índex abans de *crawlejar* els documents.

Per introduir noves metadades només cal agafar la *template d'Elasticsearch*, anar a la part de "*mappings*" i afegir la nova metadada juntament amb els analitzadors i subcamps que es desitgin aplicar.

Per exemple:

```
"mappings": {
   "generictype": {
    "_source": {
      "enabled": true
    },
    "_all": {
      "enabled": false
    },
    "properties": {
      "Content": {
        "type": "text",
        "analyzer": "basic",
        "fields": {
```

```
          "cast": {
            "type": "text",
            "analyzer": "spanish"
          },
          "eng": {
            "type": "text",
            "analyzer": "english"
          }
        }
      },
      "Keywords": {
        "type": "text",
        "analyzer": "basic",
        "fields": {
          "cast": {
            "type": "text",
            "analyzer": "spanish"
          },
          "eng": {
            "type": "text",
            "analyzer": "english"
          }
        }
      },
      "Assignatura": {
        "type": "text",
        "analyzer": "basic",
        "fields": {
          "cast": {
            "type": "text",
            "analyzer": "spanish"
          },
          "eng": {
            "type": "text",
            "analyzer": "english"
          }
        }
      },
...
```

***Figura 27**. Exemple inclussió noves metadades template Elasticsearch*

Continuant amb l'exemple anterior afegiríem el camp "Assignatura" i definiríem l'analitzador a aplicar sobre el camp bàsic i els subcamps per idioma. Aquests subcamps portarien els analitzadors corresponents a l'idioma escollit. (*****)

(*) Recordem que no cal definir subcamps amb idioma si així ho desitgem. És un exemple.

10.5.3 Modificacions sobre la query del servei

Actualment la *query* del servei inclou un conjunt de metadades sobre les que realitzar la cerca. Així dons s'haurien d'afegir les noves metadades a la cerca que s'enviarà a *Elasticsearch*.

Per exemple, si volguéssim afegir la metadada “Assignatura” comentada als anteriors exemples d’aquest punt, la *query* s’hauria de modificar de la següent manera:

Query:

```
{
  "query":{
    "multi_match": {
      “query”: “text exemple de cerca”
      “fields”:  ["Content.cast", “keywords.cast”, “title.cast”, “dc:title.cast”, “Assignatura.cast”]
    }
  },
  "highlight" : {
    "fields" : {
      "Content.cast": {
        "fragment_size" : 200,
        "number_of_fragments" : 2,
        "no_match_size": 400
      }
    }
  },
  "sort": [
    {
      "_score": {
        "order": "desc"
      }
    }
  ]
}
```

***Figura 28**. Exemple inclussió noves metadades a la query d'Elasticsearch*

10.5.4 Modificacions al front-end

El *front-end* no requereix de modificacions concretes ja que la *query* porta incorporades les metadades de la cerca. Així i tot, com a idea de millora es podria ficar un desplegable amb les metadades sobre les que es vol buscar si es volgués aplicar un filtre. Això implicaria modificar la *query* fent que a part dels camps que s’envien fins ara també s’enviés el filtre amb la metadada escollida per l’usuari (per exemple “Autor del document”).

El servei requeriria modificar la *query* afegint a *l’endpoint* de cerca aquest camp i generant la *query* amb el camp informat únicament.

Aquest seria només un exemple, ja que aquest és el camp d’estudi d’aquest projecte.

11. Procés d'implantació

En aquest punt explicarem el conjunt de passos necessaris a seguir de cara a realitzar la implantació d'aquest projecte a una organització o entitat pública final. També explicarem quin seria el rol més apropiat de la persona que hauria de realitzar cadascuna de les tasques d'implantació i els desenvolupaments finals necessaris a portar a terme en aquest procés.

Tasca 1:	Implantació del *Crawler*
Responsable:	Tècnic de sistemes
Requisits previs:	Un servidor on implantar el *Crawler*
Procés:	1. Descarregar el *Crawler* de la pàgina oficial de *ManifoldCF*: https://manifoldcf.apache.org/en_US/download.html 2. Modificar del fitxer *properties.xml* si es vol modificar la base de dades interna de *ManifoldCF* o alguna altre configuració. 3. Executar *ManifoldCF* des del fitxer *start.bat*

Tasca 2:	Implantació i configuració de l'indexador
Responsable:	Tècnic de sistemes
Requisits previs:	Un servidor on implantar l'indexador
Procés:	1. Descarregar *Elasticsearch* des de la pàgina oficial: https://www.elastic.co/es/downloads/elasticsearch 2. Executar *Elasticsearch* des del fitxer *elasticsearch.bat* 3. Si no es vol utilitzar la *template* genèrica d'aquest projecte, caldria generar la *template* amb les metadades que es vulguin explotar 4. Introduir la *template* generada a *Elasticsearch* mitjançant una crida *HTTP Put* a *l'endpoint*: http://hostElasticsearch:port/_template/nomTemplate

Tasca 3:	Configuració del Crawler i indexació del contingut
Responsable:	Tècnic de sistemes
Requisits previs:	Crawler implantat i indexador implantat i configurat
Procés:	1. Accedir al *ManifoldCF* implantat en la tasca 1. 2. Generar els connectors d'entrada de dades cap als diferents repositoris 3. Generar els connectors de sortida de dades cap a *Elasticsearch* 4. Generar el connector de transformació *Tika* 5. Generar el *Job* o *Jobs* en cas de múltiples entrades a dades. 6. Llençar els *Jobs* generats per indexar els diferents continguts
Nota:	Per més detall de com realitzar aquests passos veure l'Annex.

Tasca 4:	Implantació del Servei de cerca
Responsable:	Tècnic de sistemes
Requisits previs:	Un servidor a on implantar el servei de cerca
Procés:	1. Agafar el fitxer *buscador-1.0-SNAPSHOT.war* 2. Desplegar el *War* sobre un servidor (per exemple un *Tomcat* o un *Jetty*)

	El servei de cerca disposa d'un fitxer *War* amb el codi *ensamblat* i preparat per ser desplegat. Només cal desplegar aquest *War* sobre un servidor.

Tasca 5:	Disseny i implantació del frontal web
Responsable:	Tècnic de sistemes, dissenyador i programador
Requisits previs:	Servidor a on implantar el *front-end*
Procés:	1. Diseny del *front-end* web amb peticions *Http Post* al servei de cerca 2. Implementació del *front-end* amb el disseny esperat 3. Desplegar el *front-end* web sobre un servidor accessible des dels dispositius finals
Nota:	Aquesta tasca és opcional. El prototip porta integrat un *front-end* web, però la idea és que cada entitat generi el seu frontal adaptat al tipus de dispositiu final (monitors, tablets, mòbils...) i pugui afegir el logo de l'organització i metadades extres que vulgui mostrar.

Tasca 6:	Testeig de la integració completa
Responsable:	Tester
Requisits previs:	Projecte complert implantat i degudament configurat i un terminal d'usuari final.
Procés:	1. Accedir al *front-end* de l'aplicació 2. Realitzar una cerca que doni resultats 3. Accedir als resultats per comprovar que son accessibles

12. Treball futur

Actualment tenim desenvolupat un petit prototip que inclouria la major part del plantejament inicial del projecte. Tot i així, considerem que a part d'acabar el prototip, el projecte té molt potencial per ser explotat. Separaríem en tres els tipus de millores identificades a realitzar:

Reforçar arquitectura:

Es podria reforçar l'arquitectura actual plantejada. Això suposaria dos canvis sobre *ManifoldCF*. Un dels canvis seria aplicar una base de dades per substituir la *HSQL* interna que porta integrada. Això permetria un major rendiment un cop escalat el contingut indexable. També disminueix l'espai en disc ja que la *HSQL* no escala bé.

Apache (creador de *ManifoldCF*) recomana utilitzar *PostgreSql* com a base de dades, però *ManifoldCF* accepta també *MySql*.

L'altre canvi sobre l'arquitectura seria extreure el servidor intern que porta *ManifoldCF*. *ManifoldCF* porta incorporat un *Jetty*, que no es pot configurar. Com a aplicació es podria extreure i muntar com a *Jar* o *War* i desplegar sobre un altre servidor completament configurable com ara un *Tomcat*.

Noves funcionalitats:

Hem identificat un conjunt de funcionalitats que ara no formen part d'aquest projecte, que trobem que podrien ser interessants de cara al futur.

Incorporació d'un auto completar. Això permetria als usuaris finals poder realitzar cerques més ràpidament. D'altra banda també permetria suggerir opcions als usuaris en cas de no tenir clara la cerca a realitzar. Per introduir aquesta funcionalitat, s'ha pensat que es podria incloure un nou índex *d'Elasticsearch* a on, mitjançant el servei actual de cerca, vagi introduint les noves cerques i incrementant un comptador intern de les cerques que no siguin noves. D'altra banda es requeriria mitjançant *Ajax* sobre el frontal web que s'ataqués, per cada caràcter introduïda per un usuari, a un nou *endpoint* que retornés un nombre limitat de coincidències sobre les cerques amb comptador més alt.

Incorporació d'un servei d'estadístiques d'ús. Això permetria monitoritzar les cerques més realitzades, el volum de peticions en funció d'un rang de temps... etc. Per realitzar aquest servei, caldria emmagatzemar aquestes estadístiques a una base de dades o sobre un nou índex a *Elasticsearch* si es vol seguir utilitzant com a BBDD. També requeriria de la incorporació d'un frontal que permeti visualitzar aquestes estadístiques on es podrien incorporar gràfiques i llistats per facilitar la revisió d'aquestes.

Ampliació de la *template* actual *d'Elasticsearch*. Actualment les cerques es realitzen sobre un conjunt molt reduït de metadades. Un increment de metadades analitzades

permetria realitzar cerques més complexes que podrien aportar molt valor als usuaris finals.

Ampliar funcionalitats existents:

Es podria ampliar el prototip realitzat durant els mesos que ha durat el Treball de Final de Grau. Separaríem en quatre aquestes ampliacions o modificacions que podrien aportar valor.

Afegir un sistema de introducció massiva de metadades. Actualment el sistema de introducció de metadades addicionals per les regles de negoci, es totalment manual i això el fa totalment ineficient. Es podria buscar o desenvolupar un software que permeti realitzar aquest procés de forma massiva o almenys de manera més eficient.

Ampliació amb nous connectors d'entrada a repositoris. Actualment el prototip incorpora dos connectors d'entrada, el connector Web i el connector de Sistema de Fitxers. Es podria ampliar incorporant altres connexions a repositoris, ja que *ManifoldCF* porta incorporats molts connectors d'entrada (mencionats en aquest mateix document). També es podrien generar nous connectors que no estiguin incorporats i incorporar-los. *ManifoldCF* es *opensource*, cosa que permet aquesta incorporació sense gaires complicacions.

Afegir al *front-end* el selector d'idioma. Actualment el prototip ja té preparat el servei per incloure un idioma entre els paràmetres d'entrada. A més l'índex *d'Elasticsearch* ja porta incorporat els analitzadors en Anglès. Caldria també *crawlejar* continguts en els idiomes escollits i modificar la *template d'Elasticsearch* si l'idioma escollit no es troba configurat.

Afegir al *front-end* un *paginador* d'elements. Actualment el servei ja porta incorporada aquesta paginació, però el *front-end* no pagina degut a la limitació per calendari, que ens va suposar haver de prescindir d'aquesta funcionalitat.

13. El projecte en l'especialitat d'Enginyeria del Software

L'especialitat que he cursat correspon a l'Enginyeria del *Software*. Considero que aquest projecte s'adequa perfectament a aquesta especialitat en diferents aspectes.

Per una banda aquest projecte consisteix en especificar un sistema *software* des de zero per poder satisfer les necessitats d'un client (en aquest cas qualsevol institució que vulgui incorporar un sistema flexible de cerca en múltiples repositoris), implementar un prototip i avaluar els resultats.

A més el projecte requereix de diversos rols com per exemple, analista, arquitecte, programador, cap de projecte, etc. Tots corresponents als rols apresos durant l'especialitat.

També s'ha de poder ser capaç de avaluar diferents opcions de forma crítica. No només s'han d'avaluar les decisions sinó justificar-les i poder argumentar el perquè de les decisions d'implementació escollides i poder posteriorment avaluar els resultats. S'ha de poder portar a terme un sistema *software* que compleixi uns certs requisits de qualitat, però també que sigui escalable, portable i òptim.

A part, el disseny i gestió de serveis (i més concretament serveis web) només s'ensenyen a l'especialitat d'enginyeria del *software*.

Per tant, crec que tots aquests motius justifiquen el considerar aquest projecte com a un projecte enfocat a l'especialitat d'enginyeria del *software*.

13.1 Assignatures relacionades amb el projecte

Les assignatures de l'especialitat que fins ara he trobat més enfocades a ajudar-me amb el projecte, són:

ASW: En aquesta assignatura es toca molt el tema de serveis web, connexions remotes, etc. Crec que és directament la més relacionada amb el projecte que he cursat. Sobretot per veure la part pràctica. També s'aprèn una mica el funcionament de servidors (en concret del *Tomcat*). A part també s'explica com dissenyar programes (o projectes) juntament amb accessos a serveis.

AS: Aquesta assignatura ens ensenya a dissenyar completament un projecte. A fer-lo òptim, portable, canviable, etc. És l'assignatura base a cursar per poder dissenyar una aplicació ben feta. També es toca el tema d'accessos a serveis tot i que no tan a fons com a ASW. Veiem com dissenyar patrons de disseny i les decisions que hem de prendre a l'hora d'enfocar el nostre disseny d'aplicació. L'habilitat, escalabilitat i qualitat del sistema dissenyat recau directament sobre tots els coneixements apresos en aquesta assignatura.

DBD: Sobre aquesta assignatura també hi ajuntaria BD, tot i ser comuna i no directament de l'especialitat, però si correspon al mateix departament. Ambdues estan molt relacionades amb el projecte per l'enfocament sobre la gestió de dades que tindrà el nostre projecte. Considero que el disseny de l'índex de *Elasticsearch* (com a "base

de dades no relacional") ve directament influenciat per tot l'aprés a BD i amb les optimitzacions i dissenys apresos a DBD. Un bon disseny ens permet fer accessos més ràpids a les dades i per tant agilitzar molt el sistema. També ens ajuda a fer-lo portable en cas que es vulgui realitzar o reutilitzar aquesta "base de dades no relacional" de cara a altres projectes similars.

GPS i **PES**: Totes dues assignatures ens ensenyen a planificar i enfocar tota la part relacionada amb GEP. Gràcies a aquestes dues assignatures he pogut realitzar la planificació del projecte i enfocar-lo adequadament sense requerir gairebé informació de tot el material de suport aportat a GEP. Crec que aquestes dues assignatures s'haurien de donar també a les altres especialitats, ja que son claus per portar a terme qualsevol projecte independentment de l'especialitat en concret que es realitzi.

ER: Una de les parts del projecte, recau en què s'ha de fer. És a dir, quins punts entren directament al projecte en concret i què es considera un suplement o complement al projecte. És a dir, enfocar quins son els requisits del nostre projecte (una tasca realment difícil) l'he aprés d'aquesta assignatura.

14. Conclusions

En aquest apartat es comentarà el grau en que s'han complert les competències marcades inicialment i també les conclusions extretes sobre el projecte realitzat.

14.1 Justificació de les competències

En el nostre projecte estaran associades un total de cinc competències tècniques:

- **CES1.1:** Desenvolupar, mantenir i avaluar sistemes i serveis software complexos i/o crítics. [Bastant]

El projecte basa la cerca sobre les dades mitjançant un servei *software*. Per tant, la pròpia realització del software de cerca sobre *Elasticsearch* assoleix la competència. El nivell d'assoliment és degut a que gran part del software es basava en desenvolupar aquest servei des de 0.

- **CES1.2:** Donar solució a problemes d'integració en funció de les estratègies, dels estàndards i de les tecnologies disponibles. [Bastant]

La idea del projecte era precisament integrar un conjunt de softwares per solucionar una problemàtica actual. Considerem que aquesta competència s'ha resolt per si sola al complir l'objectiu del projecte de realitzar un cercador flexible sobre múltiples repositoris.

- **CES1.3:** Identificar, avaluar i gestionar els riscos potencials associats a la construcció de *software* que es poguessin presentar. [En profunditat]

Un *software* sobre sistemes de cerca en múltiples repositoris requereix d'un disseny i implementació molt robust i amb molta seguretat. Considerem que s'han de considerar molts riscos i possibles escenaris a tenir en compte de cara a aquest *software*, per tant s'ha realitzat un estudi d'aquests riscos amb profunditat i força cura per evitar que se'ns escapés cap possible risc.

- **CES1.7:** Controlar la qualitat i dissenyar proves en la producció de software. [Una mica]

Totes les integracions que ens ha suposat realitzar aquest projecte han sigut degudament testejades per tal d'evitar qualsevol problema a l'hora d'intentar fer funcionar el *end-to-end* de la plataforma de cerca dissenyada. A més, a l'especificació de requisits es van establir un conjunt de requisits de qualitat que s'han acabat complint. Per això, considerem que al haver complert amb els requisits, també hem complert amb aquesta competència.

CES2: Valorar les necessitats del client i especificar els requisits software per a satisfer aquestes necessitats, reconciliant objectius en conflicte mitjançant la cerca de compromisos acceptables, dintre de les limitacions derivades del cost, del temps, de l'existència de sistemes ja desenvolupats i de les organitzacions.

- **CES2.1:** Definir i gestionar els requisits d'un sistema *software*. [Bastant]

No és fàcil saber què forma part del sistema i què el complementa. Aquest projecte té moltes possibilitats, i es va considerar totalment necessari definir quins eren els

requisits. Durant la part de GEP es van establir unes bases, però a la memòria final s'han consolidat quins son concretament i en profunditat els requisits del nostre sistema. El nivell escollit es deu a la necessitat d'establir les fronteres sobre què forma part del projecte.

14.2 Conclusions del projecte

S'ha creat un prototip d'un sistema de cerca en múltiples repositoris destinat a qualsevol tipus d'entitat que disposi de documents sobre els que vulgui poder realitzar cerques del seu contingut.

A més a més, aquest sistema permet actualitzar continguts de forma automàtica gràcies a la programació automàtica que ofereix el sistema de *crawlejat* sense haver de modificar res del projecte més enllà de la programació automàtica del procés de *crawlejat*.

Hem vist també que incloure nous tipus de repositoris en aquest sistema és quelcom simple. No requereix d'estudis previs ni de programació avançada, sinó senzillament de seguir i emplenar un conjunt de camps per muntar un nou procés de *crawleig*.

D'altra banda també hem vist que amb una arquitectura senzilla com és la del portàtil usat per realitzar el prototip es pot desplegar i mantenir tot el sistema; tot i que segons la volumetria de documents i la càrrega dels usuaris finals, el projecte real requeriria una arquitectura més potent.

Gràcies als *softwares* usats, aquest projecte és fàcilment escalable i canviable si es desitja modificar o canviar part del software utilitzat.

També hem pogut monitoritzar els temps de resposta de l'aplicatiu (ja que des del front-end es mostra el temps de cerca) i aquests són força positius, ja que compleixen amb el requisit establert en un inici de projecte (inferior a 0,5 Segons) per els 10 elements resultants de paginació. En aquest tipus de projecte, els temps de resposta son crucials ja que de cara a l'usuari final és el primer que es nota al utilitzar el cercador.

Tot i així, no tot es positiu. Hem vist també que realitzar les regles de negoci porta molt treball per part dels usuaris finals; ja que d'entrada els documents no solen portar les metadades extres sobre les que realitzar les regles, sinó que s'han d'afegir una a una a tots els documents. A més requereix de coneixements mínims *d'Elasticsearch* per modificar la *query*, el servei de cerca i el *front-end* si es vol mostrar un conjunt de possibilitats extres de cerca a l'usuari final.

Per tant, podem concloure que com a sistema de *crawleig*, centralització de documents i cerques sobre aquests, el projecte compleix de sobres les expectatives inicials. Però com a gestor de cerca de negoci (regles afegides de negoci) suposa un esforç sobre la preparació inicial que potser no compensa amb el valor afegit que suposen aquestes regles de negoci.

