Treball de Fi de Grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques
Universitat de Barcelona

FoodNet – Visualización de redes de alimentos

Marc Cardus Garcia

Directors: David Sánchez Pinsach
Daniel Villatoro Segura
Realitzat a: Departament de Matemàtica Aplicada i Anàlisi. UB

Barcelona, 20 de juny de 2014

Índice

0.1 Resum

El marc d'aquest projecte es situa en alguns dels camps més punter de la informàtica tal i com la mineria de dades o la visualització de dades. A la vegada, no només esta constextualizat en camps punters sinó que te com a propòsit tota una idea innovadora, crear una gran base de dades de cuida amb la capacitat de visualitzar la xarxa complexa que forma el tots els seus ingredients i a la vegada poder extreure informació de la xarxa d'una forma senzilla. Tot això situat en un entorn web utilitzant sempre tecnologies escalables a grans quantitats de dades. Tot això també dins del marc també de la unitat UB-Bullièdia.

L'anàlisi de grafs es tot un camp en expansió, cada dia son més les estructures trobades en forma de graf, com les xarxes socials. Igualment la mineria de dades i l'anàlisi de la informació en general son camps creixents dins la informàtica, degut al ritme de creixement de la informació. Això fa molt interessant doncs poder visualitzar grans conjunts de dades per poder fer anàlisis qualitatius d'una forma senzilla i còmoda. Mitjançant l'estructura en forma de graf es fa possible l'extracció fàcil de dades com els camins mínims, agrupacions, nodes centrals, nodes populars....

Per a realizat aquestes tasques es proposa la divisió del projecte en tres parts, l'obtenció de les dades, la seva nateja i per últim la visualització en l'entorn web. Totes i cada une de les parts resulta imprescindible per a obtenir una aplicació final completa.

Tot això doncs implica la construcció de un web crawler intel·ligent que sigui capaç de descarregar totes les receptes d'una pàgina web de cuina d'una forma prou genèrica, per així aconseguir omplir la base de dades. Sempre tot utilitzant eines escalables amb grans quantitats de dades en desenvolupant l'entorn web per fer la visualització una eina encara més senzilla. La pàgina web serà implementada amb el framework de Python Django.

Es pot trobar el resultat de la pàgina web a: www.foodnet.no-ip.org

0.2 Abstract

This project its about many of the most advanced fields in computer science as data mining and data visualization. Also it's pretended to create a new leading application about a innovative idea, develop a big cooking data base with a visualization module. This feature will allow to visualize the complex network created from all the food ingredients in the data base in order to make easier the data extraction in the food network. All of this working in a web site and always using scalable technologies for a good performance in big amounts of data scenarios. This project is contextualized in the UB-Bullipèdia Unit.

Graph analysis it's a growing field in computer sciences, every day more data is organized in network form, it's the case for example of the social networks. Also data mining and information analysis in general are growing fields because the data volumes are raising every day, it makes the data visualization really interesting for extract useful data and conclusions in a easy way. In addition, using data organized in a graph form is easy to extract interesting information as clusters, minimum paths, most central nodes and popular nodes...

To sum up, the project will be segmented in different goals, data collection like data cleaning and web visualization. Every single one of this parts i's so important for achieve the completely working application.

These goals imply to construct a smart web crawler in order to fill the data base downloading all recipes from any cooking websites, use scalable technologies with the purpose of don't lose performance with huge amounts of data, and make it all work in a web application for make easier to visualize the data. The website will be developed under the python framework, Djando.

The web page resultant from this project is available in the next link: www.foodnet.no-ip.org

0.3 Resumen

El marco de este proyecto se sitúa en algunos de los campos más punteros de la informática, como son la minería y la visualización de datos. A su vez también pretende ser puntero en el objetivo, crear una gran base de datos de cocina con la capacidad de visualizar la red compleja que forman todos sus ingredientes y a la vez poder extraer información sobre la red de forma sencilla, todo esto en una página web y usando tecnologías escalables a grandes cantidades de datos. A la vez todo contextuado en el marco de la unidad UB-Bullipèdia.

El análisis de grafos es un campo en expansión dado que cada día son más las estructuras de datos en forma de grafo en uso como las redes sociales. Igualmente la minería de datos y el análisis de la información en general son campos en auge, debido a que los volúmenes de datos se multiplican cada día, así pues es muy interesante poder visualizar este conjunto de datos para poder extraer conclusiones. Mediante la organización de todos los nodos en forma de grafo podemos extraer de forma sencilla datos muy interesantes como caminos mínimos, agrupaciones, nodos con mayor centralidad, con mayor popularidad...

Para lograr todo esto, dividiremos el proyecto en tres partes, obtención de datos, limpieza de los mismos y visualización en web. Cada una de estas partes acarrea una gran importancia dentro del proyecto, ya que todas ellas son imprescindibles para lograr crear una aplicación completa.

Todo esto implica la construcción de un *crawler* web inteligente para poder descargar las recetas de las páginas de cocina de forma genérica y así llenar la base de datos, usar herramientas que escalen bien según la cantidad de datos y además desarrollar todo esto como aplicación web para poder usar la visualización de forma sencilla. La página web será implementada en el framework de python Django.

Se puede encontrar la web resultante en el siguiente enlaze: www.foodnet.no-ip.org

1.0 Introducción

En este apartado se exponen los diversos elementos de introducción al proyecto, como la definición clara del problema y los objetivos para resolverlo, las motivaciones para realizarlo y el contexto en el que es desarrollado.

1.1 Definición del problema y objetivos

En la actualidad existen múltiples recetarios y múltiples fuentes de información sobre la cocina, pero falta una estandarización de esta y falta además una manera bien estructurada y organizada de agregar nueva información a la ya existente. Así pues como se comenta en el contexto del proyecto existen ideas en desarrollo para tratar de estandarizar la información sobre el mundo de la cocina, ingredientes, recetas, relaciones, propiedades, nutrientes, componentes químicos, cantidades... pero aun no existe un estándar.

Por otro lado es necesario poder aprovechar toda aquella información dispersa en la red que podemos encontrar en múltiples páginas de cocina, para así agruparla de una forma estructurada en una macro base de datos y así facilitar su estudio y manipulación.

A la vez que hay que pensar en como estructurar esta información para lograr la mayor facilidad posible en su estudio, manipulación y agregación, así como buscar herramientas que faciliten la comprensión de la misma.

Explicado el problema en el que nace el proyecto, queda definir unos objetivos realistas a cumplir para lograr resolver el máximo de problemas posibles.

1. La creación de un *web crawler* que hará un recorrido por todas las recetas de *cualquier web de cocina* para un procesado posterior.

2. Por otro lado se plantea una herramienta que permita procesar dicha información lograda por el *web crawler*. En este proyecto, nos centraremos en poder extraer los ingredientes participantes en la receta. Mientras que en posibles ampliaciones se podría atender al tipo de relación entre estos (cantidades de cada ingrediente) o a la preparación y cocina sobre los mismos.
3. Crear un sistema para poder ordenar y almacenar la información no procesada como la ya procesada siguiendo en esta una odontología y estructura clara para facilitar mecanismos de manipulación, inserción, extracción y análisis de los datos.
4. A su vez se pretende crear una herramienta que permita visualizar y analizar la información del sistema para así lograr un análisis cualitativo de una forma sencilla, accesible y rápida. En este proyecto se trabajará para lograr la base de dicha herramienta y proporcionarle las utilidades más básicas como la propia visualización y una herramienta de búsqueda integrada. Estas herramientas serán desarrolladas como aplicación web para facilitar su acceso y utilización.
5. Por último pero no menos importante, se entiende que al ser un proyecto ambicioso es necesario marcar unas pautas de ampliación claras en el caso de que bien la comunidad o algún particular quiera ampliar las funcionalidades del proyecto.

1.2 Motivaciones

Después de mucho tiempo pensando en la temática de mi trabajo de fin de grado vi que claramente me atraían aquellas opciones más relacionadas con las tecnologías punteras, en especial las relacionadas con la inteligencia artificial y la minería de datos. Después de no encontrar un tema en concreto, estuve analizando las diversas opciones propuestas por los tutores, entre las había diversas que me llamaban la atención. La mayoría de propuestas que me interesaron fueron las de tipo más aplicado al mundo científico, que me llamaron mucho la atención en un inicio, pero posteriormente creí que pesar de parecer muy interesantes, la mayor faena que conllevaban, eran tareas más mundanas, como normalizar entradas y salidas etc Con lo que me fije en la oferta propuesta por Daniel Villatoro, la que proponía una aplicación web para el análisis de redes complejas de alimentos, tema

relacionado con el *big data*, el análisis de grafos, la visualización de datos, desarrollo web y que podía tocar otros temas como inteligencia artificial e incluso minería de datos. Todo esto unido a mi cariño por la cocina hacía una propuesta atractiva que podría permitirme acercarme a campos en los que estaba interesado y ayudarme a entender la cocina, quizá de una forma distinta y más científica. Este trabajo además se situada en el marco dela unidad UB-Bullipèdia, que como se explica en el siguiente apartado nace del soporte de la Universidad de Barcelona al proyecto la Bullipèdia, proyecto que también enmarca interesantes iniciativas y concursos, lo que hace aun más atractiva la idea.

Finalmente, otra de las motiaciones que me han llevado ha trabajar en este proyecto, era la posibilidad de empezar desde zero. La idea de empezar a trabajar en un campo no muy explorado y con un proyecto nuevo me motivaba más qe continuar el trabajo realizado por otra persona; dado que me daba más libertad.

1.3 Contexto y estado del arte

1.1.1 El mundo de la cocina y laBullipedia

En los últimos años se está progresando para ver la cocina de una forma distinta, de una manera más científica, estructurada y más adaptada a las nuevas tecnologías en general, así pues podemos decir que se está tratando de convertir la cocina en una ciencia.

Para poder ver estos cambios hechos realidad, existen un conjunto de iniciativas e inversiones para investigar sobres otras formas y maneras de ver la cocina. La más emblemática es la B*ullipedia* [1], iniciativa por parte de el B*ullifundation*, organización dirigida por el reconocido y galardonado cocinero catalán Ferran Adrià, en la que se pretende crear una estructura clara en forma de enciclopedia en web al estilo de la conocida wikipedia pero con un enfoque total a la cocina. Así facilitar el aprendizaje tanto a nivel profesional como particular y tratando de facilitar tambien la participación de los usuarios para la adición de nuevos contenidos y así aportar nuevo conocimiento e innovación. Siempre de una forma clara y bien organizada que permita un fácil acceso a datos bien organizados.

¿Qué es la BulliPedia?

“Es un proyecto promovido desde el Bullitaller que conformará una parte de los objetivos de la fundación elBulliFoundation y que pretende, utilizando las nuevas tecnologías de la información y la comunicación, aportar a la sociedad, en general, y en el mundo culinario especializado, en particular, toda la información de calidad disponible sobre el fondo creativo relacionada con la cocina el Bulli, las bases académicas que permitan su total comprensión y la entrada de otros mundos creativos y de otras aportaciones que puedan ser identificadas como novedades de conocimiento que incorporen innovación.”

El proyecto de el *Bullifundation* además está acompañado de diversas iniciativas como el concurso H*acking Bullipedia* [2] en el que se propusieron una serie de retos para que los participantes propusieran soluciones para posteriormente, ser desarrolladas por los equipos ganadores en la división de I+D de telefónica.

Aquí tenemos el video [3] de una de las la propuestas ganadoras que se basa en convertir la cocina en un lenguaje de programación normalizado y así entender laBullipedia como un repositorio de código tal y como comentan ellos mismos:

*” **Proposal:** Huevo programing language We imagine cooking as a programming language.In this context we could understand Bullpedia as a code repository. In this work we want to propose a definition of “Huevo” a cooking programming language. How new programs will be created, how the will be stored in Bullpedia and how they will be compiled and executed in a kitchen by a group of cooks. ”*

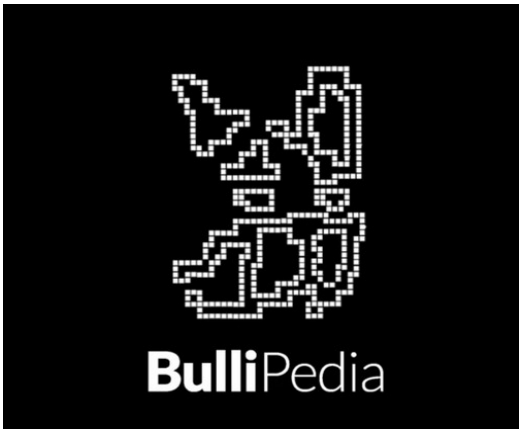


Figura número 1. Logo de la Bullièdia.

Entre las entidades participantes en la *Bullipedi*a caben destacar la participación de la UB en la unidad creada UB-Bullipèdia., marco en el que se situa el contexto e este proyecto.

1.1.2 Big Data

En la actualidad la cantidad de contenidos y datos no hace nada más que crecer , lo que hace la minería de datos y el big data, un tema interesante de estudio. Así pues podemos definir la minería de datos como aquel campo de las ciencias de computación que trata de descubrir patrones en grandes volúmenes de conjuntos de datos y tiende a utilizar métodos propios de la inteligencia artificial y grandes bases de datos.

Con el auge del uso de internet en general, el comerció electrónico y las redes sociales, es lógico pensar que cada día por internet deben pasar tamaños monstruosos de información, así pues según un articulo en la revista science [5] la capacidad tecnológica de internet crece vertiginosamente, en el caso de la capacidad de almacenamiento y de comunicación bidereccional el aumento anual es cercano al 25% mientras la tasa de capacidad computación de propósito general crece cercana al 58% anualmente. Con estos datos es fácil imaginar que la cantidad de información en internet es monstruosa, de hecho según IBM se estima que en 2012 cada día fueron creados cerca de 2,5 trillones de bytes de datos, con lo que el 90% de la información en internet ha sido creada en los dos últimos años.