També podem veure que té molt potencial de cara futurs desenvolupaments i millores, ja que, el prototip se'ns ha quedat curt degut a la limitació per calendari i no s'ha pogut desenvolupar tot el que es pretenia desenvolupar des d'un inici; Principalment les regles de negoci i un *front-end* més treballat incloent-li un *paginador* d'elements, un comptador d'elements totals...

14.3 Conclusions personals

Gràcies al projecte he pogut construir des de zero una arquitectura d'un sistema flexible de cerca sobre múltiples repositoris. He pogut comprovar la gran complexitat que suposa aquest tipus de software. També he pogut créixer sent capaç d'analitzar l'impacte i els costos d'un Software d'aquest nivell abans de realitzar-lo.

Amb les dates establertes per el calendari he pogut també aprendre que la construcció d'un nou Software no es tan senzill com sembla un cop ficat sobre paper i que sempre sorgeixen imprevistos que no havíem pogut controlar prèviament per sobre dels que ja esperàvem. Tampoc ha resultat gens fàcil extreure els requisits finals que havia de tenir el projecte, tant com a projecte de Software com de cara a complir amb les competències amb les que ens havíem compromès.

He pogut créixer també de cara a les funcions que requereix un bon analista i aprendre del treball i responsabilitat que suposa ser cap d'un projecte.

Considero personalment que la part més important apresa és sobretot la part de gestió més que sobre el desenvolupament en sí del prototip.

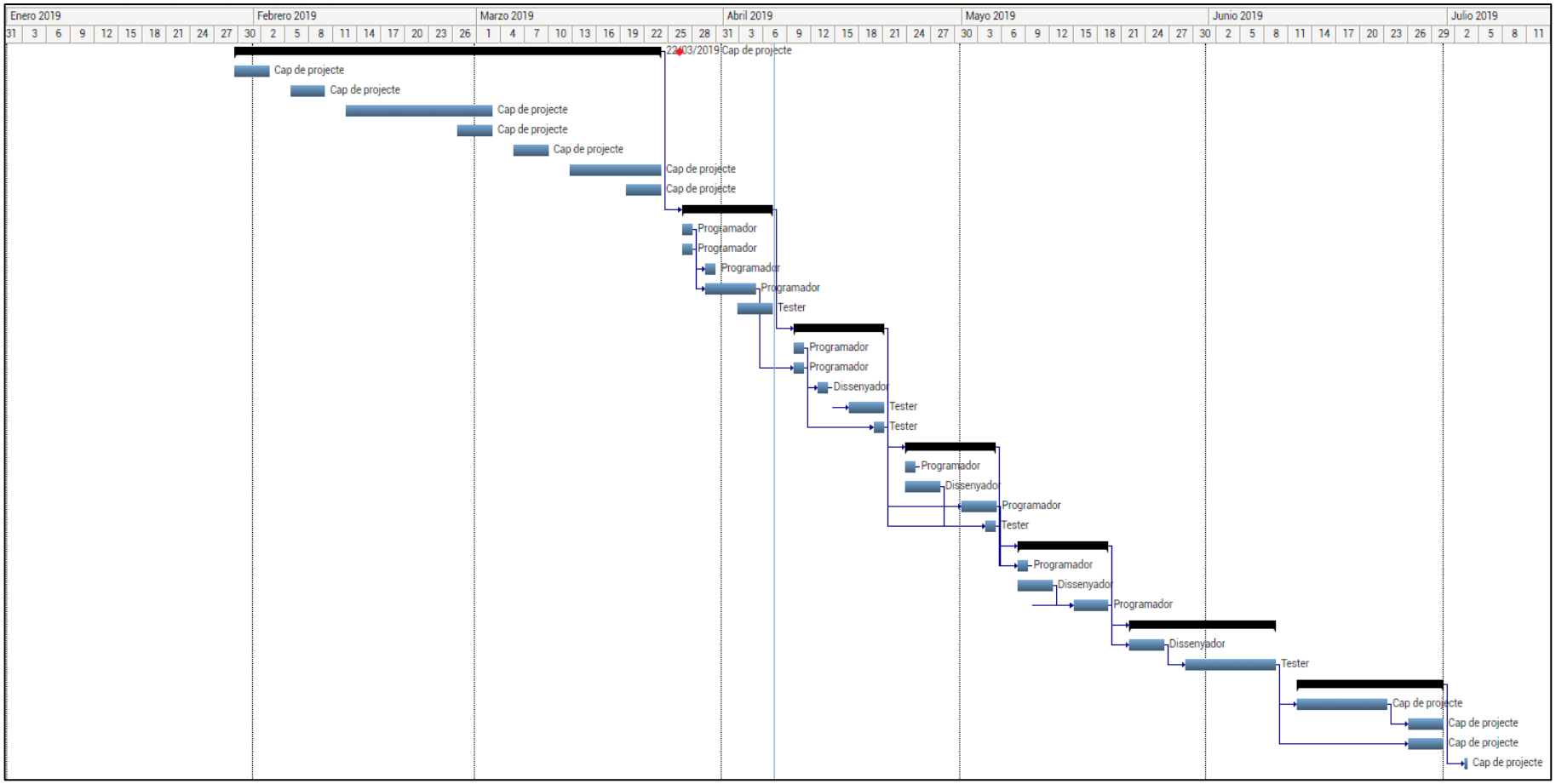
Finalment considero que el que m'ha fet evolucionar ha sigut el canvi de vista en funció d'analista i deixar l'enfocament únic que tenia abans com a desenvolupador.

15. Webgrafia

- Lucene
  - https://es.wikipedia.org/wiki/Lucene
  - http://lucene.apache.org/
- Elasticsearch
  - https://en.wikipedia.org/wiki/Elasticsearch
  - https://www.elastic.co/
- ManifoldCF
  - https://manifoldcf.apache.org/
- Solr
  - http://lucene.apache.org/solr/features.html
- Diskover
  - https://fernandogr.net/fgrblog/diskover-file-system-crawler-storage-search-engine-and-analytics-powered-by-elasticsearch-kitploit/
- GoogleBot
  - https://es.wikipedia.org/wiki/Googlebot
  - https://support.google.com/webmasters/answer/182072?hl=en
- Wikipedia
  - https://es.wikipedia.org
- Jakob Nielsen
  - https://es.semrush.com/blog/usabilidad-web-principios-jakob-nielsen/
- Web Browsers
  - https://www.muycomputer.com/2018/05/03/chrome-dominando-navegadores/
- Java
  - https://www.java.com/es/
- Google Search Apliance (GSA)
  - https://enterprise.google.com/search/
  - https://support.google.com/gsa/answer/7528111?hl=en
- Pico Search
  - https://libguides.sdsu.edu/c.php?g=666361&p=4686833
- Software biblioteques
  - https://www.collectorz.com/book
  - https://www.libib.com/
  - https://www.librarything.com/
  - http://www.spacejock.com/BookDB_Download.html
- ScanFS
  - https://www.saleensoftware.com/ScanFs.aspx
- MySQL
  - https://www.mysql.com/
- Alexa crawlers
  - https://support.alexa.com/hc/en-us/articles/200450194-Alexa-s-Web-and-Site-Audit-Crawlers

Annex:

1. Diagrama de Gantt



2. Template Elasticsearch


```
{
  "order": 0,
  "version": 0,
  "template": "tfgtest",
  "settings": {
    "index.refresh_interval": "1s",
    "number_of_shards": 3,
    "number_of_replicas": 1,
    "analysis": {
      "char_filter": {
        "basic_mapping": {
          "type": "mapping",
          "mappings": [
            "\\u0023 => hashtag",
            "\\u00A9 => copyright",
            "\\u002B => simbolomas",
            "\\u002A => simbolomultiplicacion",
            "\\u002D => simboloresta",
            "\\u002F => simbolodivision",
            "\\u003D => simboloigual",
            "\\u0040 => simboloarroba",
            "\\u0026 => simboloampersan",
            "\\u0028 => parentesisizquierda",
            "\\u0029 => parentesisderecha",
            "\\u005B => corcheteizquierdo",
            "\\u005D => corchetederecho",
            "\\u00BF => interroganteinicio",
            "\\u003F => interrogantefinal",
            "\\u00A1 => admiracioninicio",
            "\\u0021 => admiracionfinal",
            "\\u003C => menorque",
            "\\u003E => mayorque",
            "\\u007C => lineavertical",
            "\\u0025 => tantoporciento"
          ]
        }
      },
      "filter": {
        "english_stop": {
          "type": "stop",
          "stopwords": "_english_"
        },
        "english_stemmer": {
          "type": "stemmer",
          "language": "english"
        },
        "english_possessive_stemmer": {
          "type": "stemmer",
          "language": "possessive_english"
        },
        "spanish_stop": {
          "type": "stop",
          "stopwords": "_spanish_"
        },
        "spanish_stemmer": {
          "type": "stemmer",
          "language": "light_spanish"
```

```
      },
      "filter_shingle": {
        "type": "shingle",
        "min_shingle_size": 2,
        "max_shingle_size": 4,
        "filler_token": ""
      },
      "my_ascii_folding": {
        "type": "asciifolding",
        "preserve_original": "true"
      }
    },
    "analyzer": {
      "english": {
        "type": "custom",
        "tokenizer": "standard",
        "char_filter": ["basic_mapping"],
        "filter": ["lowercase", "my_ascii_folding",  "english_stemmer", "english_stop",
          "english_possessive_stemmer","filter_shingle"]
      },
      "spanish": {
        "type": "custom",
        "tokenizer": "standard",
        "char_filter": ["basic_mapping"],
        "filter": ["lowercase", "my_ascii_folding", "spanish_stemmer", "spanish_stop",
          "filter_shingle"]
      },
      "basic": {
        "tokenizer": "standard",
        "char_filter": ["basic_mapping"],
        "filter": ["lowercase", "my_ascii_folding", "filter_shingle"]
      }
    }
  }
},
"mappings": {
  "genrictype": {
    "_source": {
      "enabled": true
    },
    "_all": {
      "enabled": false
    },
    "properties": {
      "Content": {
        "type": "text",
        "analyzer": "basic",
        "fields": {
          "cast": {
            "type": "text",
            "analyzer": "spanish"
          },
          "eng": {
            "type": "text",
            "analyzer": "english"
          }
        }
      },
      "Keywords": {
        "type": "text",
```

```
          "analyzer": "basic",
          "fields": {
            "cast": {
              "type": "text",
              "analyzer": "spanish"
            },
            "eng": {
              "type": "text",
              "analyzer": "english"
            }
          }
        },
        "dc:title": {
          "type": "text",
          "analyzer": "basic",
          "fields": {
            "cast": {
              "type": "text",
              "analyzer": "spanish"
            },
            "eng": {
              "type": "text",
              "analyzer": "english"
            }
          }
        },
        "title": {
          "type": "text",
          "analyzer": "basic",
          "fields": {
            "cast": {
              "type": "text",
              "analyzer": "spanish"
            },
            "eng": {
              "type": "text",
              "analyzer": "english"
            }
          }
        }
      }
    }
  }
}
```

3. Manual de configuració de ManifoldCF

Crear connector entrada File System

En aquest punt explicarem com hem creat el connector d'entrada a Sistema de Fitxers del nostre prototip i punts a tenir en compte de cara als futurs usuaris de l'eina.

Un cop arrancat *ManifoldCF*, al llistat de la dreta podem veure les diferents categories a la que se'ns permet accedir i gestionar. En el cas de creació de connectors a repositoris d'entrada, s'ha d'anar a "*Repositorios*" → "*Lista de conexiones del Repositorio*" i prémer el botó "*Nuevo Repositorio*".

Es mostrarà la següent pantalla:

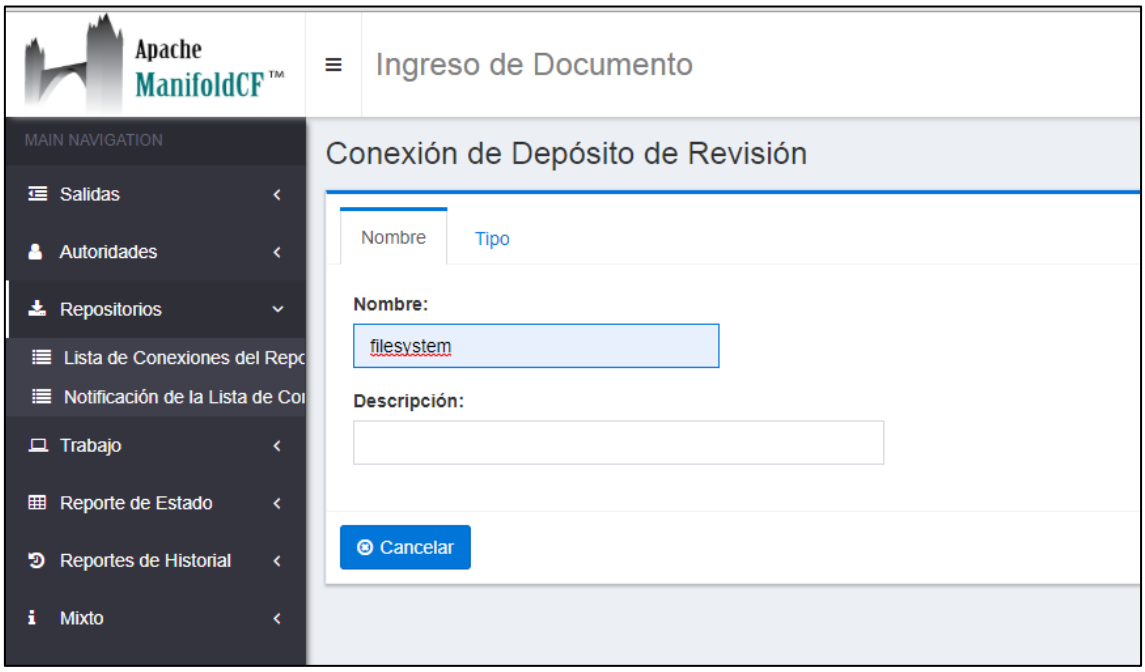


***Figura 29**. Pantalla inicial de creació de connector*

En aquesta pantalla escollim un nom que identificarà el connector d'entrada. No cal que sigui únic però és recomanable no duplicar noms ja que després a l'hora de seleccionar el connector quan creem el *job*, no podrem distingir entre dos noms iguals.

Opcionalment es pot afegir una descripció al connector.

Finalment canviem a la pestanya "*Tipo*" per escollir el tipus de connector.

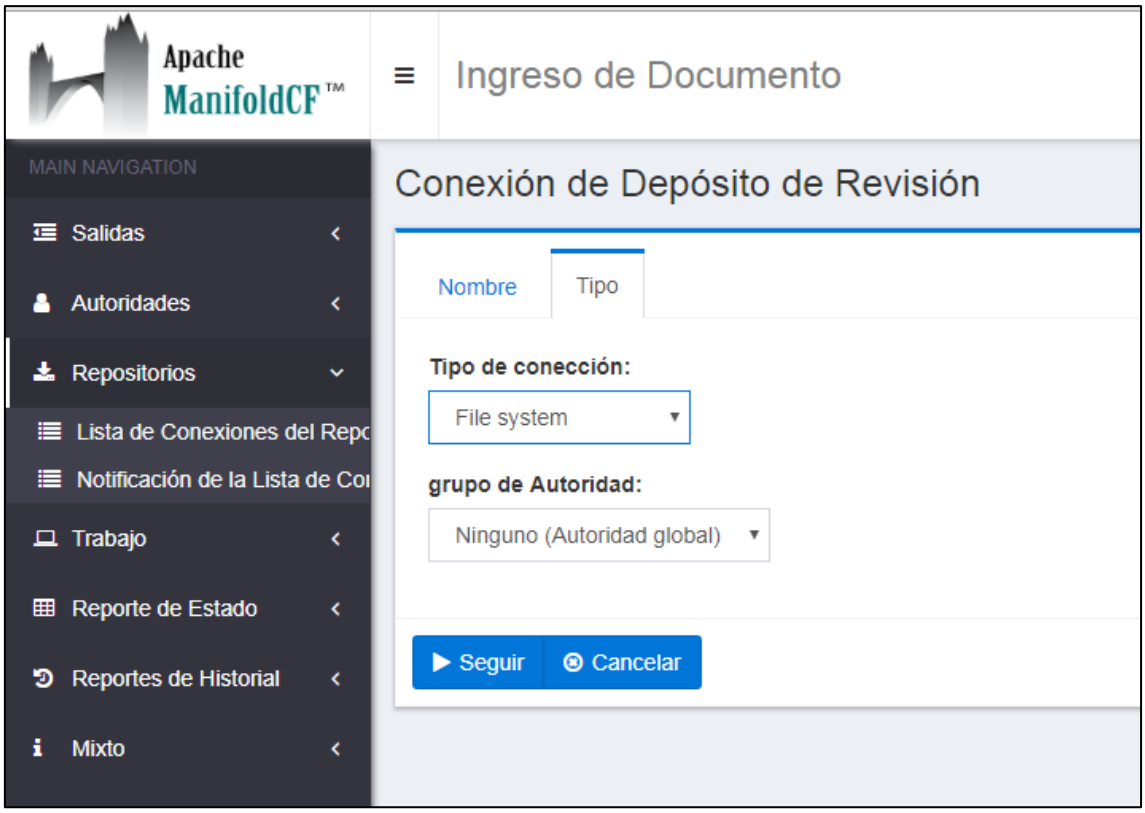


***Figura 30**. Pantalla selector de tipus de connector*

El tipus escollit és File System, ja que per al prototip necessitàvem aquest tipus de connector de cara als fitxers. En quan al grup d'autoritat, no s'ha escollit cap.
En aquest punt cal clicar el botó "*Seguir*" per obtenir la nova pestanya "*Acelerando*". Aquest botó fa que el tipus de connector quedi fixe (no es pot modificar). La resta de paràmetres, segueixen sent editables.

Finalment la pestanya "*Acelerando*":

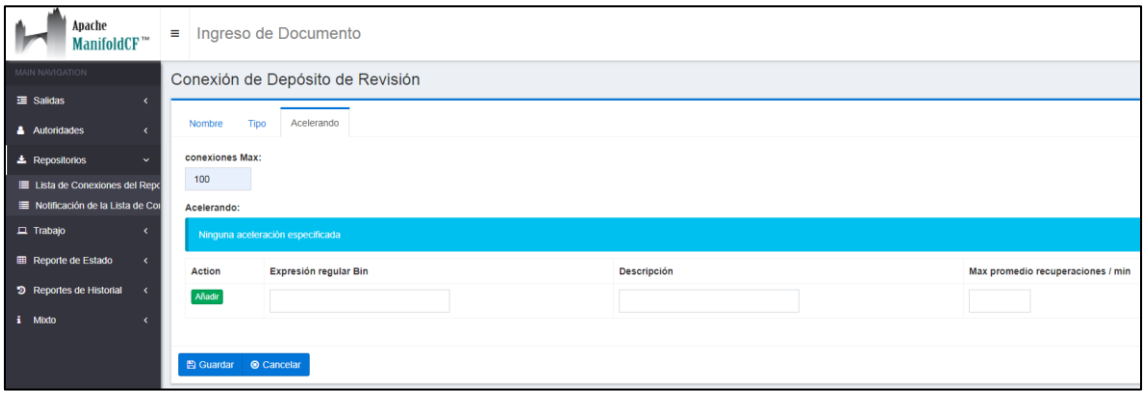


***Figura 31**. Pantalla número de connexions*

Aquí es poden escollir el número de connexions (Hem establert a 100 concurrents en el nostre prototip). A més a més, si es vol accelerar algun fitxer o carpeta en cas del *File System*, es pot afegir una expressió regular per cada element o grup d'elements que es vulgui accelerar o reduir connexions específicament.

Finalment, un cop acabada la configuració només cal prémer el botó "*Guardar*" per guardar el connector i poder utilitzar-lo en els *Jobs* posteriorment.

Crear connector entrada web

En aquest punt explicarem com hem creat el connector d'entrada cap a Web del nostre prototip i punts a tenir en compte de cara als futurs usuaris de l'eina.

Al igual que amb el connector d'entrada de Sistema de Fitxers, cal accedir a la llista de repositoris d'entrada "*Repositorios*" → "*Lista de conexiones del Repositorio*" i prémer el botó "*Nuevo Repositorio*".