1.1.3 Información grafos y visualización

Un mundo donde la información crece y crece a un ritmo frenético, esto fomenta el interés de tratar de ordenar dicha información para así poder ser procesada de una forma sencilla y simple. Así pues el interés en ordenar dicha información puede residir en lograr una mayor facilidad de consulta(accesibilidad, transparencia...), facilidad de análisis (comprensión, extracción de patrones...) o simplemente una mejor organización de la misma (facilidad en la inserción, modificación, generalización).

En el mundo real, existen cantidad de conceptos pero aun es mayor la cantidad de relaciones entre ellos (compra, conoce, gusta, trabaja...), lo que hace las estructuras en forma de grafo muy interesantes. Así pues la estructura en forma de grafo resulta muy útil ya que no solo muestra información (sobre los propios nodosI sino que también la de sus diversas relaciones, manteniendo así una estructura clara, sencilla y semejante al mundo real sin ser una estructura tediosa.

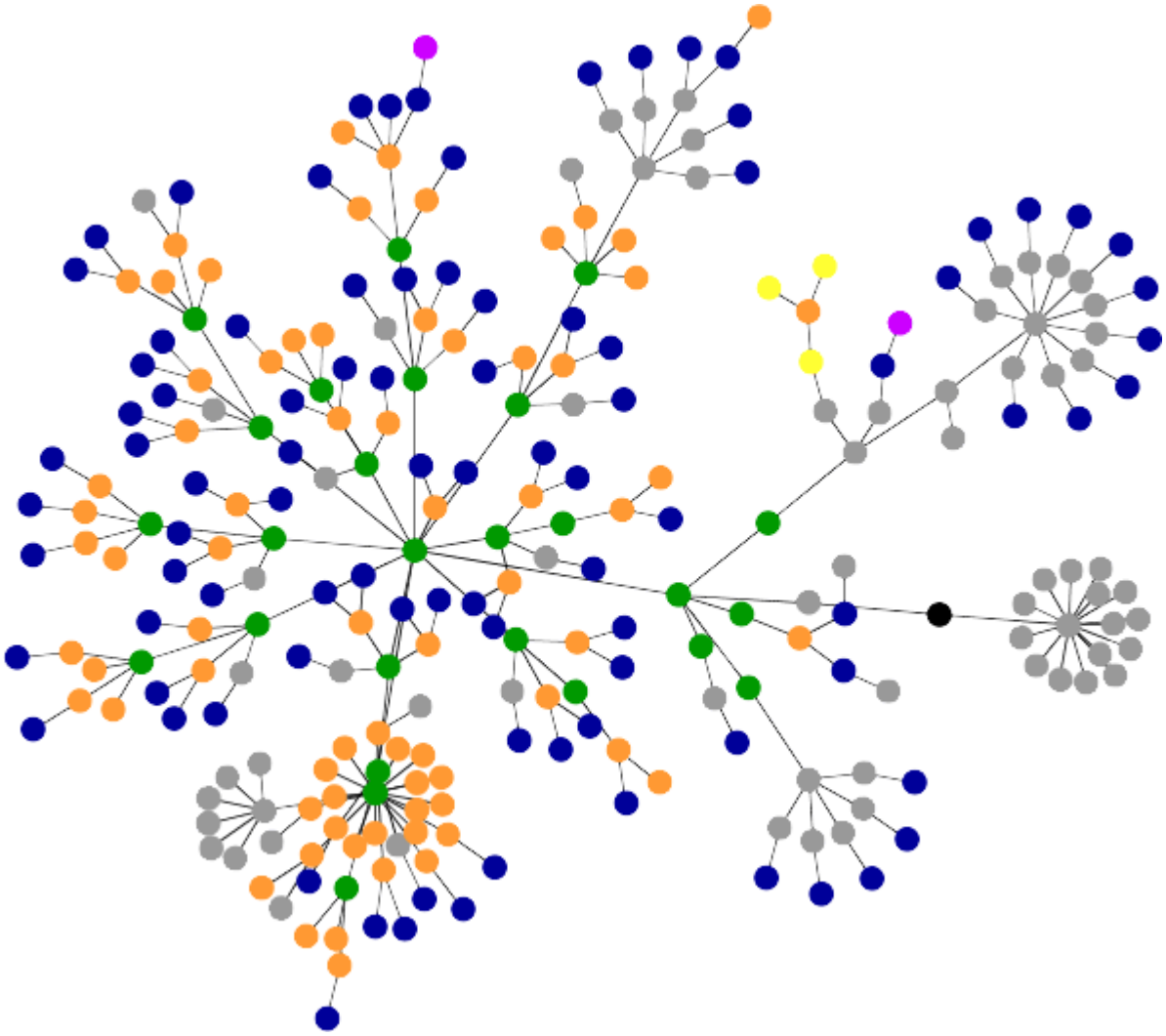


Figura número 2. Ejemplo grafo visualizado por la librería d3.js[7]

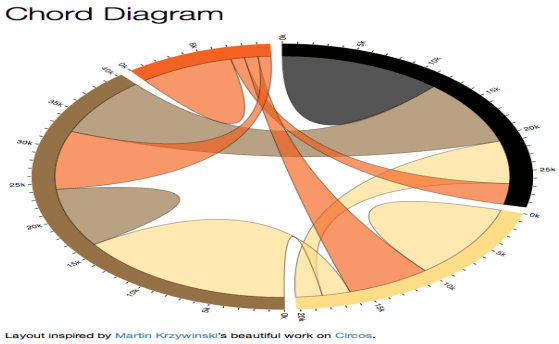
Todo esto hace que sea una estructura en auge actualmente[8], cuando vivimos en un mundo donde la información esta muy interrelacionada, más aun con la popularización de las redes sociales. Ejemplo son compañías como Twitter, que ha desarrollado su propia base de datos en forma de grafo [9].

Así pues la estructura en forma de grafo también tiene otras bondades, como la fácil aplicación de algoritmos sobre ella, como son diversos análisis de teoría de grafos: caminos mínimos, clusterización, máximos flujos, búsquedas, nodos con mayor centralidad... y además otros muchos algoritmos que son más fácil y intuitivos de aplicar sobre grafos como el famoso *Page Rank* [10] utilizado por Google.

Las estructuras en forma de grafo tienen otra característica interesante, son sencillas de visualizar, entrando en el campo de la visualización de datos podemos hacer visualizaciones interesantes done se aprecien a simple vista agrupaciones de nodos y patrones.

La visualización de datos pues claramente es otro campo en auge debido al aumento de la información y a que como dice el dicho, una imagen vale más que mil palabras, pues mediante una visualización correcta se pueden apreciar muchas características de un set de datos, mostrando información que en un primer momento desconocíamos como es el caso de lo cerca que están dos conceptos.

Figura número 2.
Visualización de palabras principales



Figúra número 3. Chord Diagram.

Existen pues muchos tipos de visualizaciones, a la vez que muchos tipos de softwares para la visualización de datos como Gephi [11] y librerías java Script como d3.js [8]. En este proyecto nos centraremos en la visualización de las estructuras en forma de grafos. Aun así convivimos a diario con múltiples tipos de visualizaciones que nos hacen comprender fácilmente grandes cantidades de datos.

1.1.4 Software Libre

La ambición de este proyecto hace que sea un elemento interesante para la comunidad, pues esta aplicación en si no deja de ser la construcción de una serie de herramientas básicas sobre las que se podrán añadir nuevas funcionalidades, para ello se subirá próximamente el proyecto a GitHub[12], para que la gente que este interesada pueda trabajar sobre el, así además será útil para añadir documentación sobre la implementación de la visualización o del web crawler.

Figura número 4. Logo de la comuniad de repositorio de software GitHub. Como curosidad, cabe recordar que uno de los planteamiento de como ver laBullipedia consite en una repositorio de recetas que podria asemejarse a GitHub

1.1.5 Antecedentes

Este proyectos abarca diversos conceptos muy distintos, así pues desenlosaremos los antecedentes en diversas ramas.

Por un lado existen diversos proyectos en el ámbito de las redes complejas y la visualización, como es el caso el caso de *The human desease network*[13,14], un trabajo

muy interesante sobre la relación de las enfermedades humanas, sobre el que se han realizado múltiples visualizaciones con múltiples tecnologías distintas[15,16,17]

Por otro lado la misma BulliPèdia va por un camino paralelo al de este proyecto, tratar de normalizar y almacenar la información sobre la cocina en un solo punto y adaptarla a las nuevas tecnologías. Este proyecto pretende ser pues un grano más de arena en este campo, aportando ideas interesantes como el web crawler o la herramienta de visualización en web.

También en el apartado de redes complejas y visualización existe un proyecto muy interesante llamado *Flavor network and the principles of food pairing[18]*, que es un análisis gastronómico sobre la cocina Coreana y Estadounidense, también se analizan componentes químicos en común de los alimentos.

Como punto diferenciador con mi proyecto, mi tesis tiene un ámbito más general donde conviven varias finalidades, no solo el análisis gastronómico y la visualización, sino que se trabaja para estandarizar y almacenar la información gastronómica a largo plazo, de una forma nueva más adaptada a las tecnologías de la información, dando le una gran facilidad de uso al proyecto. Además mediante el entorno web y dejando el camino abierto a la comunidad para que trabaje sobre el mismo proyecto.

2.0 Análisis de requisitos

A continuación mostraremos el análisis de requisitos realizado, en el que se examinan requisitos funcionales, no funcionales y casos de uso.

2.1 Requisitos funcionales

Los requisitos funcionales son aquellos que definen un sistema o sus componentes. Estos han de ser explicitados de una forma clara y son descritos como un conjuntos de entradas, comportamientos y salidas.

A continuación se muestran los requisitos funcionales identificados principalmente a partir del diagrama de casos de uso.

1. El usuario podra introducir los datos necesarios para el crawler (dominio, url de la receta y contenido)
2. El sistema podrá parsear mediante el crawler las webs introducidas por el usuario
3. Cada url parseada quedará almacenada en la base de datos, al igual que las recetas encontradas.
4. Una vez el crawler encuentre una receta, extraerá sus ingredientes y almacenará la receta ya limpiada y sus ingredientes.
5. El sistema solo almacenará aquella información que no exista en la base de datos
6. Los usuarios podrán ver la visualización del grafo
7. Los usuarios podrán buscar ingredientes en la visualización del grafo
8. Los usuarios podrán seleccionar nodos del grafo para que el sistema muestre sus vecinos a distancia uno.
9. El usuario podrá hacer zoom a través del grafo
10. El usuario podrá leer la información sobre el proyecto disponible en la web
11. El usuario podrá acceder a la página de administrador
12. El sistema pedirá credenciales a los usuarios que entren en la página de administrador
13. El usuario en la página de administrador ya acreditado podrá realizar funciones básicas en la base de datos:
    - Actualizar el grafo
    - Cargar en la base de datos los archivos de ingredientes

2.2 Requisitos no funcionales

Los requisitos no funcionales son aquellos que no describen el comportamiento de un sistema, quiere decir todos aquellos requisitos que no describen información a guardar ni funciones a realizar.

Los requisitos no funcionales encontrados son los siguientes.

1. Rendimiento y escalabilidad: Como se ha comentado en los objetivos, es importante que el sistema sea escalable en términos de rendimiento, es decir, su rendimiento no puede verse gravemente afectado en grandes cantidades de datos, pues uno de los fines del proyecto es poder soportar muchísima información en una vista a largo plazo del proyecto.
2. Accesibilidad: La aplicación ha de ser fácil de usar y accesible. Para ello se ha considerado crear el sistema como una aplicación web, para hacer más liviano el uso de la misma sin tener que instalar software adicional.
3. Bases de datos: Es requisito que la base de datos preferencia la lectura para una mayor facilidad en la visualización y en el análisis. Además debe mantener un modelo claro pero flexible
4. Usabilidad: Igual que en el punto anterior, es importante que la aplicación sea usable, sencilla de utilizar para cualquier persona sin necesidad de demasiada experiencia en el mundo web.
5. Mantenibilidad: Igualmente que en los objetivos, esta aplicación tiene una gran visión de largo plazo, consecuentemente ha de ser fácil de añadir nuevas funcionalidades a la misma, al igual que cambiar o ampliar algunas de las ya existentes.