Es mostrarà la següent pantalla:

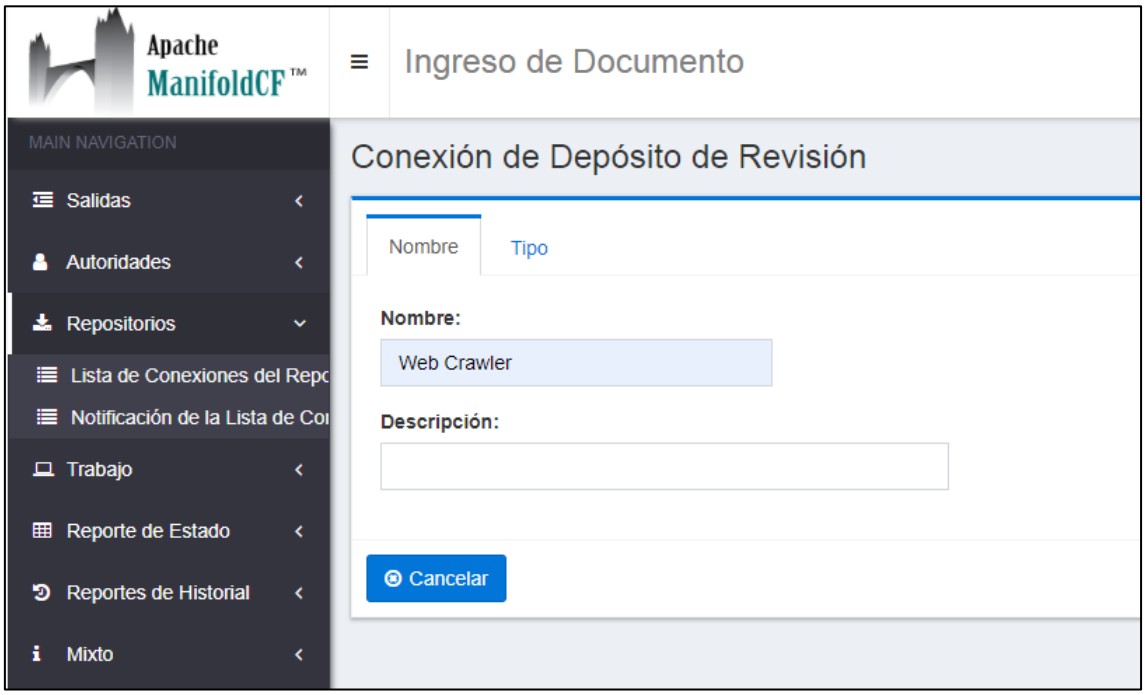


***Figura 32**. Pantalla inicial de creació de connector*

En aquesta pantalla escollim un nom que identificarà el connector d'entrada. Al igual que amb el de File System, no cal que sigui únic però és recomanable no duplicar noms ja que després a l'hora de seleccionar el connector quan creem el *job*, no podrem distingir entre dos noms iguals.

Opcionalment es pot afegir una descripció al connector.

Finalment canviem a la pestanya "*Tipo*" per escollir el tipus de connector.

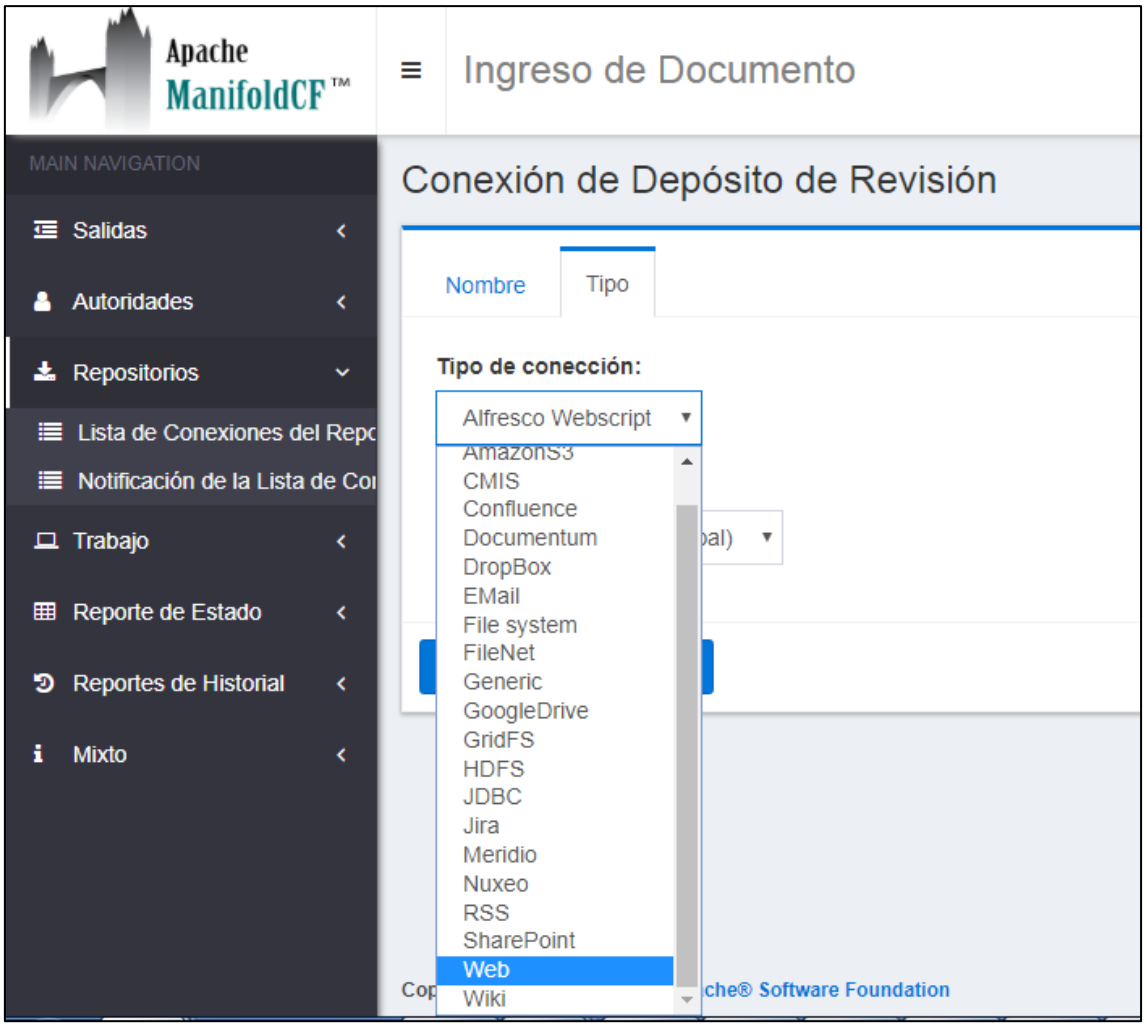


Figura 33. *Pantalla selector de tipus de connector*

El tipus escollit és Web, ja que per al prototip necessitàvem aquest tipus de connector de cara a les pàgines web a obtenir. En quan al grup d'autoritat, no s'ha escollit cap.
En aquest punt cal clicar el botó "*Seguir*" per obtenir noves pestanyes. Aquest botó fa que el tipus de connector quedi fixe (no es pot modificar). La resta de paràmetres, segueixen sent editables.

Continuem cap a la pestanya "*Acelerando*":

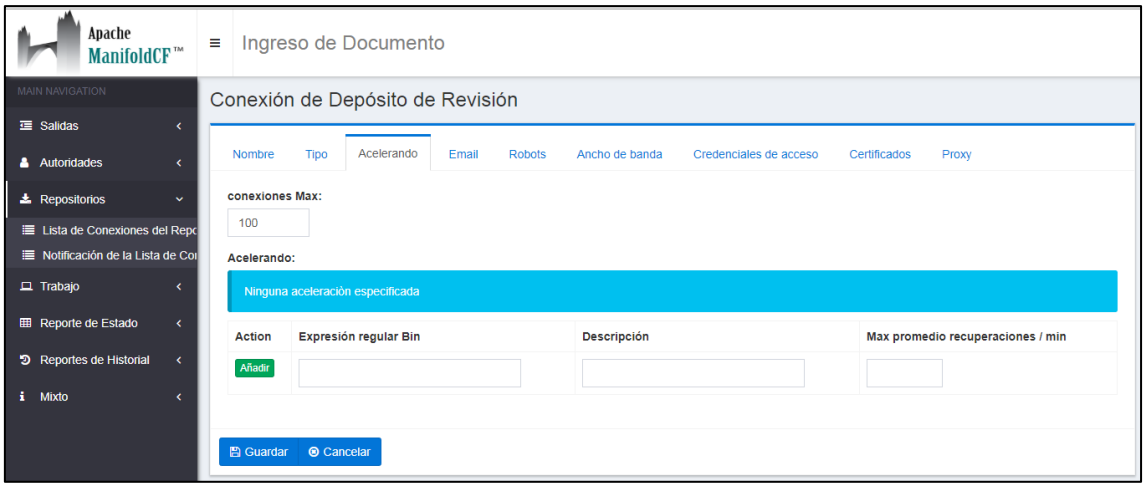


***Figura 34**. Pantalla número de connexions*

Aquí es poden escollir el número de connexions (Hem establert a 100 concurrents en el nostre prototip). A més a més, si es vol accelerar alguna pàgina o grup de pàgines, es pot afegir una expressió regular per cada element o grup d'elements que es vulgui accelerar o reduir connexions específicament.

Continuem amb la pestanya següent "*Email*".

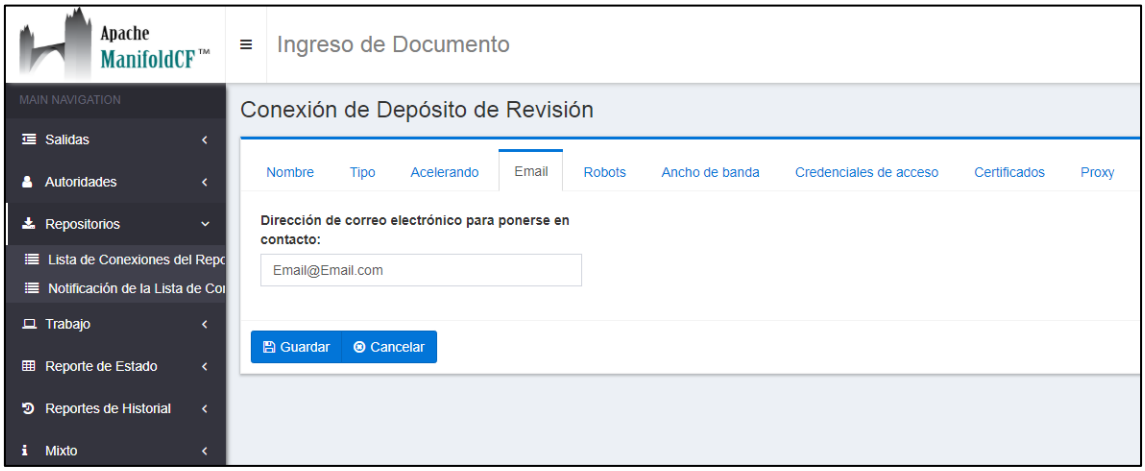


***Figura 35**. Pantalla configuració correu electrònic*

En aquesta pestanya se'ns demana un correu. A priori no s'utilitza per res, però, es pot configurar a l'apartat de la dreta "*Notificación de la lista de Conectores*" que s'enviïn notificacions sobre el connector en cas d'Error, aturada i d'altres opcions, que en qualsevol cas anirien a parar al correu definit. En el nostre prototip hem ficat un correu imaginari purament conceptual.

La següent pestanya correspon als "*Robots*":

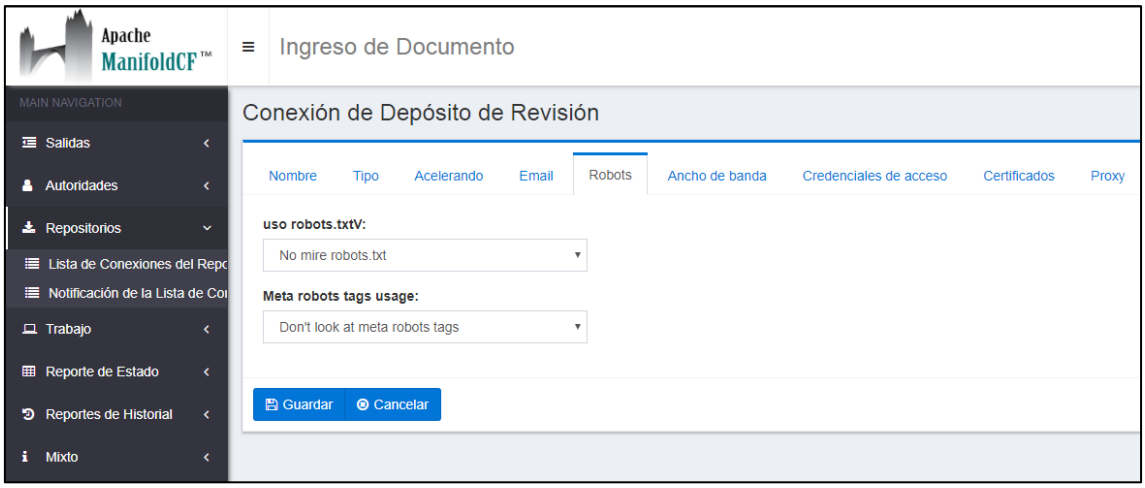


***Figura 36**. Pantalla configuració Robots*

Els "*Robots*" son unes etiquetes que poden afegir els dissenyadors a les pàgines web. Aquestes etiquetes serveixen per definir un comportament concret dels *Crawlers* sobre la pàgina web. Per exemple es pot ficar una etiqueta que obligui al *Crawler* a no extreure les dades d'un apartat concret de la web, o no actuar sobre una web o a reduir la velocitat aplicada, etc.

En el nostre prototip, donat que es va decidir *Crawlejar* la web de *Wikipedia*. No ens ha calgut mirar ni fer cas als "*Robots*" ja que la *Wikipedia* no en té definits.

Continuem amb la configuració de l'ampli de banda:

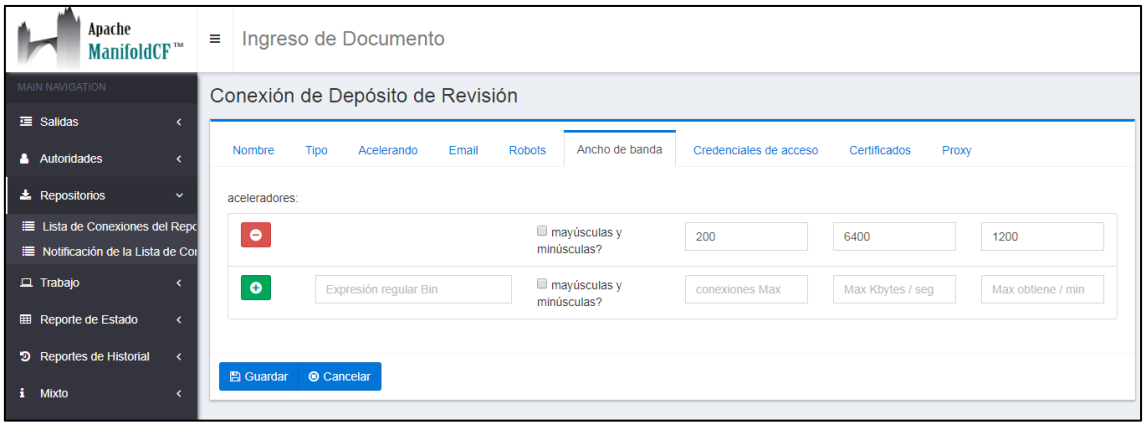


***Figura 37**. Pantalla configuració velocitat de crawleig*

En aquesta pestanya se'ns permet definir un número màxim de connexions, velocitat de transmissió i número màxim d'elements a obtenir. Aquesta configuració és genèrica però se'ns permet definir també de manera concreta (mitjançant expressió regular) si es vol una configuració diferent en certs dominis o certes webs.

En el nostre prototip s'ha establert un volum de connexió màxim de 200, una velocitat de transmissió màxima de 6,4 Kbytes/ segon i un numero màxim d'elements obtinguts de 1200.

Un cop definit l'ampli de banda, continuem amb la pestanya "*credenciales de acceso*":

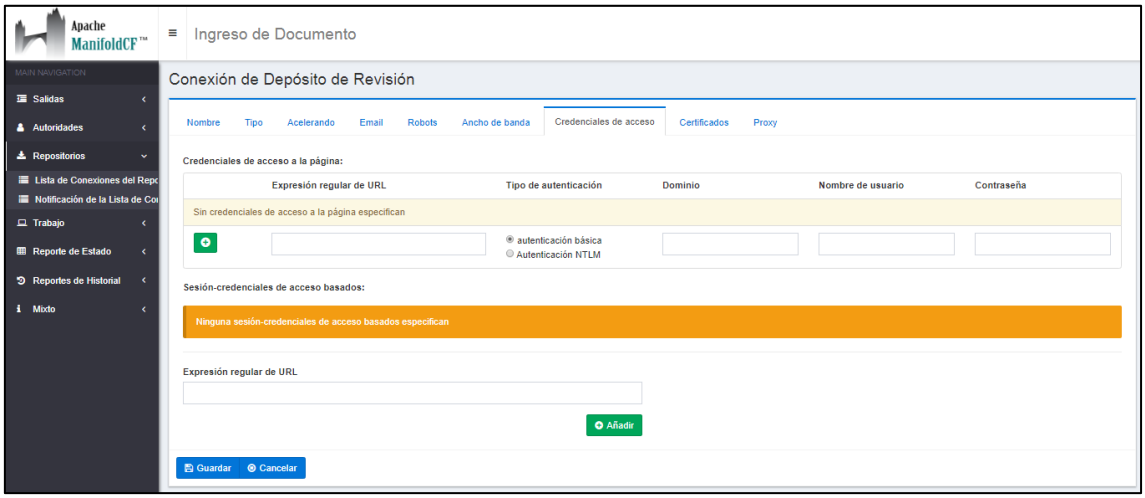


***Figura 38**. Pantalla configuració credencials d'accés*

Aquí es poden configurar diferents credencials per accedir a les webs. Es poden establir tantes com sigui necessari i amb dos protocols d'accés diferents, autenticació bàsica o via *NTLM*. Només cal definir el nom o subconjunt web via expressió regular, el domini aplicat, el nom d'usuari i la contrasenya.

En el nostre prototip no ens ha calgut modificar aquesta pestanya ja que la *Wikipedia* no exigeix estar connectat amb un usuari per poder accedir als continguts publicats.

A continuació veiem la pestanya de "*Certificados*":

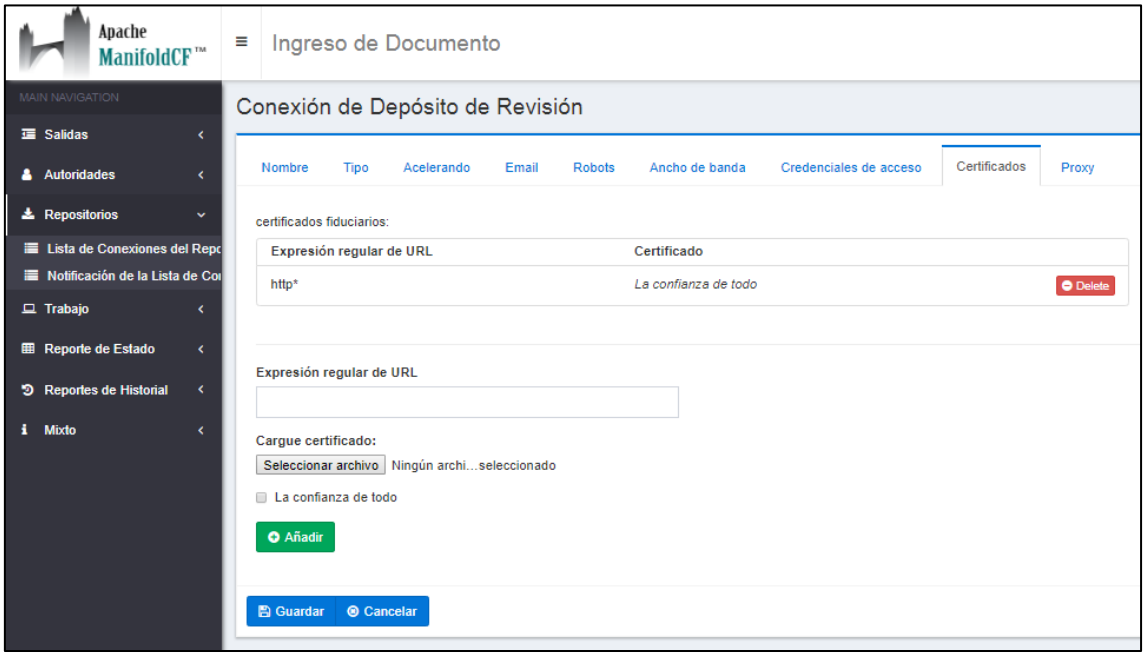


***Figura 39**. Pantalla configuració certificats*

Aquí es poden configurar els diferents certificats digitals, que siguin necessaris per accedir a les webs o dominis definits a les expressions regulars. En el cas del nostre prototip, s'ha establert per tota web que es confiï en totes les peticions sense necessitat d'adjuntar cap certificat digital ja que la *Wikipedia* no està protegida mitjançant protocol SSL.