2.3 Casos de uso

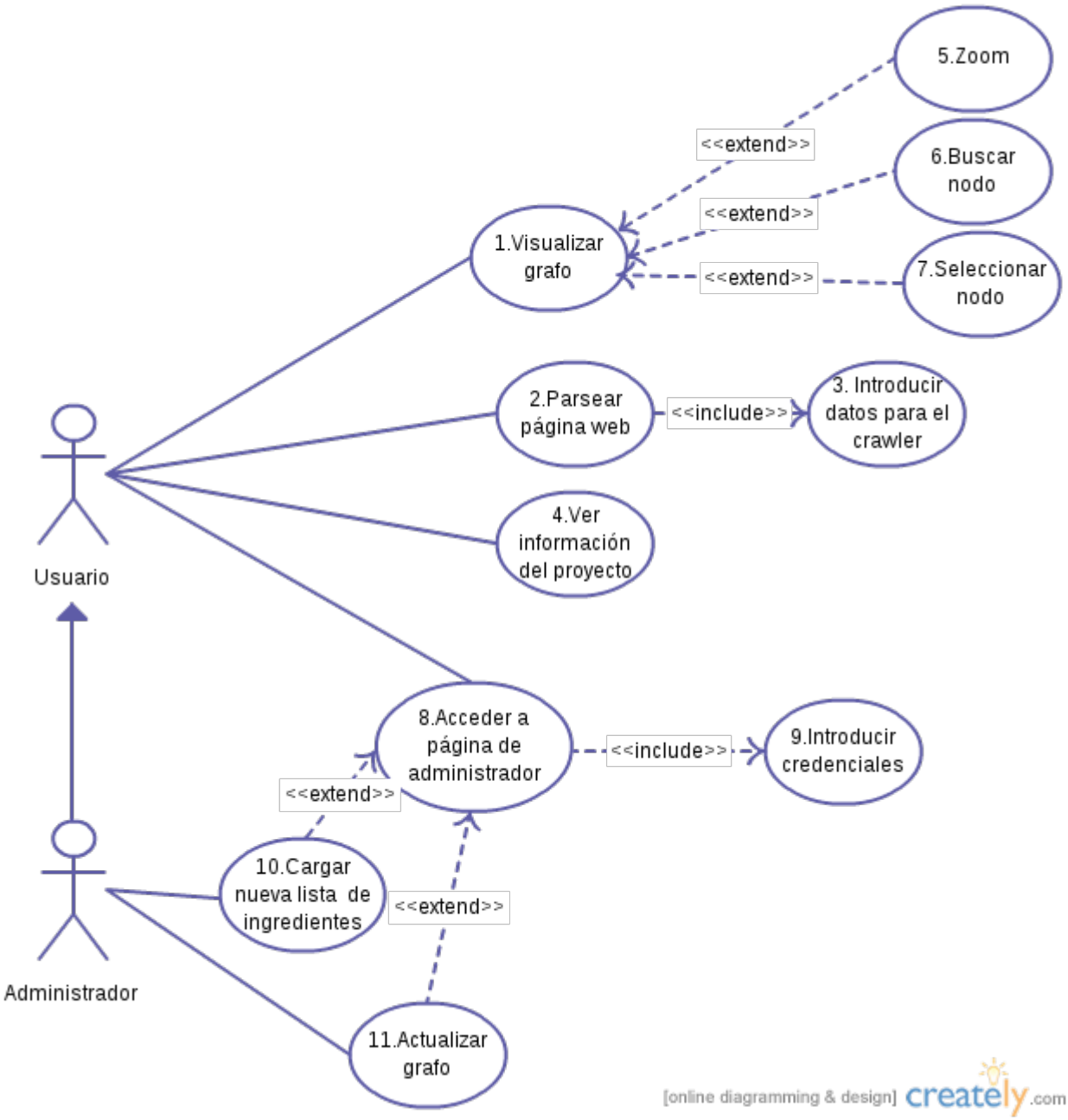


Figura número 5. Casos de uso

2.4 Casos de uso textuales

DC 1. Visualizar grafo
Descripción: Visualización del grafo por parte del usuario
Actor principal: Usuario
Flujo básico

1. El usuario accede a la url de la visualización del grafo 2. Se visualiza el grafo en la web
Flujo Alternativo 1. En el caso que la url sea incorrecta, aparecera el mensaje de error pertinente

DC 2. Parsear página web
Descripción: Parseo web por parte del sistema mediante los datos introducidos por el usuario.
Actor principal: Usuario
Flujo básico 1. El usuario accede a la url del crawler 2. El usuario pasa a introducir y enviar los datos del crawler 3. El sistema parsear el portal web entero
Flujo Alternativo 1. En el caso que la url sea incorrecta, aparecerá el mensaje de error pertinente 2. El usuario podrá ver el vídeo de ejemplo para no tener dudas en como introducir los datos. 2. Los datos introducidos son incorrectos y el sistema muestra un mensaje de error

DC 3. Introducir datos para el crawler
Descripción: Acceso a la página de información del proyecto
Actor principal: Usuario
Precondiciones: La página actual deberá de ser la del crawler
Flujo básico 1. El usuario introduce el dominio, la web de la receta i el texto de la receta 2. El usuario pulsa enter o el botón parsear 3. El sistema parsear el portal web entero
Flujo Alternativo 2. Los datos introducidos son incorrectos y el sistema muestra un mensaje de error

DC 4. Ver información del proyecto
Descripción: Acceso a la página de información del proyecto

Actor principal: Usuario
Flujo básico 1. El usuario accede a la url del crawler 2. El usuario visualiza la página de información sobre el proyecto
Flujo Alternativo 1. En el caso que la url sea incorrecta, aparecera el mensaje de error pertinente

DC 5.Zoom
Descripción: Parseo web por parte del sistema mediante los datos introducidos por el usuario.
Actor principal: Usuario
Precondiciones: La página actual deberá de ser la de visualización del grafo.
Flujo básico 1. El usuario hace zoom en el grafo moviendo la rueda del raton 2. El sistema zon el zoom correspondiente al movimiento

DC 6.Buscar nodo
Descripción: Búsqueda de nodos en la visualización del grafo
Actor principal: Usuario
Precondiciones: La página actual deberá de ser la de visualización del grafo.
Flujo básico 1. El usuario escribe el nombre del nodo a encontrar 2. El usuario ve una lista de los posibles resultados auto-completados 3. El usuario pulsa enter o el botón de búsqueda 4. El sistema muestra el nodo seleccionado y sus vecinos de distancia 1
Flujo Alternativo 4. La caja de búsqueda parpadea en color rojo haciendo ver que no existe tal nodo

DC 7. Seleccionar nodo
Descripción: El usuario selecciona uno de los nodos del grafo
Actor principal: Usuario
Flujo básico

1. El usuario selecciona uno de los nodos 2. El sistema muestra ese nodo y sus vecinos a distancia uno.

DC 8. Acceder a la página de administrador
Descripción: El usuario que supuestamente es administrador accederá a la página *oculta* de administrador
Actor principal: Usuario
Flujo básico 1. El usuario accede a la url *oculta* de administrador 2. El usuario podrá elegir seleccionar alguna de las funcionalidades de a página 3. El sistema realizara la función seleccionada en caso del usuario
Flujo Alternativo 1. En el caso que la url sea incorrecta, aparecerá el mensaje de error pertinente 2. El usuario no selecciona ninguna funcionalidad. El sistema pues no reacciona.

DC 9. Introducir credenciales
Descripción: El usuario introduce sus credenciales como administrador
Actor principal: Usuario
Precondición: La página actual deberá ser la de administrador
Flujo básico 1. El usuario introduce sus crredenciales 2. El usuario envia sus credenciales 3. Si estas son correctas, visualiza la página de administrador
Flujo Alternativo 2. Las credenciales son incorrectas, se muestra mensaje de error.

DC 10. Cargar lista de ingredientes
Descripción: El usuario carga una nueva lista de ingredientes
Actor principal: Administrador
Precondiciones: La página actual deberá de ser de ser la de administrador.
Flujo básico 1. El usuario selecciona el botón para cargar la nueva lista de ingredientes

2. El sistema carga la nueva lista de ingredientes
Flujo Alternativo 2. El archivo no existe, aparece un mensaje de error.

DC 11. Actualizar grafo
Descripción: El usuario actualiza los el grafo
Actor principal: Usuario
Precondiciones: La página actual deberá de ser de ser la de administrador.
Flujo básico 1. El usuario selecciona el botón para actualizar el grafo 2. El sistema actualiza el grafo con los nuevos datos de la base de datos.

3.0 Diseño

3.2 Base de datos y ontología

Las bases de datos y su gestión, como se comprende en el tercer objetivo del proyecto, es una parte vital del proyecto.

“Adicionalmente se pretende crear un sistema para poder ordenar y almacenar la información no procesada y la ya procesada siguiendo en esta una odontología y estructura clara y facilitar mecanismos de manipulación, inserción, extracción y análisis de los datos.”

Una vez capturados los requisitos de la base de datos, es importante hacer un análisis en algo más de profundidad y hacer el diseño consecuente.

Requisitos no funcionales de la base de datos:

“Bases de datos: Es requisito que la base de datos preferencia la lectura para una mayor facilidad en la visualización y en el análisis. Además debe mantener un modelo claro pero flexible”

Cual es entonces la naturaleza del sistema de almacenamiento de este proyecto?

1. Datos muy interrelacionados (los ingredientes participantes en las distintas recetas estarán muy interconectados entre si)
2. El modelo de datos ha de ser claro, pero flexible, pues a medida que el proyecto sea ampliado nuevos campos serán añadidos y en un futuro puede que los tipos de campos de las recetas puedan variar.
3. Siempre es interesante escribir rápido, pero en este caso es de mayor importancia poder leer rápido, pues no hay demasiado problema para escribir, mientras que para realizar análisis de los datos con eficiencia es importante poder leer con rapidez.
4. Es importante mantener tanto datos no procesados (urls parseadas o recetas crudas, es decir recetas que aun no han sido limpiadas) como los ya procesados, pues con el almacenamiento de datos no procesados se puede hacer un nuevo postprocesado adaptado de forma más rápida.
5. La estructura de los datos en la medida de lo posible ha de facilitar los análisis que puedan surgir.

La ontología de almacenamiento será la siguiente:

- Documento Html
    1. Url : String
    2. Xpath : String

Por un lado se almacenarán todas las url visitadas con sus Xpath, para facilitar un procesado futuro, sin necesidad de tener que volver a extraer las url.

- RecetasCrudas
    1. Url : String
    2. Nombre :String
    3. Ingredientes :StringList

Por otro lado se guardaran las recetras crudas, es decir, aquellas que aun no se han limpiado y contienen datos como cantidades, ingredientes...

- Recetas

1. Nombre : String
2. Url : String
3. Ingredientes : Objetos

- Ingredientes

1. Nombre : String
2. Grupo : String
3. Identificador : Int
4. Entradas : Int

- Lista de ingredientes

1. Identificador : Int
2. Nombre en Español : String
3. Nombre en Inglés : String
4. Grupo : String

Este último elemento es una lista de alimentons, unos 668, que será últil para la limpieza de los datos y en ampliaciones futuras, ya qye la fuente original tambien contienen valores nutritivos[19]

3.5 Planificación

Durante el desarrollo del proyecto, se ha trabajado de una forma *scrum like* [20], haciendo una o más reuniones por semana tutor-alumno para revisar el trabajo realizado desde la última reunión, discutir los progresos y crear objetivos para la siguiente reunión, siguiendo

como se ha comentado una filosofía semejante a la del desarrollo ágil de scrum contrapuesto a los métodos secuenciales tradicionales de desarrollo.

La mayoría de las reuniones han sido presenciales, a excepción de las realizadas pasado enero, que se han hecho por S*kype* debido a mi estancia fuera del país.

Así pues la experiencia de desarrollo basada en metodologías ágiles ha sido satisfactoria debida a la buena predisposición y al continuo contacto profesor-alumno.

4.0 Implementación

4.1 Tecnologías y introducción a la implementación

Después de realizar en análisis de requisitos y tener la idea clara del proyecto, se empezó a decidir las tecnologías adecuadas para el desarrollo del proyecto, llegando a la conclusión que el lenguaje de programación más adecuado para la aplicación es Python con su framework web Django. A la vez como bien se desarrollara posteriormente se decidió usar bases de datos NoSql MongoDB y Neo4j para el almacenamiento de la información. Paralelamente para el desarrollo web se ha utilizado html, css, java script, jQuery y Ajax mientras que para el desarrollo de la visualización del grafo se ha utilizado la librería java scipt sigma.js. Cabe destacar también que el entorno elegido para editar el código ha sido SublimeText 2 con su plugin SFTP para trabajar de una forma cómoda con el servidor. Otras tecnologías también son dignas de mención, como Firebug para el testeo web o LibreOffice para la redacción de esta memoria.

4.2 Python

La primera pregunta a resolver consiste en decidir en lenguaje, así pues, ¿Por qué Python?

Python[21] es un lenguaje de programación interpretado multiplataforma nacido a en 1991, su filosofía está basada en hacer códigos legibles eliminando todo aquello que pueda ser superfluo, así pues es un lenguaje adecuado para el *scripting*, en el que con pocas lineas de código podemos conseguir mucho a diferencia de otros más densos como Java o C++. Python además goza de tener una buena comunidad y múltiples librerías con funcionalidades muy diversas.

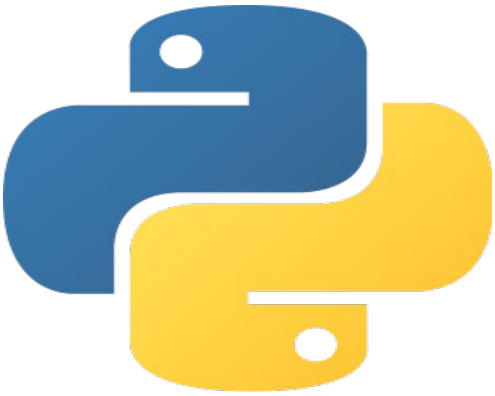


Figura 6. El logo de Python es la en python, serpiente pitón (Python en ingés)

```
1
2  def fib(n):
3      if n == 0:
4          return 0
5      elif n == 1:
6          return 1
7      else:
8          return fib(n-1) + fib(n-2)
9
10 #Call
11 print fib(5)
```

Figura 7. Fibonacci recursivo implementado python. Se puede apreciar fácilmente la sensillez de su implementación

Todo esto hace que Python sea un lenguaje muy adecuado, no para enormes proyectos con decenas de clases, donde algún lenguaje como java transmite un mayor sensación de orden sino más bien para el desarrollo de códigos con funcionalidades complejas donde se agradece tener el mínimo código posible para lograr mayor legibilidad, siendo así ideal para la implementación de algoritmos, centradose en los propios objetivos evitando mucho código superfluo para dicho fin.

Como de desarrolla más tarde, Python tambien ofrece utilidades en web mediante potentes frameworks como Django, lo que es muy importante para la implementación de dicho proyecto.

Así pues debido a que este proyecto exige faenas más cualitativa que cuantitativas en el sentido de que no se necesita mantener decenas de clases como bien se puede apreciar en el diagrama de clases, sino que se necesita más bien codificar procesos complejos a nivel

de flujo para el desarrollo del crawler o de la limpieza de datos. Se ha concluido que Python facilitará la implementación de dichos procesos sin la necesidad de escribir más código del necesario con la ayuda demás de librerías muy interesantes en el ámbito del web parsing.