Finalment procedim a la pestanya de "*Proxy*":

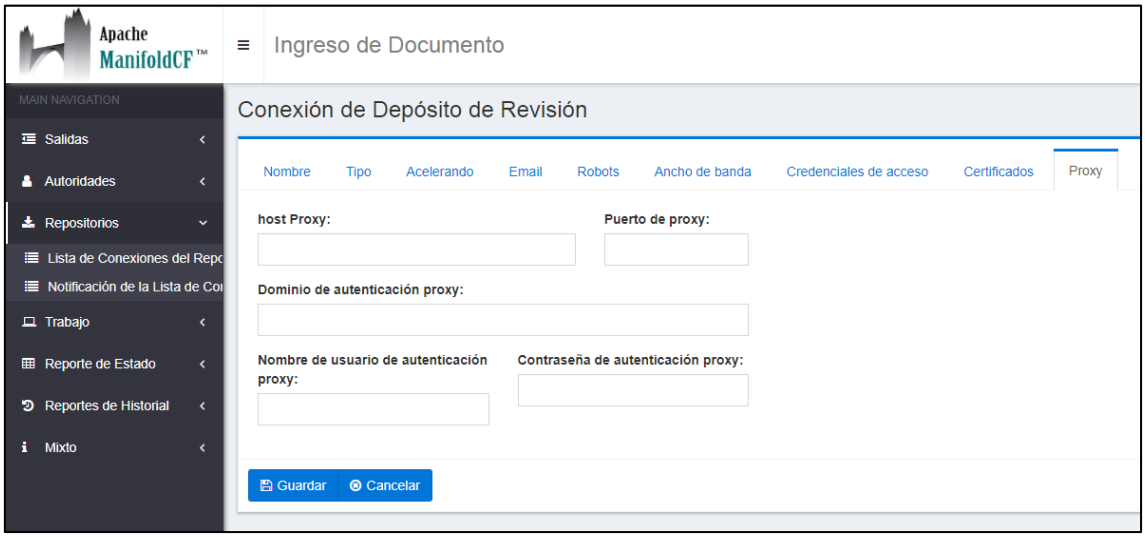


***Figura 40**. Pantalla configuració Proxy*

En aquesta última pestanya es configura el host, port i domini del Proxy en cas d'existir, juntament amb l'usuari i contrasenya per accedir a través d'aquest cap al servidor que es vulgui *crawlejar*. El nostre prototip no disposa de cap *Proxy*.

Finalment, un cop acabada la configuració només cal prémer el botó "*Guardar*" per guardar el connector i poder utilitzar-lo en els *Jobs* posteriorment.

Crear connector de transformació Tika

En aquest punt explicarem com hem creat el transformador per extreure continguts dels fitxers i les webs en el nostre prototip.

Un cop arrancat *ManifoldCF*, al llistat de la dreta podem veure les diferents categories a la que se'ns permet accedir i gestionar. En el cas de creació de connectors a repositoris d'entrada, s'ha d'anar a "*Salidas*" → "*Lista de conexiones de Transformación*" i prémer el botó "*Nuevo Transformador*".

Es mostrarà la següent pantalla:

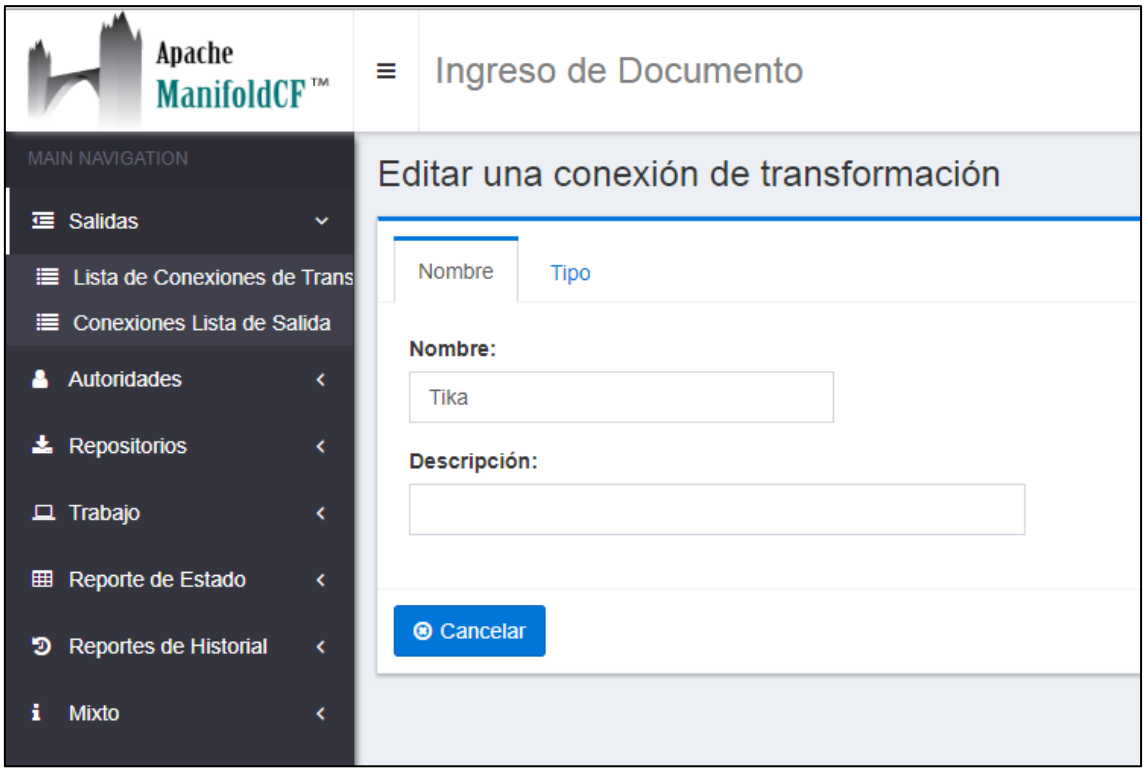


***Figura 41**. Pantalla inicial de creació de transformador*

En aquesta pantalla escollim un nom que identificarà el transformador. Al igual que els connectors, no cal que sigui un nom únic, però és recomanable no duplicar noms ja que després a l'hora de seleccionar el transformador quan creem el *job*, no podrem distingir entre dos noms iguals.

Opcionalment es pot afegir una descripció al transformador.

Finalment canviem a la pestanya "*Tipo*" per escollir el tipus de transformador.

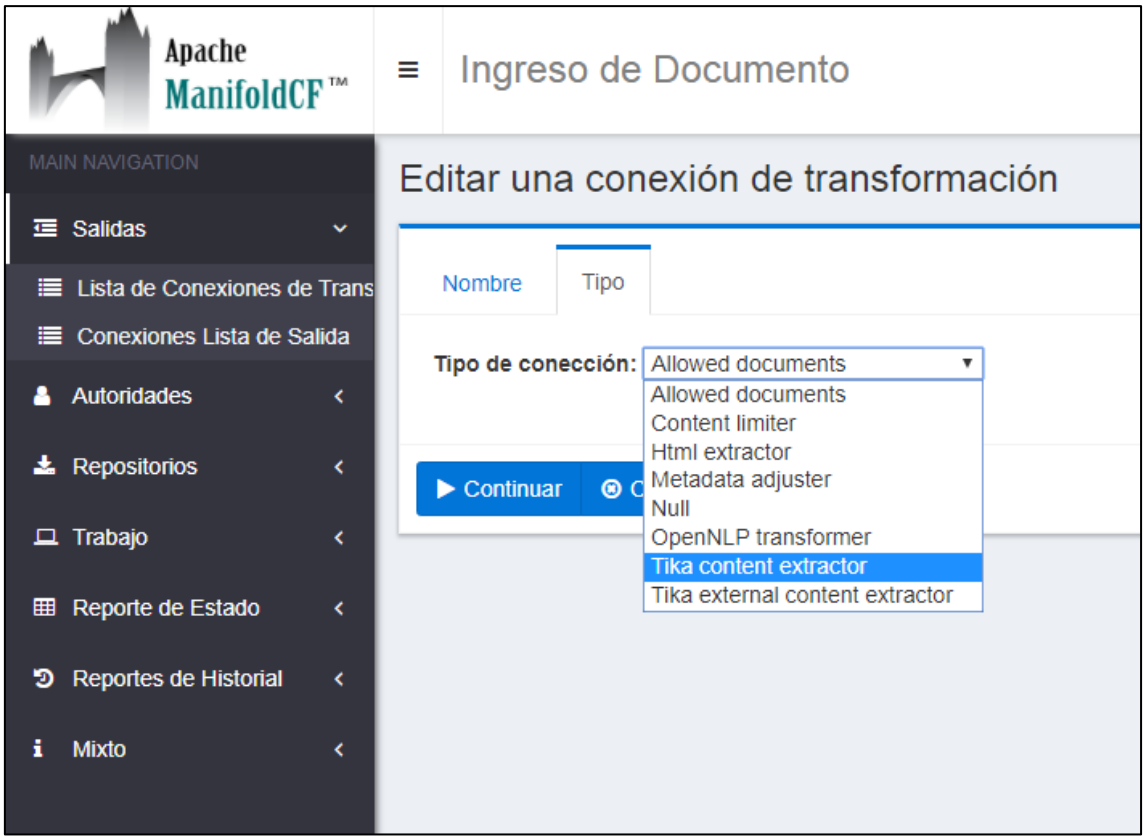


***Figura 42**. Pantalla selector de tipus de transformador*

A la pestanya tipus, escollim "*Tika content extractor*", que correspon al extractor de continguts i metadades de *Tika* com bé indica el seu nom i premem el botó "*siguiente*". Un cop passat aquest punt, ja no es permetrà modificar el tipus de transformador, la resta de dades del transformador seguiran sent editables.

Ens apareixerà la següent pestanya nova:

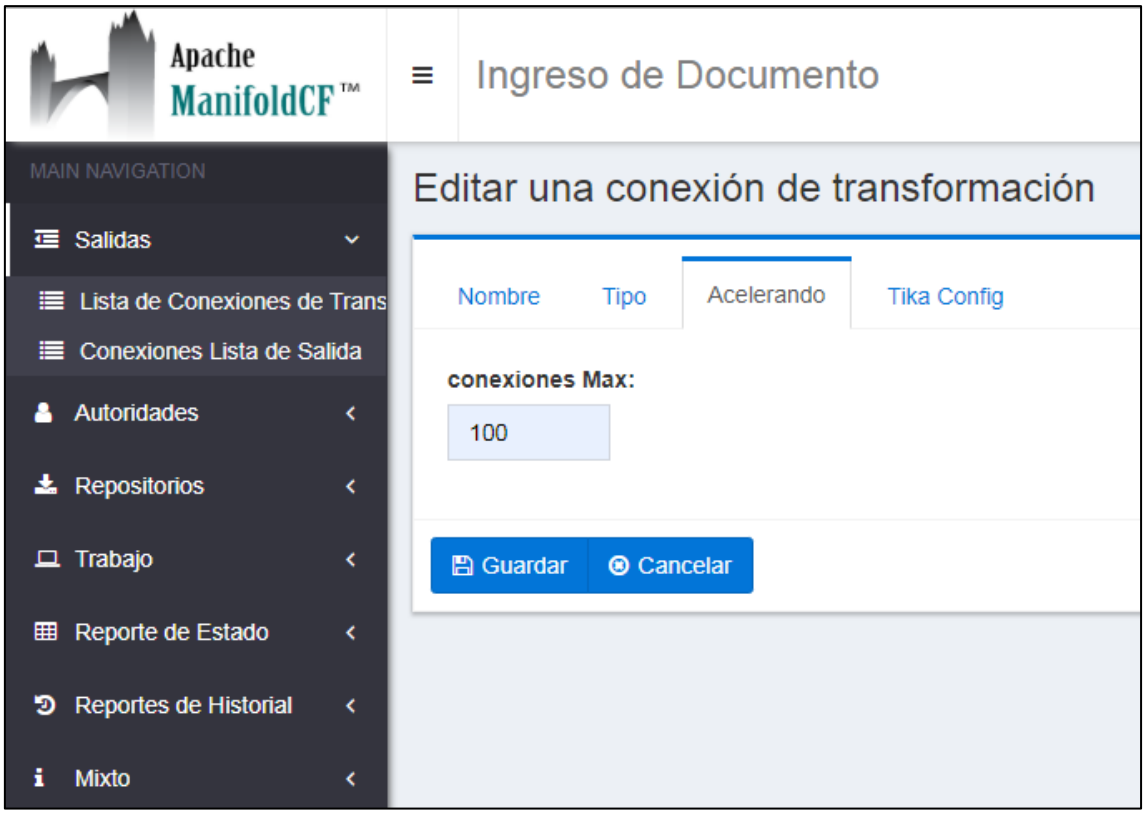


***Figura 43**. Pantalla número de connexions*

Aquí s'estableix el número màxim de connexions pel transformador de *Tika*. En el cas del nostre prototip, l'hem establert a 100 concurrents.

Finalment la pestanya de configuració de *Tika*, l'hem deixat en blanc ja que no requerim de configuracions addicionals.

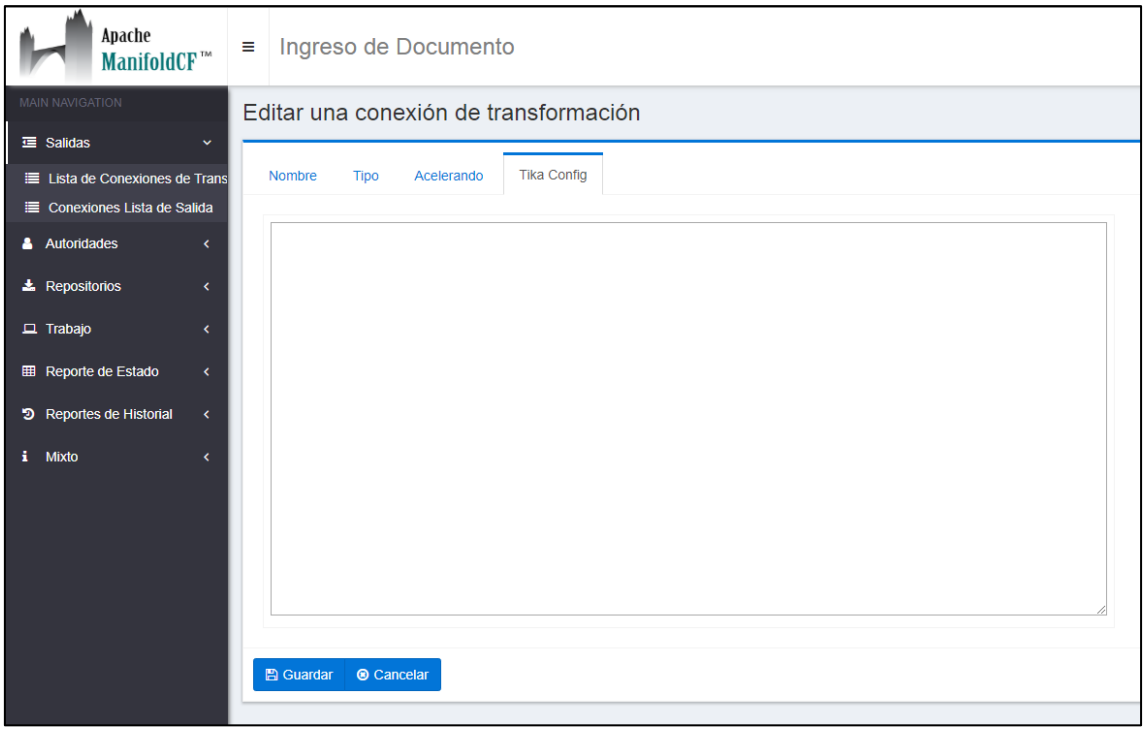


***Figura 44**. Pantalla configuracións adicionals Tika*

Un cop acabats aquests passos només cal prémer el botó "*Guardar*" per guardar el connector i poder-lo utilitzar posteriorment en els *Jobs*.

Crear connector de sortida Elasticsearch

ManifoldCF disposa de un ampli conjunt de connectors de sortida. En el nostre projecte es va escollir el motor de cerca *Elasticsearch*, per tant en aquest punt explicarem com generar el connector cap a *Elasticsearch* i les diferents configuracions tal com les hem fet al nostre prototip.

Des de la pàgina inicial de *ManifoldCF*, al llistat de la dreta podem veure les diferents categories a la que se'ns permet accedir i gestionar. En el cas de creació de connectors de sortida, s'ha d'anar a "*Salidas*" → "*Conexiones Lista de Salida*" i prémer el botó "*Nuevo Conector*".

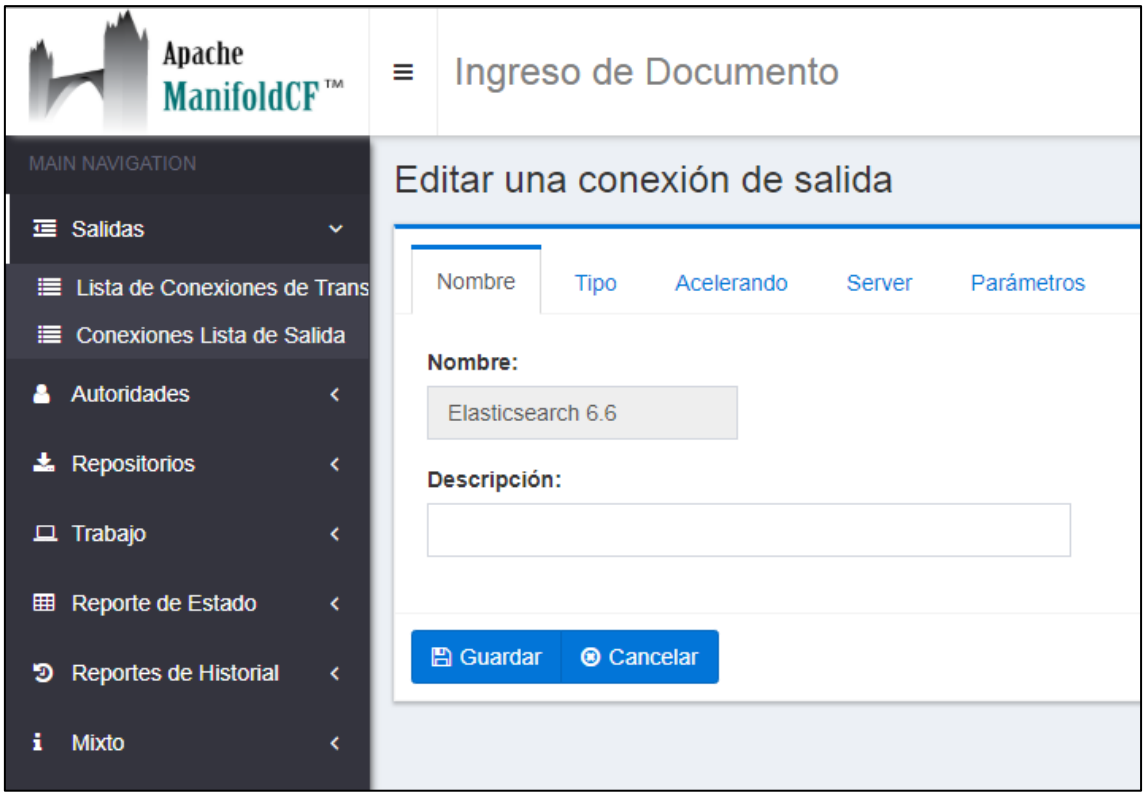


Figura 45. *Pantalla inicial de creació de connector*

En aquesta pantalla escollim un nom que identificarà el connector de sortida. Al igual que la resta de connectors, no cal que sigui únic però és recomanable no duplicar noms ja que després a l'hora de seleccionar el connector quan creem el *job*, no podrem distingir entre dos noms iguals.

Opcionalment es pot afegir una descripció al connector.

Finalment canviem a la pestanya "*Tipo*" per escollir el tipus de connector.

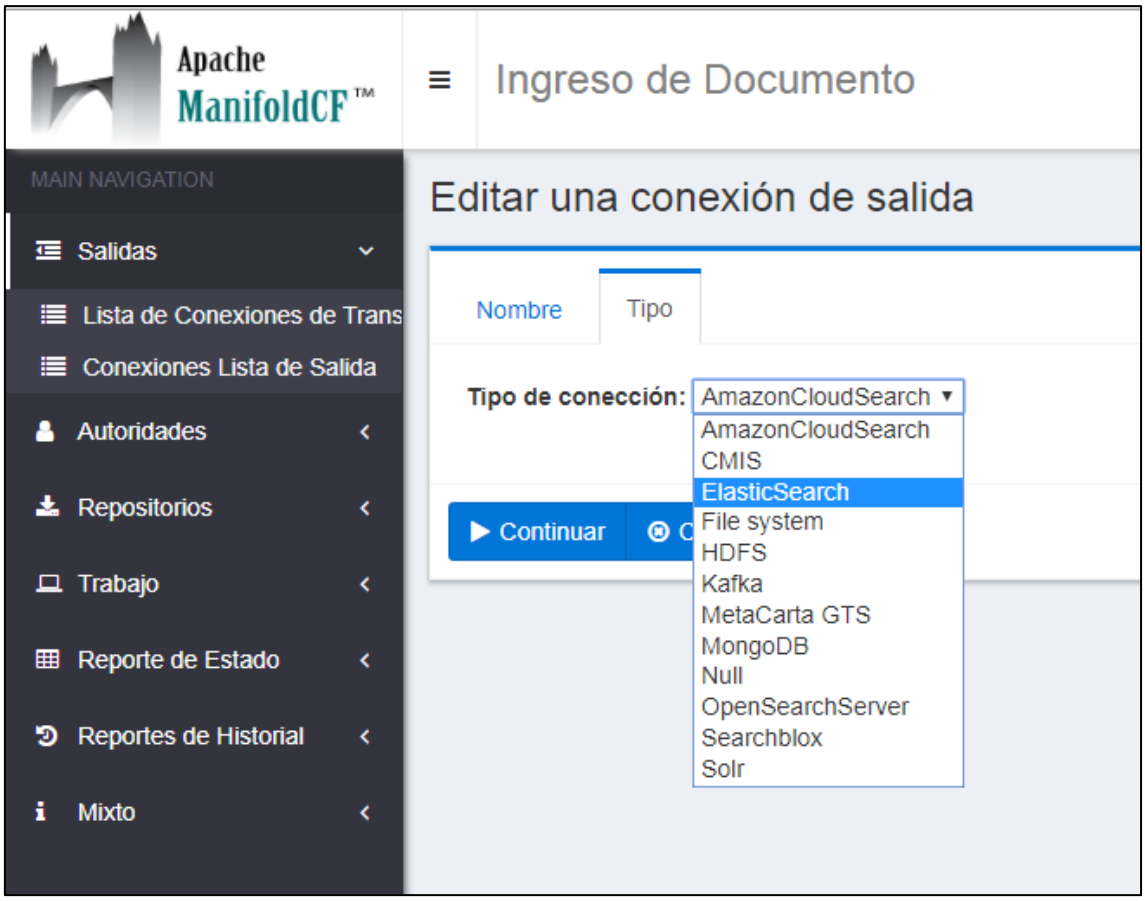


***Figura 46**. Pantalla selector de tipus de connector*

A la pestanya tipus, escollim "*Elasticsearch*", que correspon al connector cap a *Elasticsearch* com bé indica el seu nom i premem el botó "*Continuar*". Un cop passat aquest punt, ja no es permetrà modificar el tipus de connector, la resta de dades del connector seguiran sent editables.

Ens apareixerà la següent pestanya nova:

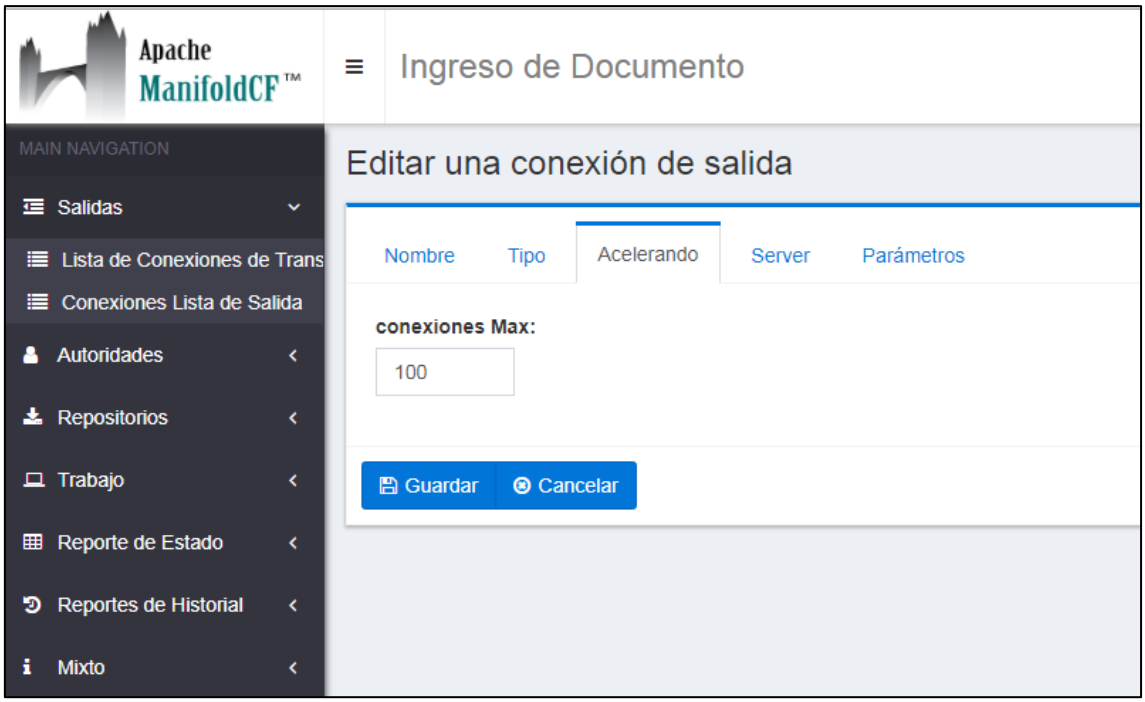


***Figura 47**. Pantalla número de connexions*

Aquí s'estableix el número màxim de connexions pel connector de sortida de *Elasticsearch*. En el cas del nostre prototip, l'hem establert a 100 concurrents.

Continuem amb la pestanya de configuració del servidor on està *Elasticsearch*:

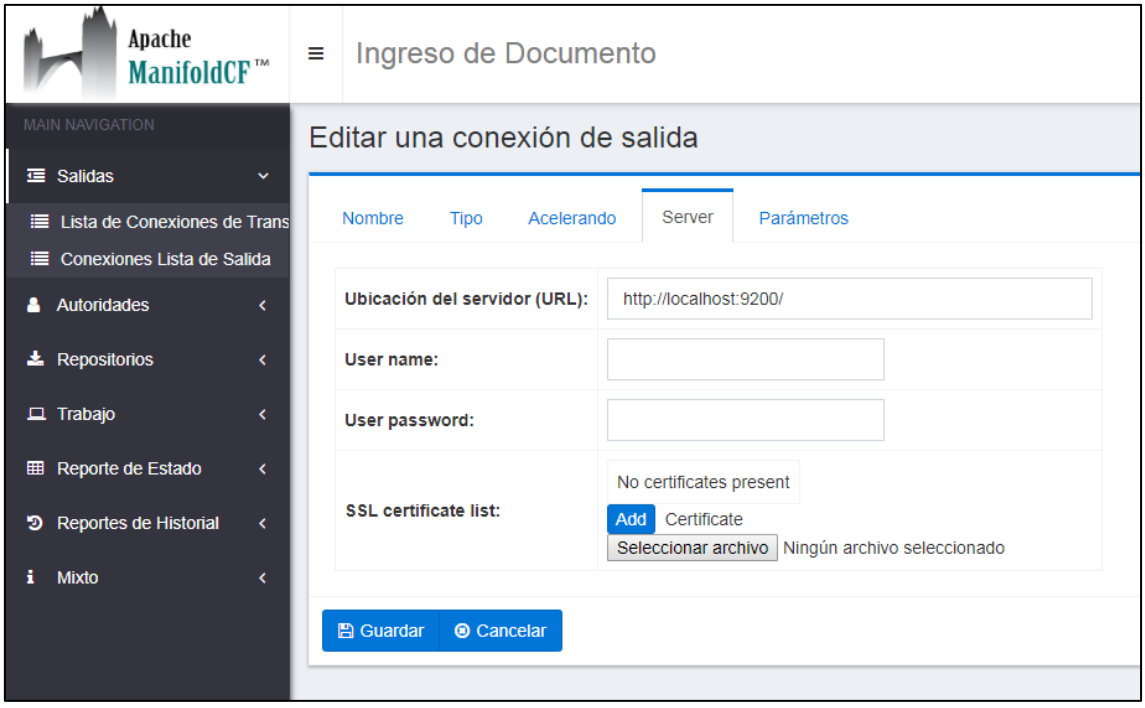


***Figura 48**. Pantalla de configuració del servidor d'Elasticsearch*

Aqui es configura la ruta on es troba el servidor on està l'Elasticsearch juntament amb el port corresponent. Per defecte Elasticsearch sempre es desplega sobre el port 9200, però podria canviar-se. Al tenir el prototip en el mateix PC a on està el ManifoldCF, hem deixat la ruta de "localhost".

Aquí també permet configurar usuari i contrasenya en cas que existís i certificat digital si es disposes de seguretat SSL. No es el cas en el nostre prototip.

Finalment anem a la següent pestanya "*Parámetros*":

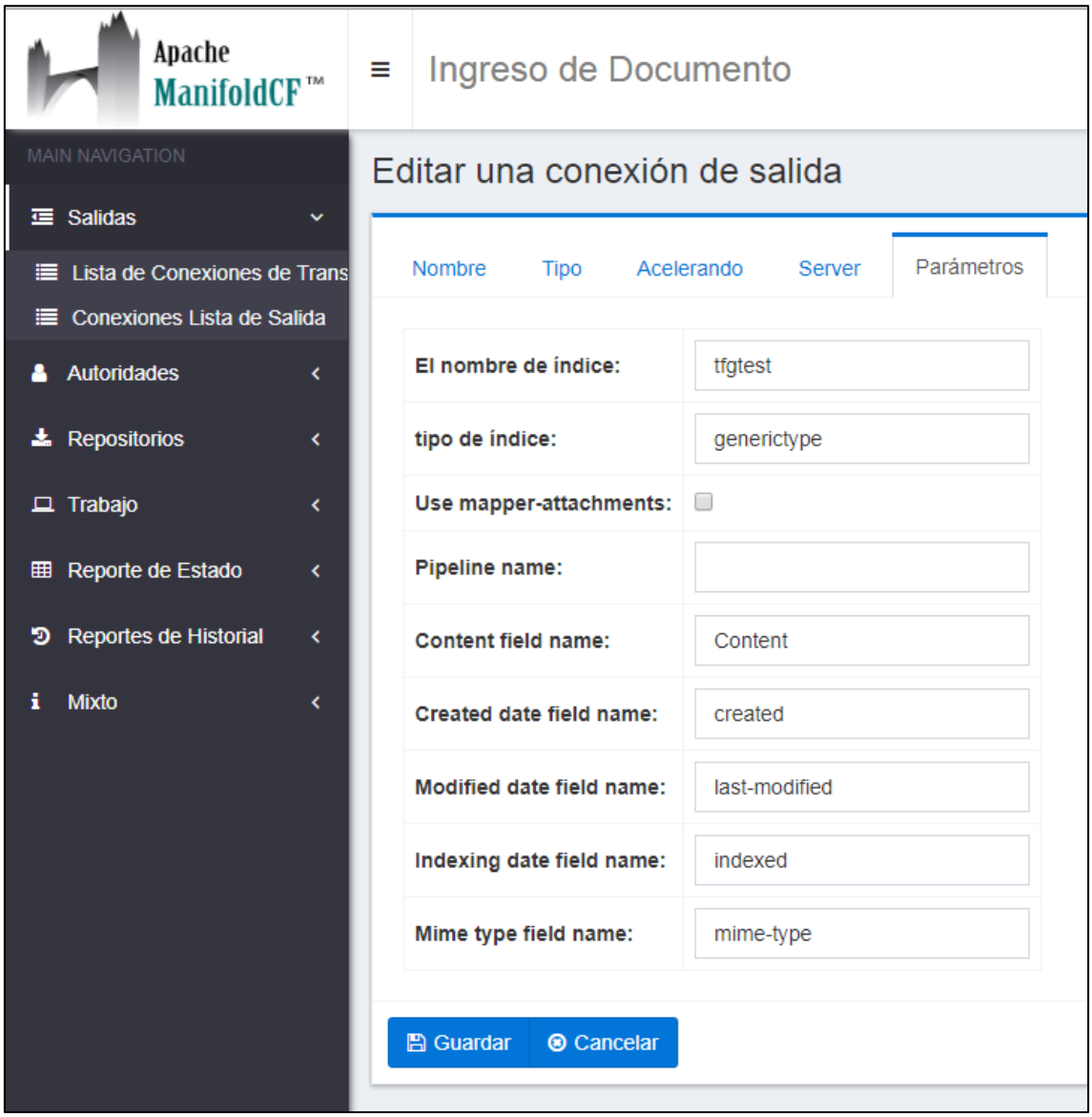


***Figura 49**. Pantalla configuració paràmetres Elasticsearch*

En aquesta pestanya se'ns demanaran que aportem les següents dades sobre l'índex *d'Elasticsearch* al que enllaçarem el connector: Nom de l'índex, tipus d'índex (és purament conceptual, el tipus el defineix el propi usuari i no té cap afectació sobre l'índex. Serveix per si es vol subdividir l'índex en subtipus d'elements), si s'utilitza el "*mapper-attachments*" (és un *plugin* de *Elasticsearch* que descodifica els continguts en

base64 a text pla. A les noves versions *d'Elasticsearch* ja no cal aquest *plugin* ja que està integrat), Noms alternatius a diferents camps com ara el camp contingut, data de creació, data de modificació...

Finalment, un cop acabada la configuració només cal prémer el botó "*Guardar*" per guardar el connector i poder utilitzar-lo en els *Jobs* posteriorment.

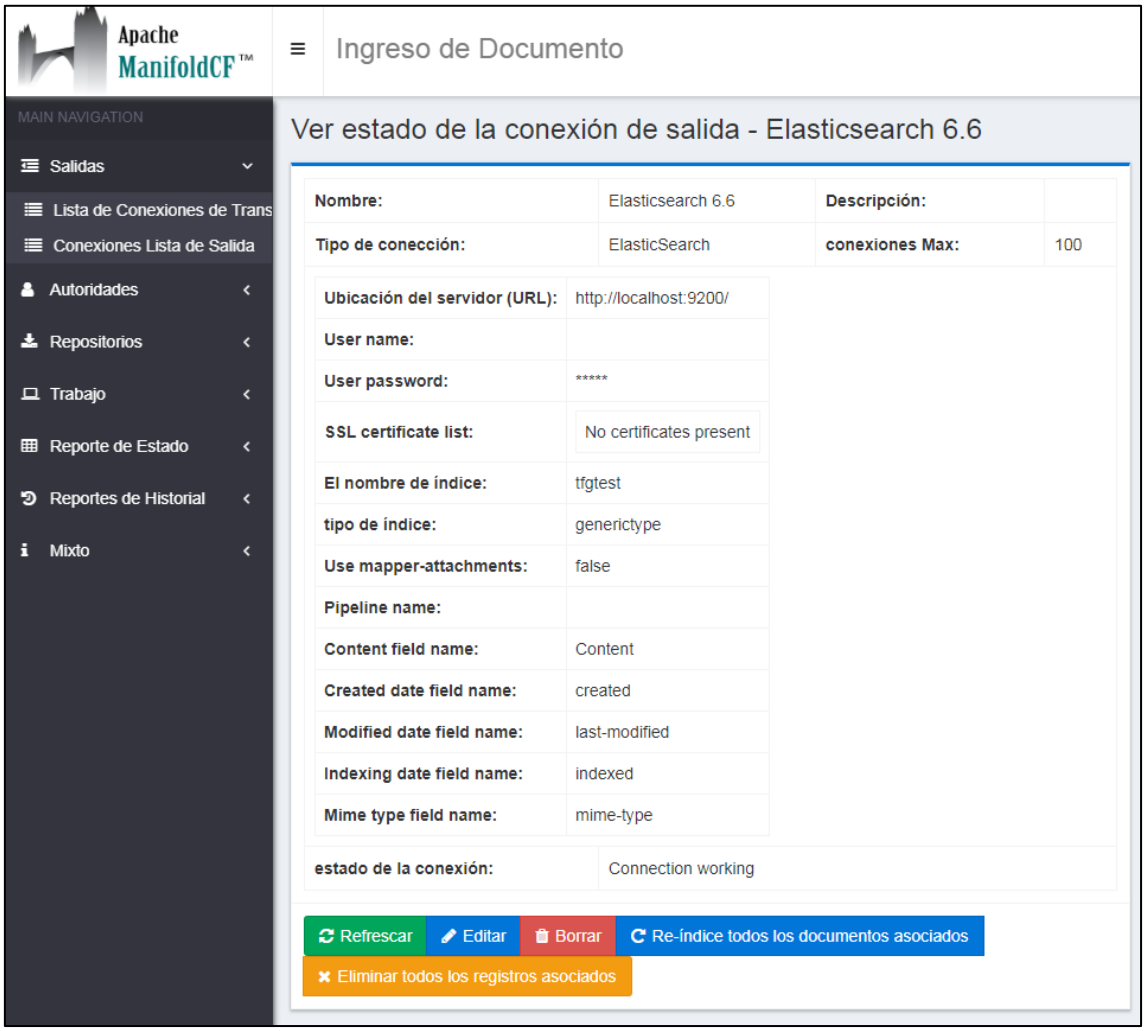


***Figura 50**. Pantalla de resum general del connector de sortida*

A aquesta pantalla "resum" podem veure l'estat de la connexió. Si ha pogut establir connexió amb *l'Elasticsearch* veurem que surt "*Connection working*".

Crear Jobs

En aquest punt explicarem com generar *jobs* amb *ManifoldCF* utilitzant els connectors i transformadors generats anteriorment. En el nostre prototip s'han utilitzat dos *jobs*, un per *crawlejar* un sistema de fitxers i l'altre per *crawlejar* un entorn web.

A continuació explicarem els punts a tenir en compte en la creació d'aquests processos de *crawlejat*.

File System

El *job* de *File System* ens permetrà donat una ruta o conjunt de rutes, obtenir tots els fitxers i guardar-los a *Elasticsearch* fent ús dels connectors generats amb anterioritat.

Per generar un nou *job*, ca anar des de la pàgina inicial de *ManifoldCF* a "*Trabajo*" → "*Listar todos los trabajos*" i prémer el botó "*Nuevo Trabajo*".