4.3 Crawler

Como ya se ha explicado en los objetivos, el primer elemento a desarrollar es el web crawler, que deberá de poder descargar las recetas de casi cualquier página web de cocina solo a partir de unos mínimos *inputs* por parte del usuario.

Pero aun así definamos que es un web crawler.

Un Web crawler es un tipo de programa (llamado bot en ingés) con la capacidad de inspeccionar páginas web de forma metódica y automatizada.

Entonces el comportamiento del crawler es simple, nada más que un programa que navega en este caso por las diversas urls de una web de cocina con el objetivo de alcanzar todas las url de dicha web y encontrar por cada url si esta contiene o no una receta. Receta que posteriormente pasara a recibir una serie de procesos.

4.3.1 Investigación inicial sobre la implementación del crawler

La investigación sobre como hacer el crawler posible empezó estudiando la manera de poder localizar en aquellas url por las que navega el crawler si existe una receta o no y en casa aformativo extraerla, todo eso a partir de algún input por determinar del usuario.

Para ello se empezo por ver y analizar como estaban distribuidas las recetas en las páginas web, si existía algún patrón que poder utilizar para poder capturarlas bien en las urls o en los códigos html. Además había que concretar que inputs por parte de los usuarios eran los necesarios.

La primera observación fue descubrir que no todas las recetas estaban en un mismo tipo de url como *www.dominio.com/recetas/...*, sino que en la mayoría de las webs estas estaban estas dispersas por múltiples tipos de url.

La segunda observación fue ver que en muchas webs las recetas estaban situadas en algún o bullet point de html con alguna clase css determinada , pese que en otras webs esto no se cumplía e incluso en las mismas webs a veces este patrón se cumplía y otras no. Lo que hizo pensar en expresiones regulares para encontrar un patrón que poder aplicar a las páginas web con un entrenamiento previo, idea que se descarto porque el entrenamiento parecía inviable, ya que cada página web era muy distinta como para extraer una expresion regular suficientemente genérica.

El gran descubrimiento fue ver, que a partir de unos inputs muy pequeños del usuario: la url de una receta en concreto y el contenido de una receta, se puede extraer el XPath[22] de la receta. Después de hacer varias pruebas se comprobó que el XPath de una determinada receta en una determinada web se puede extrapolar a la mayoría de recetas de dicha web.

Así pues el XPath[22] exáctamente es un lenguaje que permite construir expresiones que recorren y procesan un documento tipo XML como el html. La idea del XPath es muy parecida a la de las expresiones regulares, para seleccionar partes de un texto sin atributos. El XPath pues permite usar y seleccionar teniendo en cuenta la estructura jerárquica del XML.

Wildcard	Description
*	Matches any element node
@*	Matches any attribute node
node()	Matches any node of any kind

Figura 8. Ejemplo sintaxis del XPath Figura.

$$[''//dd[@itemprop =' ingredient']/text()'']$$

Figura 9. Ejemplo XPath para las recetas de http://canalcocina.es

4.3.2 Tecnologías del crawler

Paralelamente a la investigación que se comenta en el apartado superior, se medito cual podría ser la tecnología indicada para la implementación del sistema de navegación automático (*bot*) del crawler a traves de las url de una página web determinada.

Despues de cierta busqueda, se descubrieron dos herramientas para realizar tal propósito.

1. Scrapy [23]
2. BeautifulSoup [24]

En el caso de Scrapy fue la decisión inicial, ya que no es una librería para parsear sino todo un framework, lo que comporta una curva de aprendizaje algo mayor, pero a la vez una forma muy cómoda de trabajar y con una muy buena comunidad con ejemplos y buena documentación.

El problema con Scrapy apareció al obtener los primeros resultados, pues solo llegaba a cierta profundidad de la web, con lo que no obtenía todas las urls de la página web. Despues de estar tiempo intentando lograr mayor profundidad en la web, vi que había gente tambien con problemas parecidos, con lo que decidi probar BeautifulSoup.

BeautifulSoup resulto mucho más sencillo de utilizar, al ser solo una librería se hacia muy facil de manejar, además es una librería muy intuitiva, conseqüentemente tiene una menor curva de aprendizaje. Documentación y ejemplos correctos. Parecía una buena decisión pues parecía que lograba pasear sin el problema de profundidad de Scrapy, el problema llego cuando descubri el XPath como técnica para lograr encontrar las recetas, pues pese a la fácil utilización de BeautifulSoup, sus selectores no trabajaban con lenguaje XPath, con lo que BeautifulSoup paso a ser inviable.

Despues dee un poco más de búsqueda logre encontrar lxml [17], una librería para Python escrita en C con múltiples herramientas para el tratamientos de documentos XML como el parseo, que además tambien se puede utilizar en documentos Hml.

Curiosamente los selectores de Scrapy están basados en lxml justo por ello pueden usar expresiones XPath, expresiones regulares...

Figura 10. Logo de Lxml

Lxml ofrece herramientas para el parsero y la sereialización de decoumentos XML, que tambien pueden usa ser usadas para documentos Html. Lxml además es rápido[17] y como comenta en su propia web, tiene puntos a favor y en contra con BeautifulSoup, pero lxml tiende a superarlo en casos generales[18]. A nivel de sintaxis, lxml es algo más desagradable en comparación con las tecnologías analizadas previamente, lo mismo ocurre en la documentación. Pero lxml resuelve los problems encontrados hasta el momento, pues no hay conflictos con la profundiad y sus selectores son compatibles con XPath, además de ser igual o más rápido que los anteriores.

4.3.3 Funcionamiento del crawler

Una vez encontrada la tecnología para automatizar la obtencion de las url de la página y obtenido el método de localización de recetas, llega el momento de implementar crawler como tal.

Para codificar el crawler se ha creado una clase llamada Crawler en crawler.py.