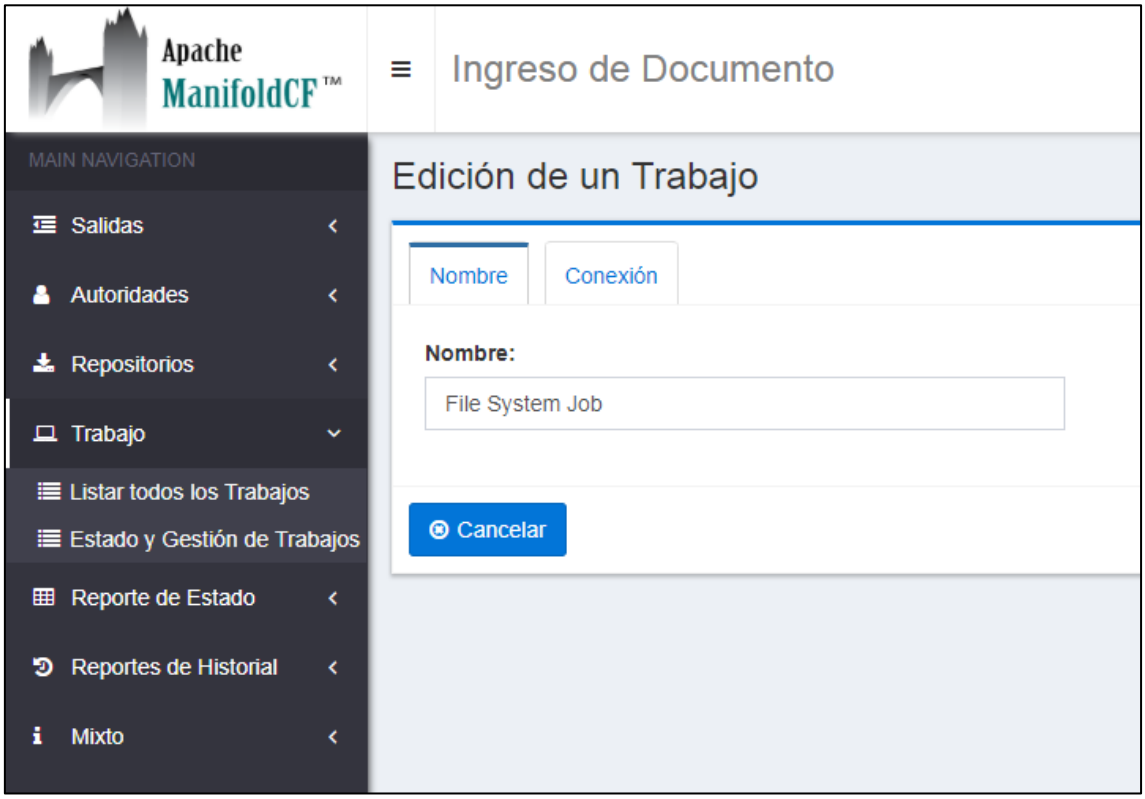


***Figura 51**. Pantalla inicial de creació de Job*

En aquesta pantalla escollim un nom que identificarà el *job*. Al igual que els connectors, no cal que sigui un nom únic, però és recomanable no duplicar noms ja que després a l'hora de llençar el procés no podrem distingir entre dos noms iguals.

A continuació canviem a la pestanya "*Conexión*" per muntar el procés que seguirà el *job*.

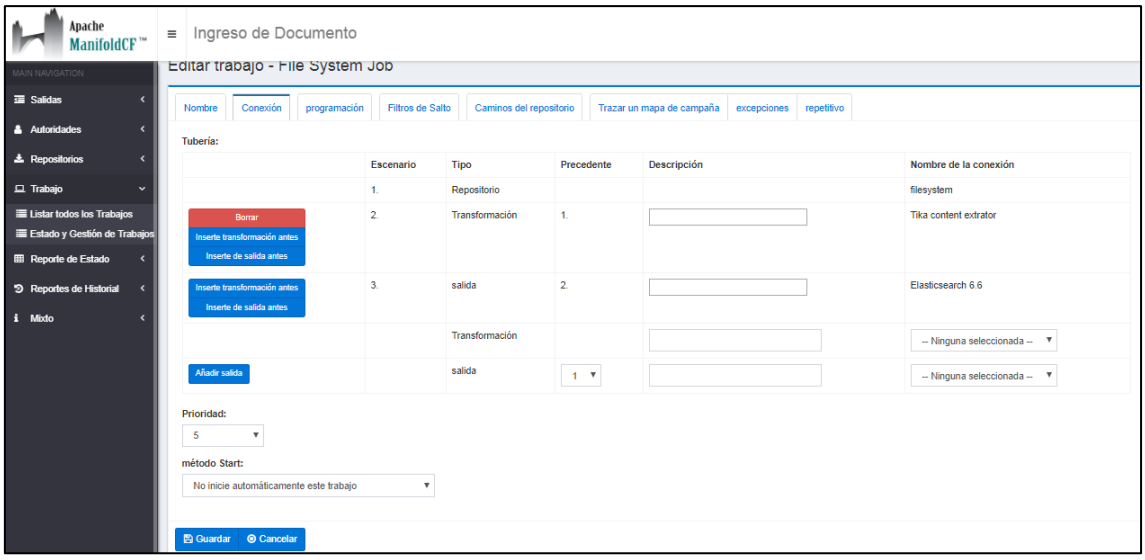


***Figura 52**. Pantalla de configuració de connexió del job*

En aquesta pestanya es defineix el procés que seguirà el *job*. En aquest cas, s'ha escollit el connector d'entrada de *File System*, seguit del transformador de *Tika* i la sortida de dades de *Elasticsearch*. Es poden afegir tants transformadors i sortides com es desitgi però només una entrada.

Un cop definit el procés que seguirà el *job*, continuem amb la pestanya "*Programación*":

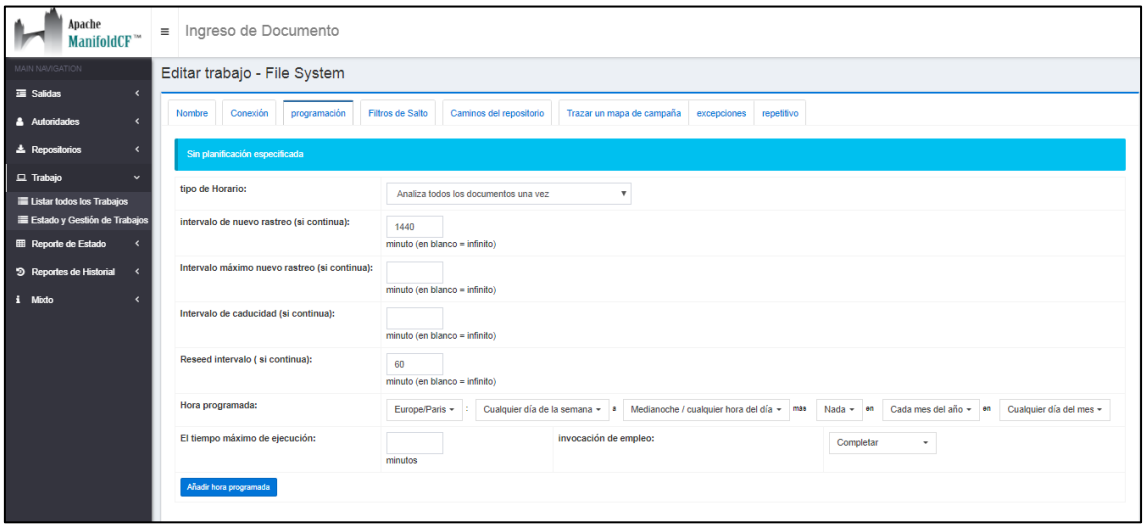


***Figura 53**. Pantalla de configuració de programació automàtica*

Per defecte està establert per llençar-se únicament de forma manual (i així ho hem deixat al prototip), però aquí es permet seleccionar els dies, hores i fins i tot un interval de temps en que vol que es pari si el *job* el supera.

Un cop establerta la programació de llançament del *job*, continuem amb la pestanya "*Filtros de salto*":

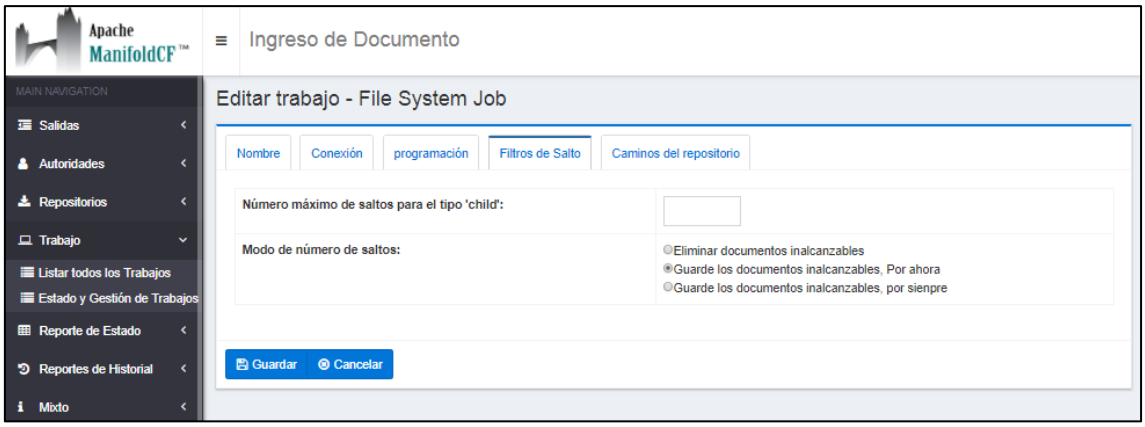


***Figura 54**. Pantalla de configuració de profunditat de crawleig*

Aquí es permet limitar la "profunditat" de *crawleig*. Per defecte, sense valor significa que no té límit i *crawlejarà* tot el que trobi.

Finalment cal configurar la ruta a *crawlejar* en la següent pestanya:

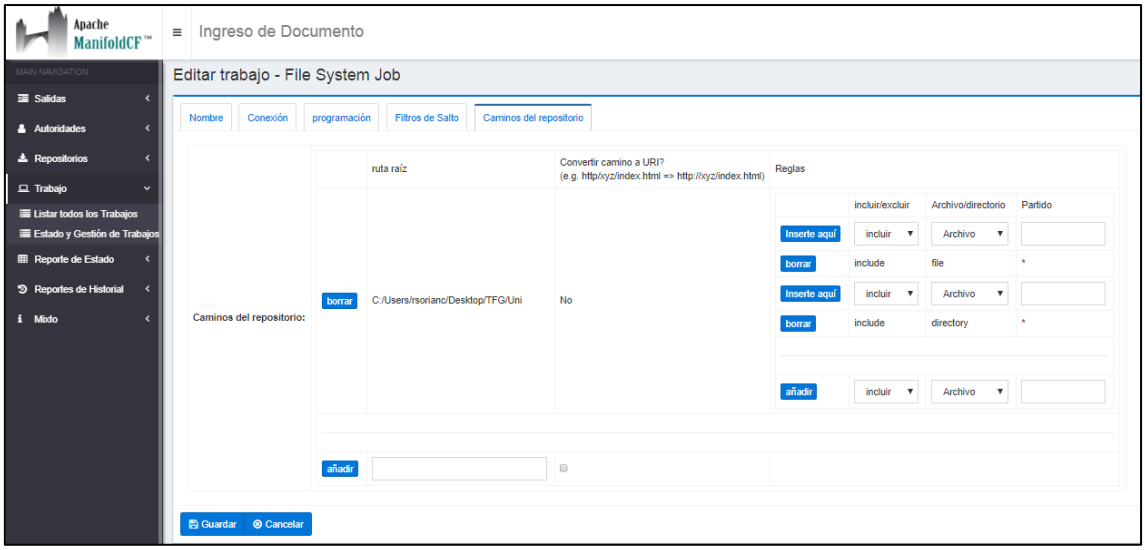


***Figura 55**. Pantalla de configuració de ruta de crawleig*

Es poden definir tantes rutes com es vulgui. A més no cal que siguin del mateix entorn a on estigui el *ManifoldCF*, sinó que també permet el *crawleig* a rutes remotes.

En el nostre prototip hem establert una única ruta a on teníem tots els documents que volíem *crawlejar*.

Finalment cal prémer el botó "*Guardar*" per guardar el "*job*".

Web

El *job* de *crawleig* Web ens permetrà dona una ruta o conjunt de rutes, obtenir totes les webs que estiguin *linkades* i guardar-les a *Elasticsearch* fent ús dels connectors generats amb anterioritat.

Per generar un nou *job*, ca anar des de la pàgina inicial de *ManifoldCF* a "*Trabajo*" → "*Listar todos los trabajos*" i prémer el botó "*Nuevo Trabajo*".

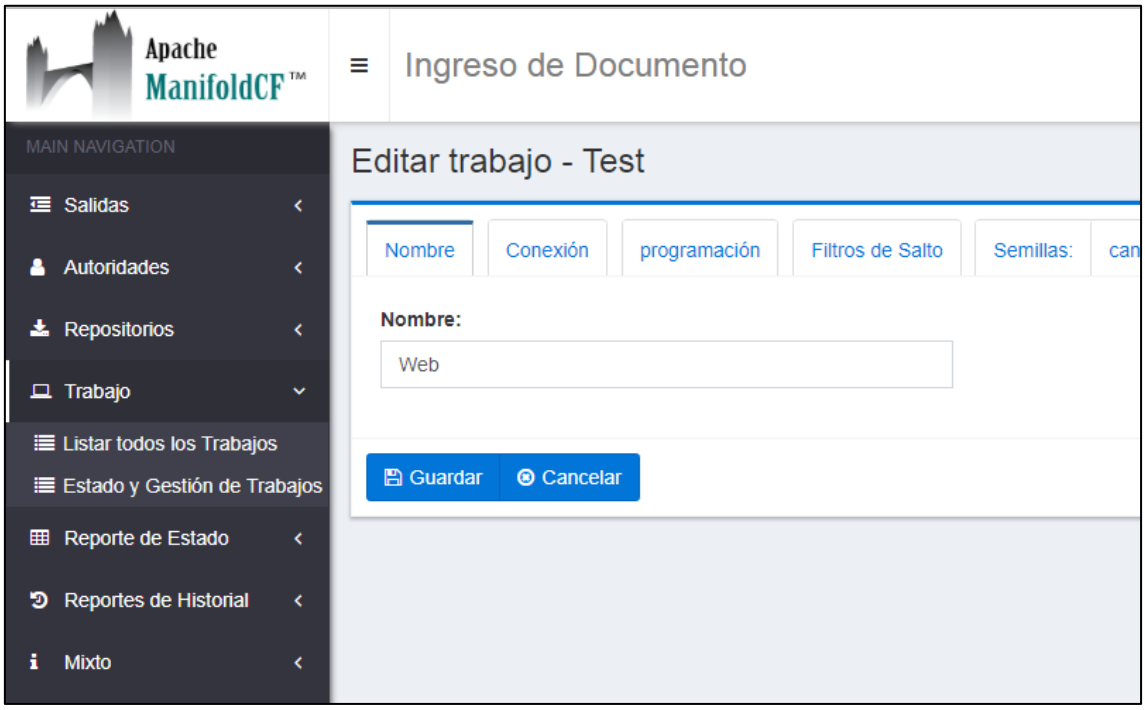


***Figura 56**. Pantalla inicial de creació de Job*

En aquesta pantalla escollim un nom que identificarà el *job*. Al igual que els connectors, no cal que sigui un nom únic, però és recomanable no duplicar noms ja que després a l'hora de llençar el procés no podrem distingir entre dos noms iguals.

A continuació canviem a la pestanya "*Conexión*" per muntar el procés que seguirà el *job*.

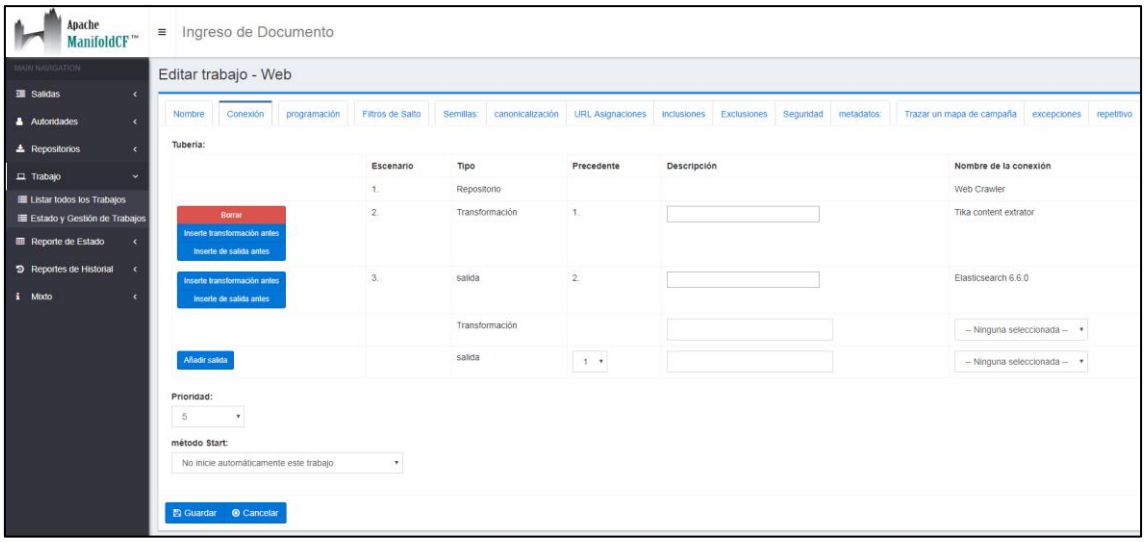


***Figura 57**. Pantalla de configuració de connexió del job*

En aquesta pestanya es defineix el procés que seguirà el *job*. En aquest cas, s'ha escollit el connector d'entrada de Web, seguit del transformador de *Tika* i la sortida de dades de *Elasticsearch*. Es poden afegir tants transformadors i sortides com es desitgi però només una entrada.

Un cop definit el procés que seguirà el *job*, continuem amb la pestanya "*Programación*":

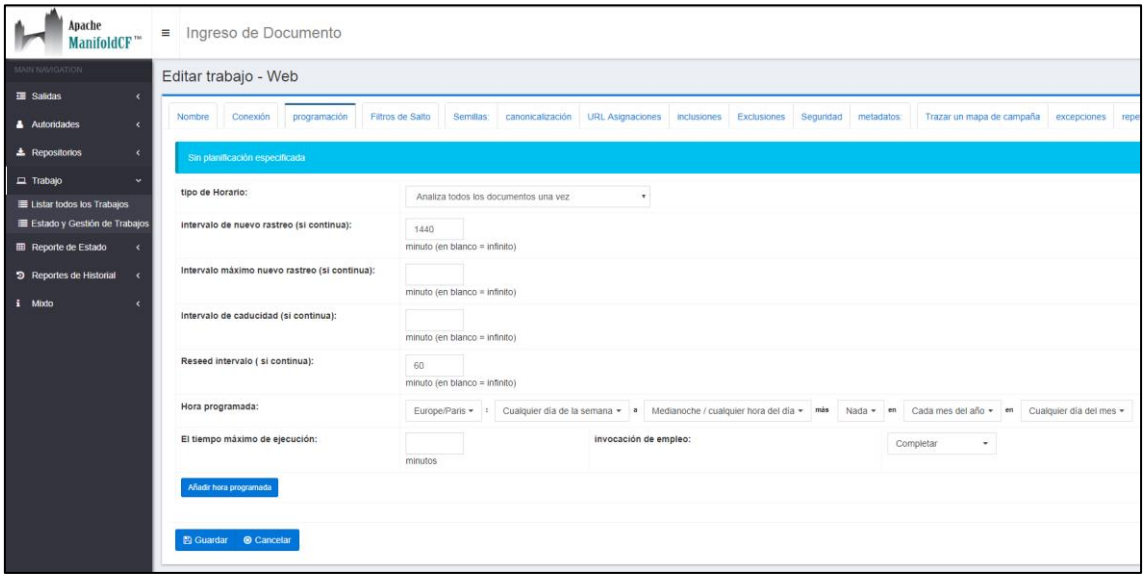


***Figura 58**. Pantalla de configuració de programació automàtica*

Per defecte està establert per llençar-se únicament de forma manual (i així ho hem deixat al prototip), però aquí es permet seleccionar els dies, hores i fins i tot un interval de temps en que vol que es pari si el *job* el supera.

Un cop establerta la programació de llançament del *job*, continuem amb la pestanya "*Filtros de salto*":

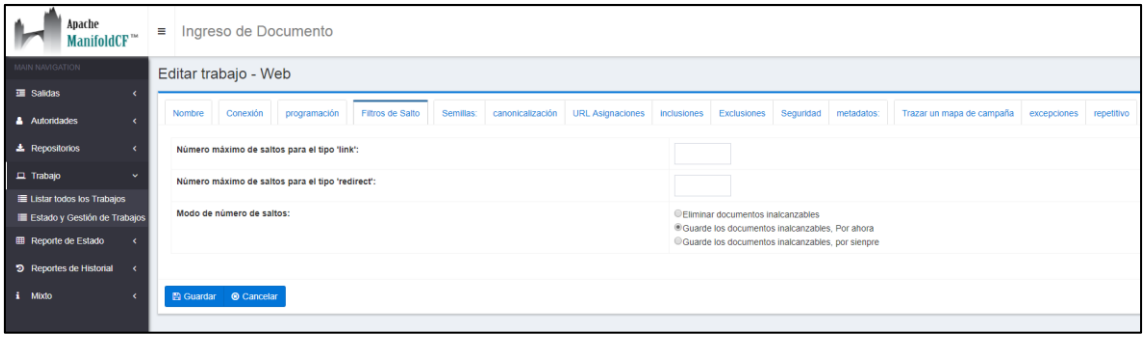


***Figura 59**. Pantalla de configuració de profunditat de crawleig*

Aquí es permet limitar la "profunditat" de *crawleig*. Per defecte, sense valor significa que no té límit i *crawlejarà* tot el que trobi.

Finalment cal configurar la url a *crawlejar* en la següent pestanya "*Semillas*":

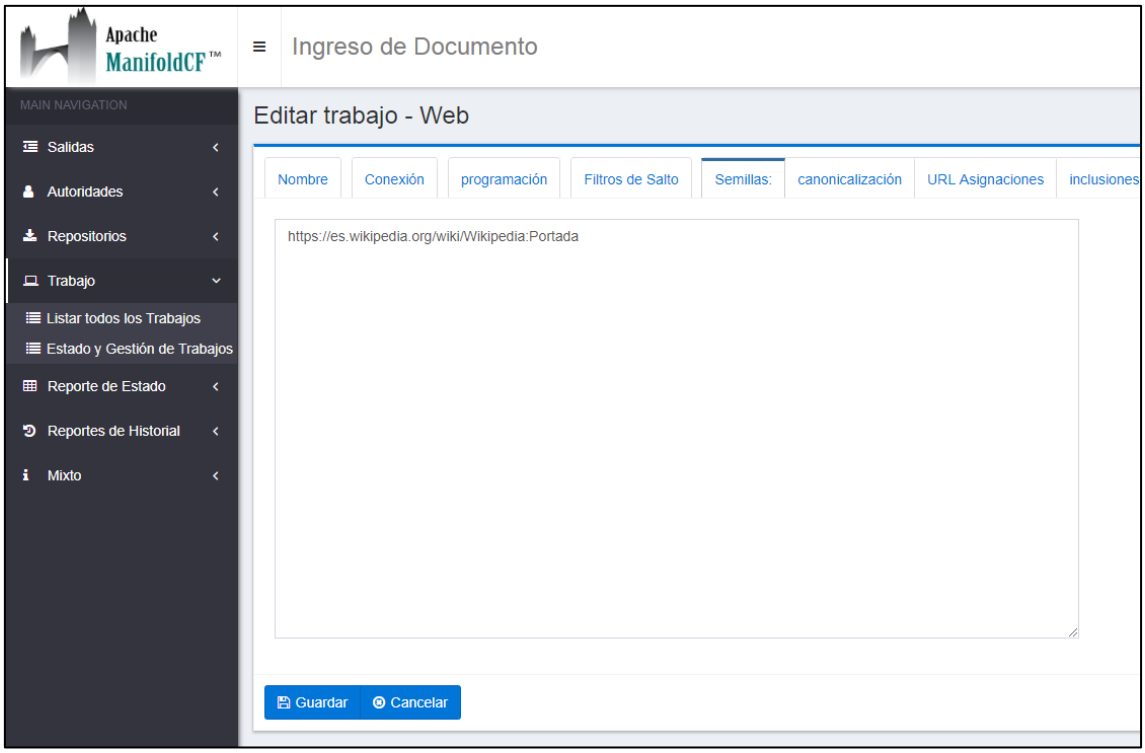


***Figura 60**. Pantalla de introducció de rutes d'inici de crawleig*

Aquesta pestanya inclourà la *url* (o *urls*) a partir de les quals es començarà a *crawlejar* fins a la profunditat definida en la pestanya "*Saltos*". Cada *url* definida anirà seguida d'un salt de línia. En el cas del nostre prototip, només s'ha definit una *url* a partir de la qual es *crawlejarà* la *Wikipedia*.

Un cop definides les urls passem a la següent pestanya "Canonicalización" :

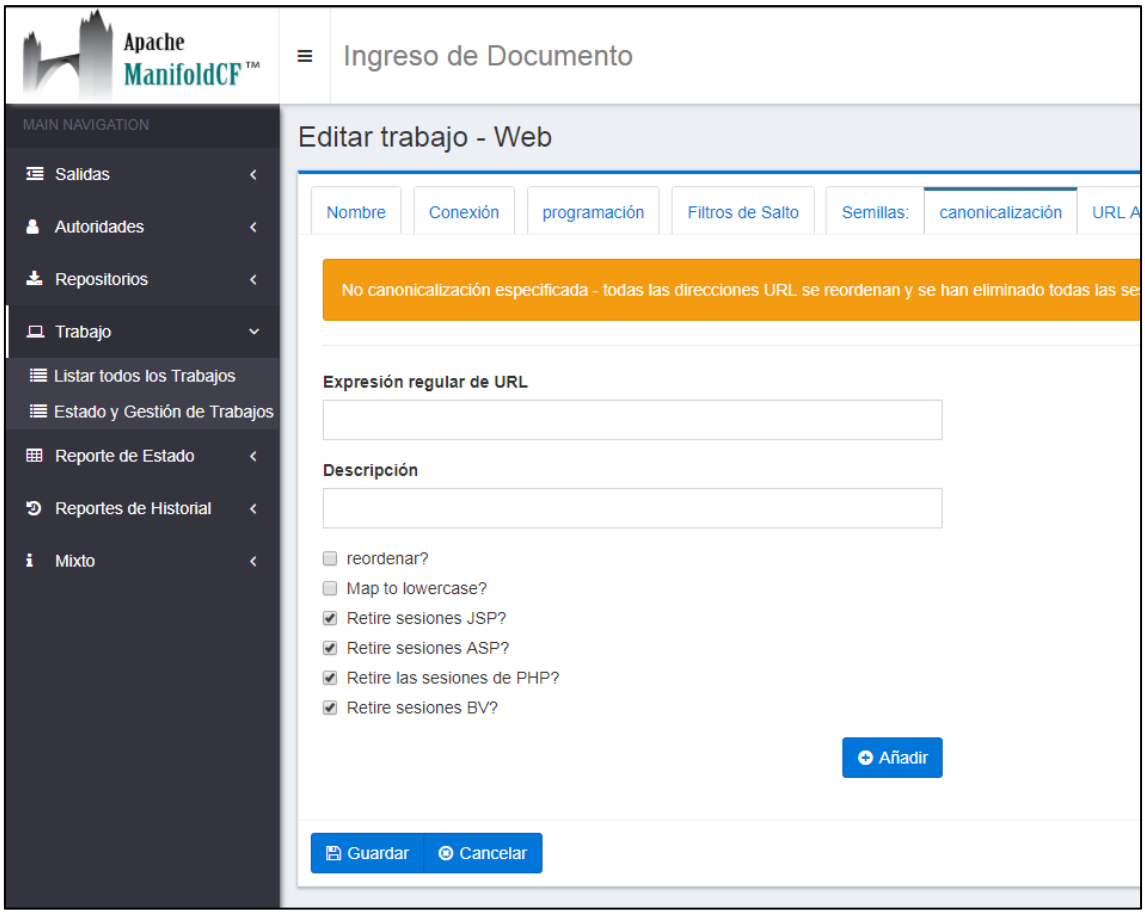


***Figura 61**. Pantalla de configuració de sessions*

En aquesta part de la configuració es pot afegir per cada *url* desitjada, si es vol mantenir les sessions iniciades o no fer-ne cas. A més també es permet *mapejar* a minúscules o inclòs reordenar les *urls*. En el cas del nostre prototip l'hem deixat com venia per defecte, tal com es veu a la imatge.

Continuem amb la pestanya "*URL Asignaciones*":

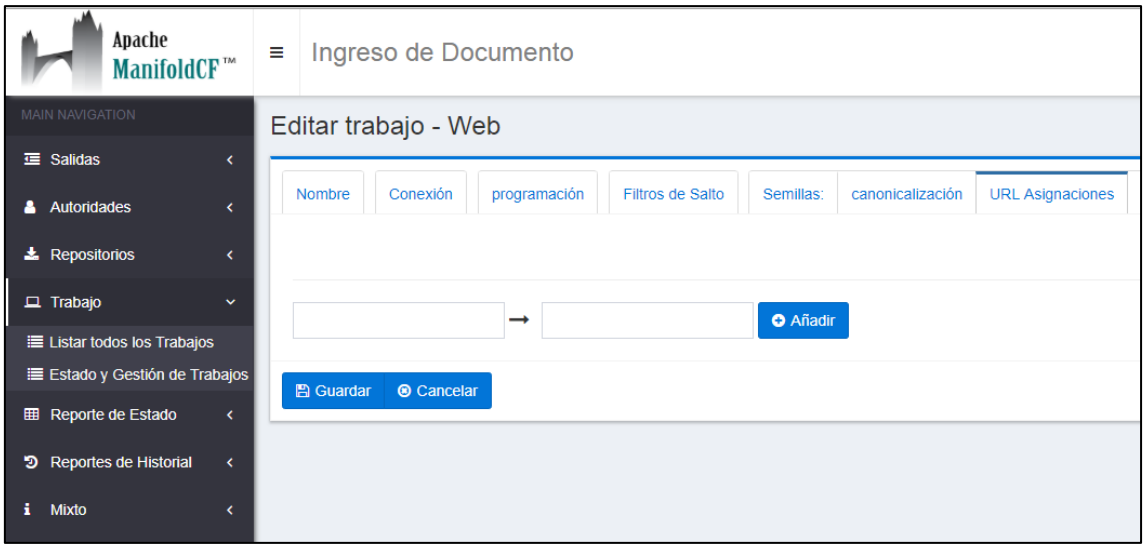


***Figura 62**. Pantalla de mapejat de Urls*

Aqui es permet assignar/modificar urls introduïnt-es com a expressió regular i assignant un nou mapejat. Això és comú utilitzar-ho quan les url requereixen paràmetres concrets. En el cas del nostre prototip la Wikipedia no requereix passar-li cap paràmetre ni modificar la ruta de cap url.

Passem a les inclusions de l'índex durant el crawleig.

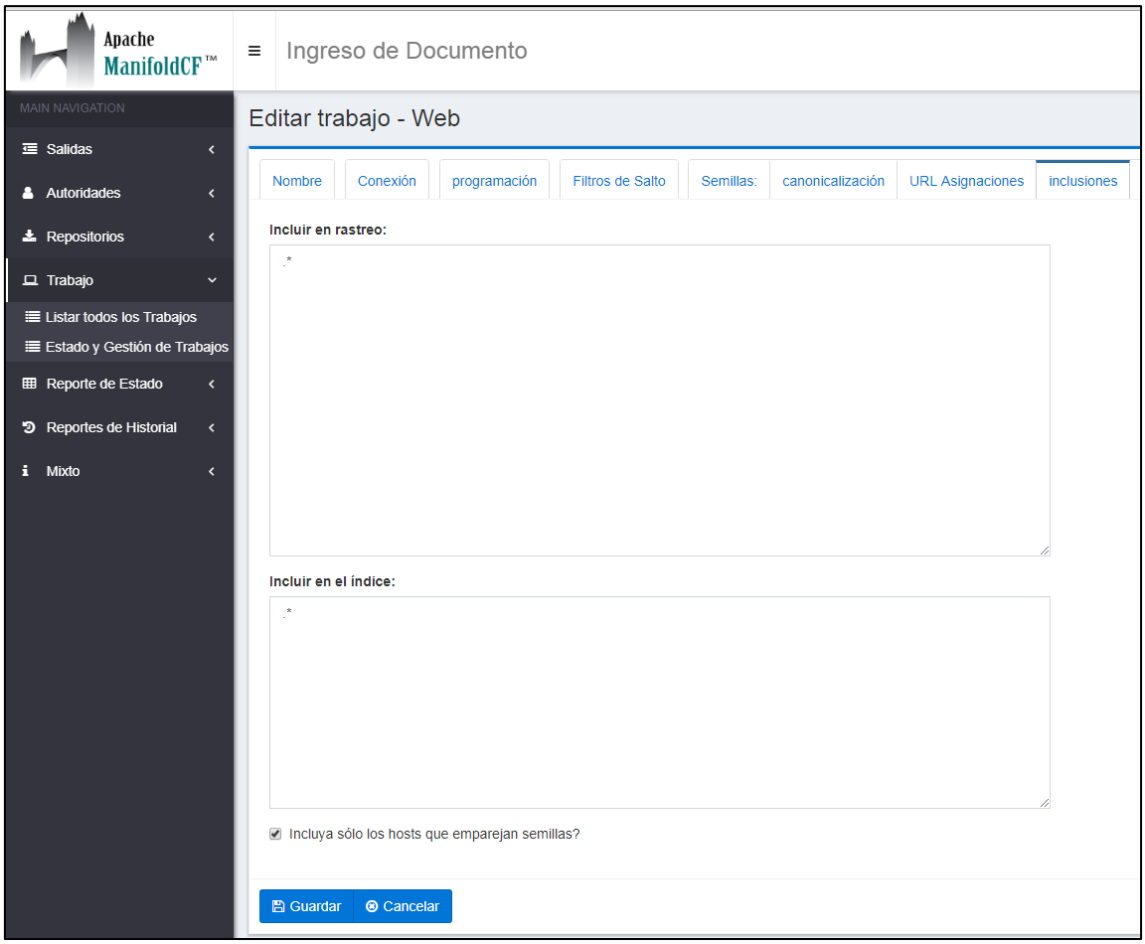


***Figura 63**. Pantalla de configuració d'inclusions*

Aquí es permet decidir si es vol incloure només un conjunt d'elements a l'índex. Per efecte al nostre prototip hem decidit incloure tot i excloure elements concrets des de la següent pestanya "*Exclusiones*"

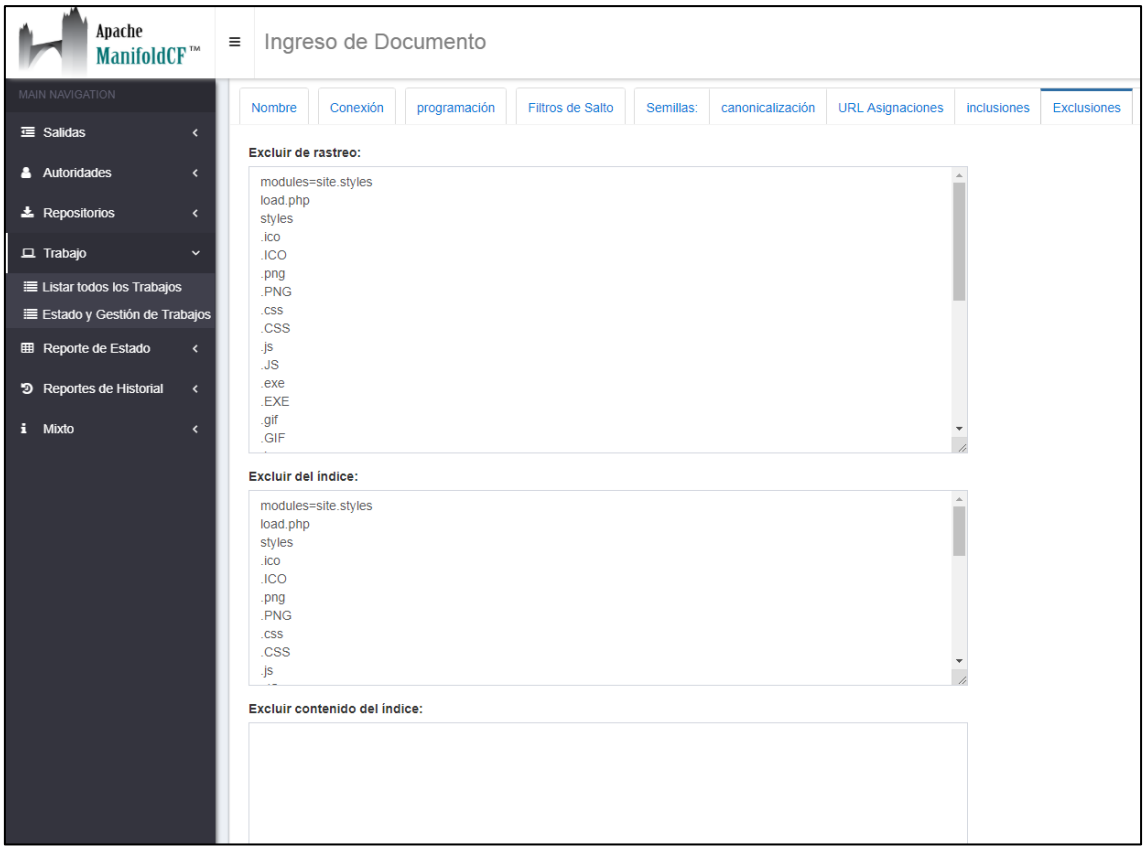


***Figura 64**. Pantalla de configuració d'exclusions*

Aquí es permet excloure elements de diferents formes:

- **Excloure del rastreig**: Salta les *url's* i no les indexa ni *crawleja* els links que contengui.
- **Excloure de l'índex**: Al igual que l'anterior, no indexa les *url's*, però si *crawleja* els links que conté.
- **Excloure contingut del índex**: Si el contingut havia sigut indexat amb anterioritat, (*crawlejos* anteriors) s'encarrega d'esborrar-los de l'índex *d'Elasticsearch*.

En el cas del prototip, hem escollit el següent llistat d'exclusions, comú per excloure tant del rastreig com de l'índex.

Llistat exclusions:

```
modules=site.styles
load.php
styles
.ico
.ICO
.png
.PNG
.css
.CSS
.js
.JS
.exe
```

.EXE
.gif
.GIF
.bmp
.BMP
.jpg
.JPG
.jpeg
.JPEG
.xls
.XLS
.xlsx
.XLSX
.peg
.PEG
.mp4
.MP4
.mp3
.MP3

***Figura 65**. Llistat d'exclusions*

Continuem amb la pestanya de “Seguridad”:

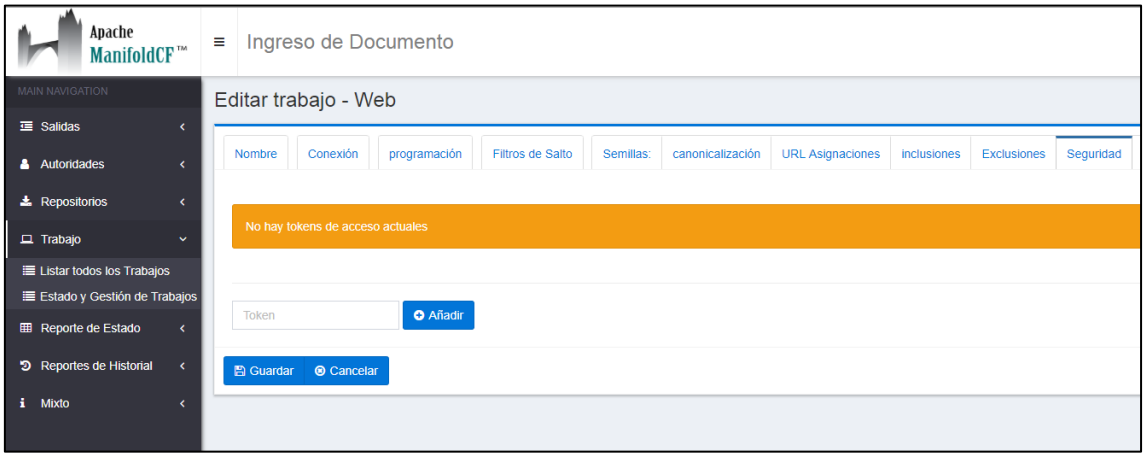


***Figura 66**. Pantalla configuració de tokens de sessió*

Hi ha moltes webs que per temes de seguretat requereixen de *tokens* per “autenticar” l’accés a aquestes. *ManifoldCF* permet afegir *tokens* d’accés per poder accedir a aquestes *urls*.

De cara al prototip, la *Wikipedia* no requereix de l’ús de *tokens*.

Passem a la següent pestanya “*Metadatos*”:

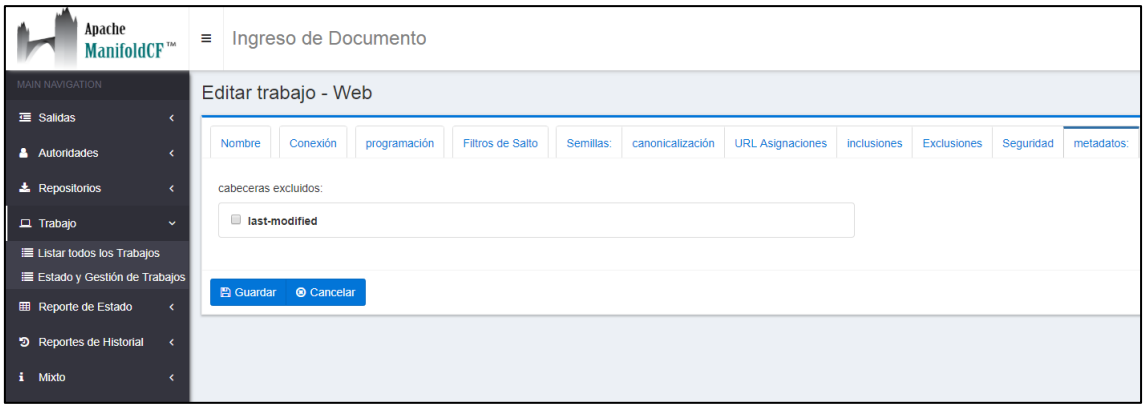


***Figura 67**. Pantalla d'exclussió de capçaleres*

Aquí es permet excloure la capçalera "*last-modified*" de la indexació a *Elasticsearch*. Al nostre prototip s'ha deixat aquesta capçalera com indexable.

Passem a la pestanya "*Trazar un mapa de campaña*":

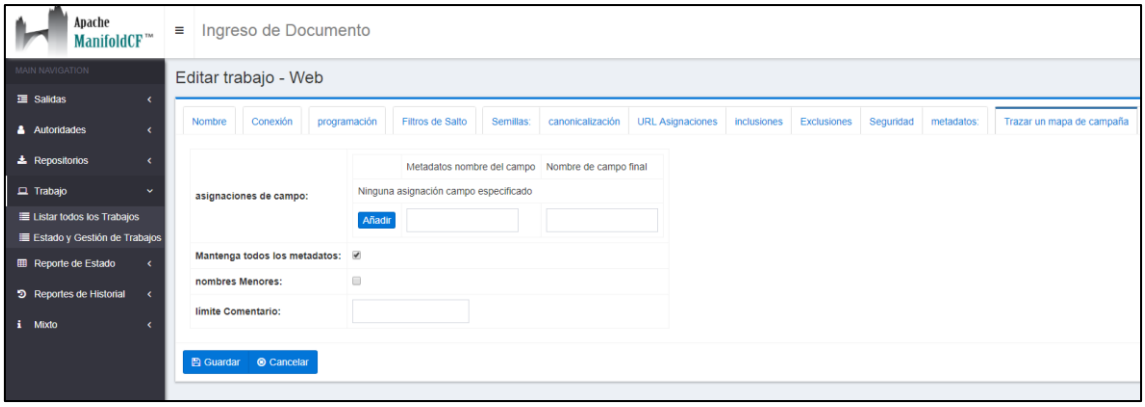


***Figura 68**. Pantalla de mapeig de metadades*

Aquí es permet *mapejar* els camps de les metadades si no es volen deixar amb el nom automàtic. També permet mitjançant expressions regulars, modificar el contingut d'aquests camps.

Podem trobar també la opció de mantenir totes les metadades que es puguin extreure o només les que encaixin amb les definides a l'índex de dades *d'Elasticsearch*. Posar els noms dels camps en minúscula...

En el cas del nostre prototip, només hem marcat la opció per mantenir totes les metadades.

Continuem a la pestanya "*Excepciones*":

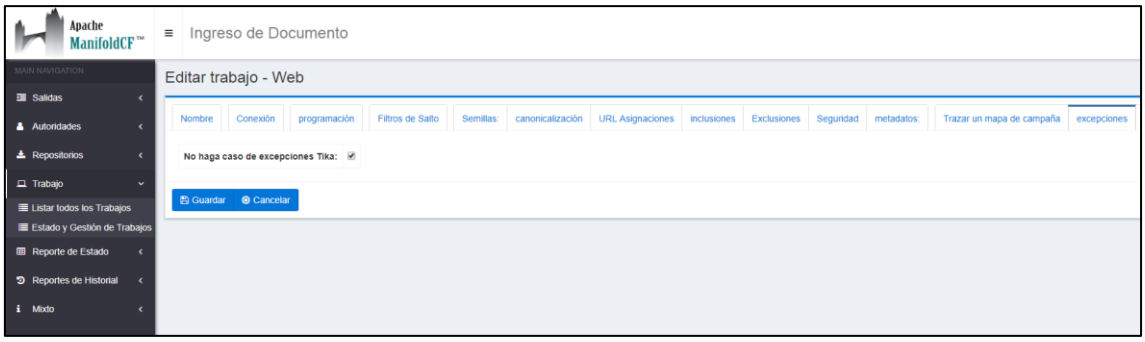


***Figura 69**. Pantalla desactivació d'excepcions de Tika*

Aquí podem excloure les excepcions que provoqui *Tika* al intentar extreure continguts. Això ens permetrà que el procés de *crawlejat* no pari quan salti una excepció a alguna de les *url*.

Finalment passem a la pestanya "*Repetitivo*":

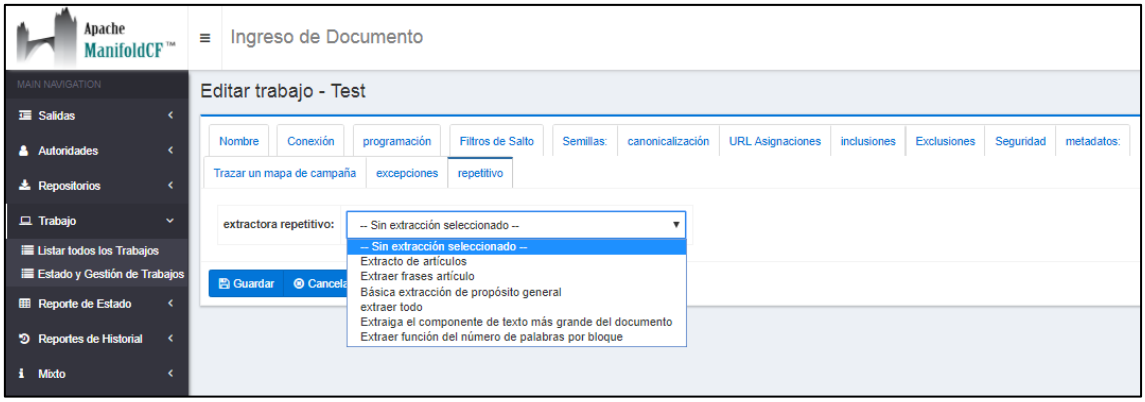


***Figura 70**. Pantalla de selecció de tipus d'extraccions*

Aquesta pestanya permet realitzar extraccions selectives. A la documentació de *ManifoldCF* es comenta que es un desenvolupament experimental i no es disposa de gaire informació sobre el funcionament, així que en el nostre prototip he deixat sense seleccionar aquesta opció ja que per defecte t'extreu tot el contingut detectat.

Finalment cal prémer el botó "*Guardar*" per guardar el "*job*".

Llençar un job

Un cop creats els *jobs*, per poder llançar-los de manera manual (en cas de no haver-los programat per auto executar-se), cal anar des del menú principal a "*Trabajo*" → "*Estado y Gestión de Trabajos*" i allà veurem el llistat de *jobs* que s'hagin generat:

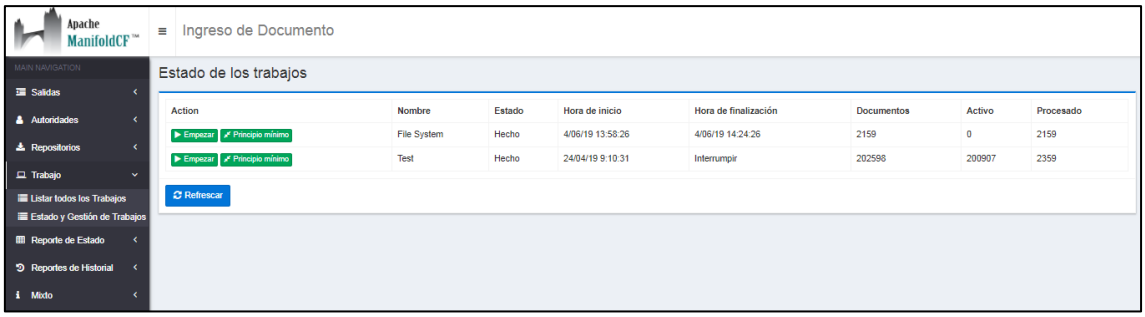


***Figura 71**. Pantalla d'estat dels jobs*

Per llançar el procés hi ha dos botons: "*Empezar*" i "*Principio mínimo*". "*Empezar*" llença el procés utilitzant tots els recursos possibles de la màquina. "*Principio mínimo*" llença el procés utilitzant els recursos mínims per poder mantenir el procés.

A la dreta podrem veure quants documents ha detectat, quants links estan actius en cua, pendents de ser processats, i quants ha processat (indistintament de si els ha exclòs o inserit a l'índex).

4. Afegir metadades a Word, PDF i Web

En el cas d'un *word*, es poden afegir anant a:

Archivo → Información → Propiedades → Propiedades avanzadas.

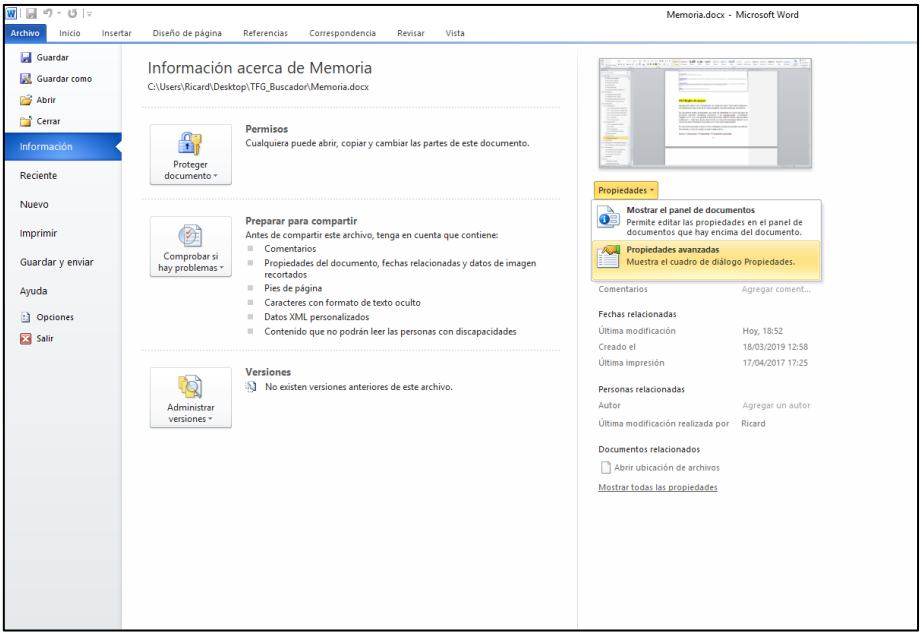


***Figura 72**. Pantalla de propietats del Word*

Un cop accedim a aquest panell, cal anar a la pestanya "*Personalizar*":

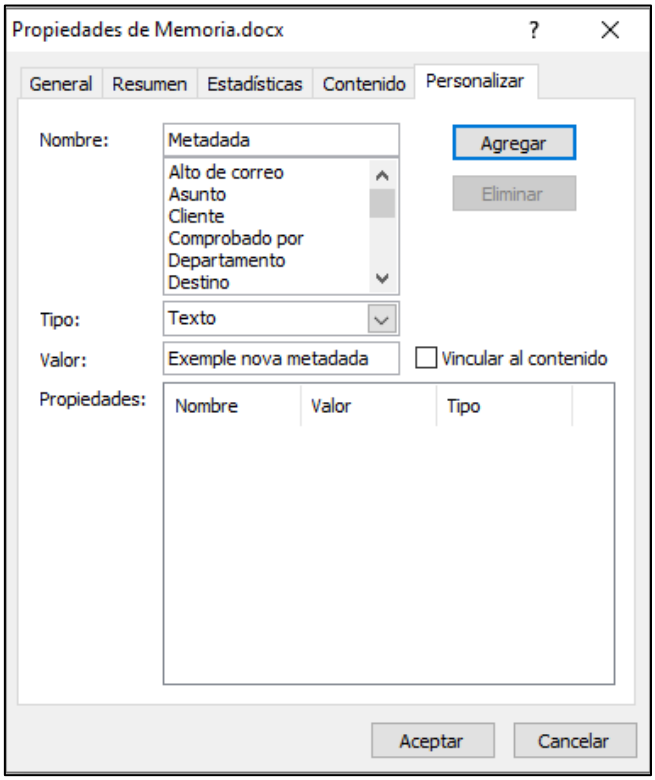


***Figura 73**. Pantalla de propietats de metadades del Word*

Aquí afegiríem un nom a la nova metadada (o escollim una metadada existent), un tipus de camp i escrivim el valor que volem afegir.

Així dons, en cas d'una universitat, si volguéssim afegir una assignatura relacionada, podríem afegir el camp assignatura i el nom d'aquesta al camp valor.

Aquestes metadades s'afegirien al document i al *crawlejar*-se s'indexarien al motor de cerca *Elasticsearch* com un nou camp.

En cas d'un PDF, es faria de la següent manera:

Archivo → *Propiedades* → *Personalizar*:

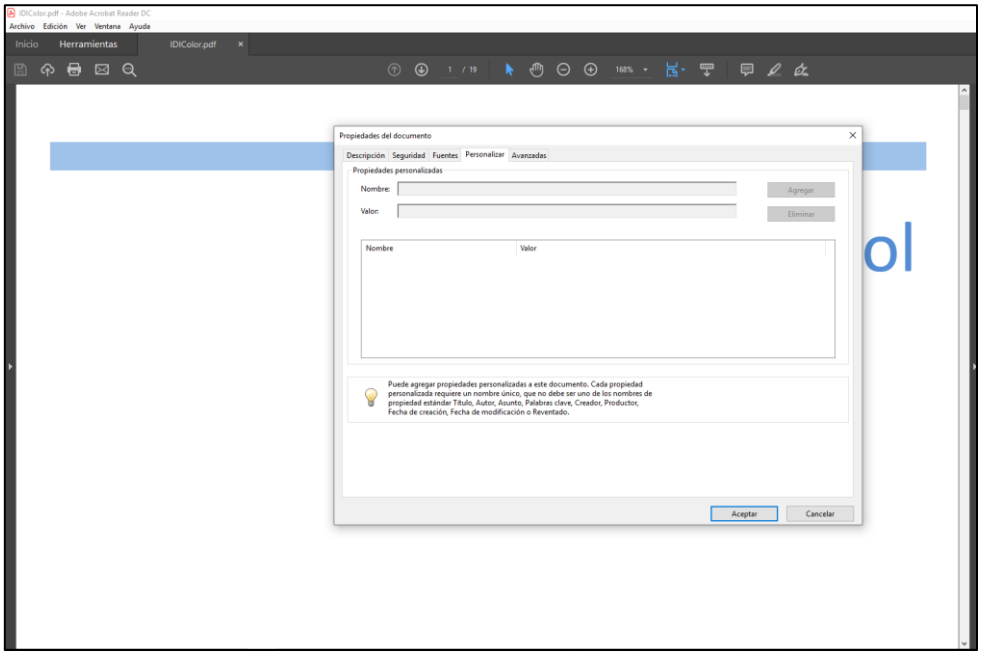


***Figura 74**. Pantalla de propietats de metadades de PDF*

Aquí afegiríem un nom a la metadada i un valor i clicaríem al botó "*Agregar*". Al igual que al cas anterior, aquestes metadades afegides s'indexarien automàticament a *Elasticsearch*.

En cas de les pàgines web, afegir una nova metadada es fa de la següent manera:
Al codi font de la web, cal afegir una etiqueta anomenada "*Meta*". Aquesta etiqueta necessita el camp "*name"* i el camp "*value"* o "*content"*:

Exemple de la pàgina principal de *Google*:

```
<!doctype html>
<html>
<!-- Copyright 2015 The Chromium Authors. All rights reserved.
     Use of this source code is governed by a BSD-style license that can be
     found in the LICENSE file. -->
<head>
  <link rel="stylesheet" href="chrome-search://local-ntp/animations.css"></link>
  <link rel="stylesheet" href="chrome-search://local-ntp/local-ntp-common.css"></link>
  <link rel="stylesheet" href="chrome-search://local-ntp/custom-backgrounds.css"></link>
  <link rel="stylesheet" href="chrome-search://local-ntp/doodles.css"></link>
  <link rel="stylesheet" href="chrome-search://local-ntp/local-ntp.css"></link>
  <link rel="stylesheet" href="chrome-search://local-ntp/theme.css"></link>
  <link rel="stylesheet" href="chrome-search://local-ntp/voice.css"></link>
  <meta http-equiv="Content-Security-Policy"
      content="object-src 'none';child-src chrome-search://most-visited/ https://*.google.com/ ;
YUYyUdaHPr4vPwR1MbbgywsO6nfhVmXMpy6ILGW3rKk=' 'sha256-ydzrIpu7RPpN+/W1u5M3gE6zl+u6KRMtbQEzdDt/Ua
jLVU+mwbF+2MzncAuPBL3AwyOEbKloRRFTaSqeUQsLE=';">
  <script src="chrome-search://local-ntp/animations.js"
      integrity="sha256-EVjR8L4xuo0c7UNzAtLmKQoojKIjdHqYIqdanwmukFQ="></script>
  <script src="chrome-search://local-ntp/config.js"
      integrity="sha256-jLVU+mwbF+2MzncAuPBL3AwyOEbKloRRFTaSqeUQsLE="></script>
  <script src="chrome-search://local-ntp/custom-backgrounds.js"
      integrity="sha256-ps+f+kvXv/saiMZBvZdqrnWV6Gcxo4dQ9UAG6SXPM6M="></script>
  <script src="chrome-search://local-ntp/doodles.js"
      integrity="sha256-YUYyUdaHPr4vPwR1MbbgywsO6nfhVmXMpy6ILGW3rKk="></script>
  <script src="chrome-search://local-ntp/local-ntp.js"
      integrity="sha256-ydzrIpu7RPpN+/W1u5M3gE6zl+u6KRMtbQEzdDt/Uao="></script>
  <script src="chrome-search://local-ntp/utils.js"
      integrity="sha256-KNY16QU2yK5dZbn5VtmVDt8Nko2v7x2ZPfUGq+DvZEs="></script>
  <meta charset="utf-8">
  <meta name="google" value="notranslate">
  <meta name="referrer" content="strict-origin">
</head>
<body>
  <div id="custom-bg"></div>
  <!-- Container for the OneGoogleBar HTML. -->
  <div id="one-google" class="hidden"></div>

```

***Figura 75**. Pantalla de metadades WEB*