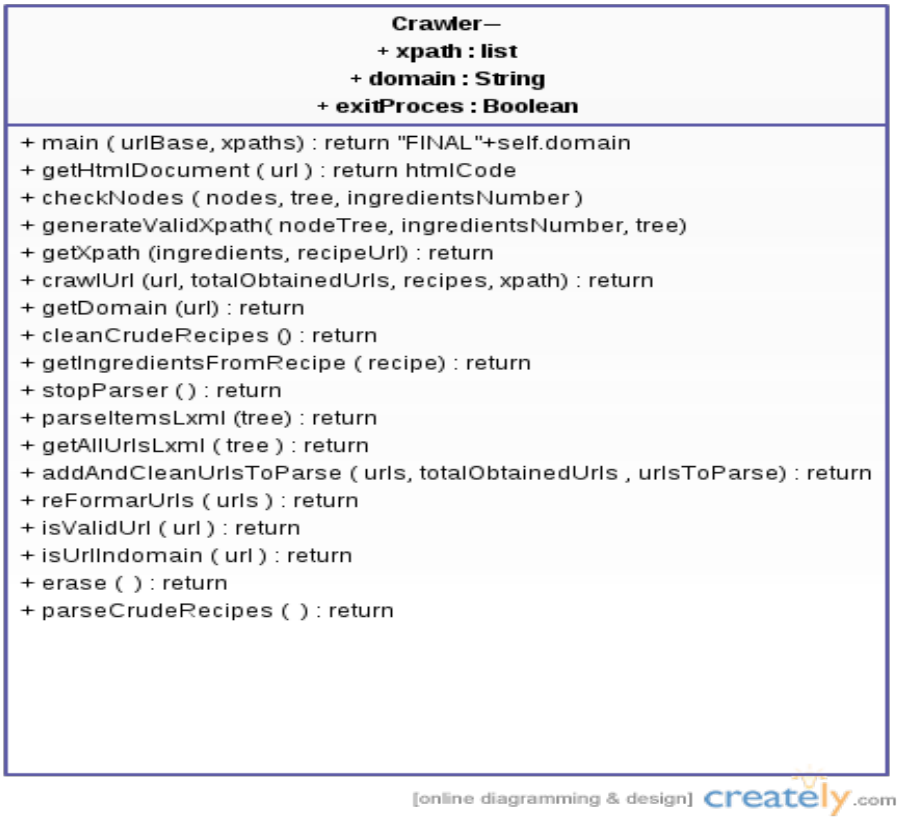


Figura 11. Crawler.py

En esta clase existen múltiples métodos y diversos atributos para hacer la implementación posible. Pero primeramente nos centraremos en el primer paso para desarrolllar dicho crawler, conseguir el XPath, para ello existe un flujo iniciado con el método *getXPath.* El funcionamiento se basa simplistamente hablando en tratar de encontrar un rasgo diferenciador, normalmente una clase css junto a un tag y si no se da el caso, a un conjunto de tags que hace de patrón.

Una vez se ha llamado al main y se ha logrado conseguir el XPath de la página web se empieza con el parseo de dicha página. Este parseo empieza con el método *crawlUrl* que es la base de la clase, ya que inicia la obtención automática de urls. Se ilustra el funcionamiento del método con un flujo descrito en la siguiente figura [Figuras Anexo, 1].

4.4 Liempieza y procesamiento de datos

Una vez lograda la implementación del Web Crawler, hay que implementar el procesamiento de los datos proporcionados por el mismo Crawler, pues hay que limpiarlos y almacenarlos.

Para limpiar los datos se ha usado un método sencillo, que consiste en iterara a traves de la la frase (por ejemplo “100g de harina”), haciendo grupos de palabras, primero grupos mas grandes y luego más pequeños, hasta que coincide que algún ingrediente registrado en la lista de ingredientes (lista de unos 600 alimentos)

4.5 Implementacion bases de datos

Como ya se ha comentado tanto en los requisitos como en el apartado de diseño sobre las bases de datos, principalmente queremos unas bases de datos fáciles de leer, flexibles, escalables y que soporten datos muy interrelacionados.

Para cumplir estos requisitos se ha decedido usar tecnología NoSql en contraposición a la tecnología relacional, más común en la actualidad.

4.5.1 SQL vs NoSQL

SQL del inglés *Structured Query Language*, es un lenguaje declarativo usado en las bases de datos relacionales, las más usadas hoy en dia, como por ejemplo MySQL [26], PostgreSql [27], SQLite [28]... Este tipo de bases de datos mantiene una estructura rígida con sus esquemas, restricciones, relaciones, tipos de datos etc.

En contraposición tenemos bases de datos NoSQL, que son todas aquellas que no siguen el modela relacional, consecuentemente tienen estructuras menos rígidas. Así pues existen muchos tipos de bases de datos NoSQl como las orientadas a documentos (estructuras de datos en documentos por ejemplo tipo JSON con un esquema dinámico) como MongoDB [29], las del tipo clave-valor como Cassandra [30] o las estructuradas en forma de grafo como Neo4j [31].

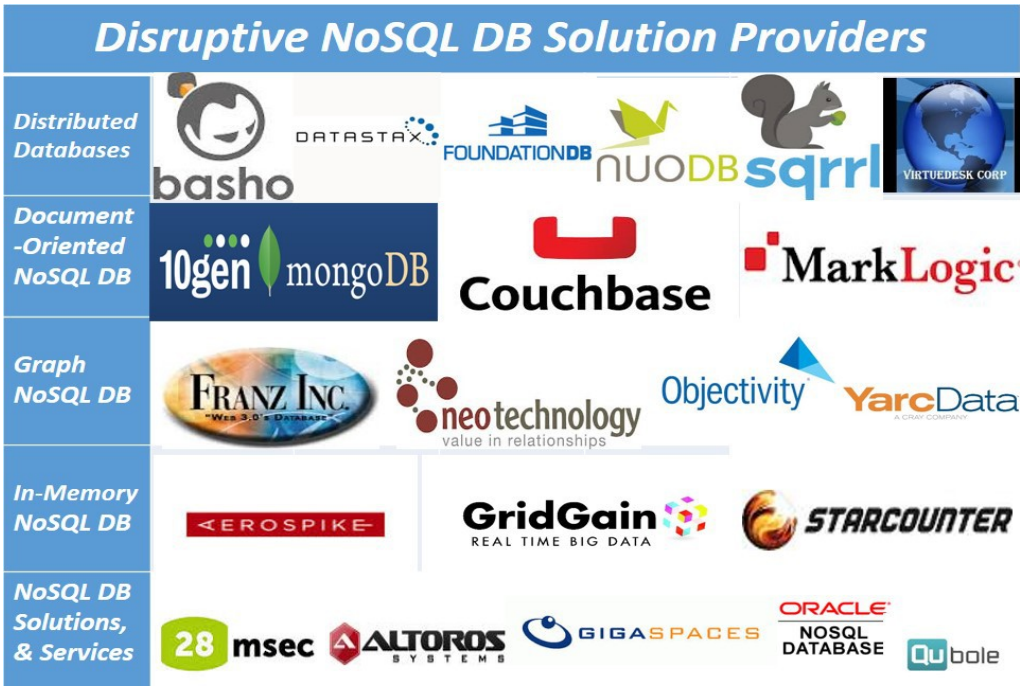


Figura 12. Clasificación de algunas de las bases de datos NoSQL

En la actualidad grandes compañiass que manejas cantidades enormes de datos como Google [32], Amazon [33], Twitter [9] o Facebook [30 han apostado por crear sus propias bases de datos NoSql.

Que ofrece NoSQL que no ofrezca SQL?

1. En primer lugar, las BD NoSQL ofrecen estructuras no rígidas, lo que las convierte en bases de datos mucho más escalables que las relacionales, en contextos de el Big Data la diferencia es substancial, mientras que escenarios más corrientes la diferencia puede pasar desapercibida Así por lo tanto en grandes volúmenes de datos, la escritura y lectura es mucho más rápida en las NoSQL [35].

   Además las BD NoSQL no siempre soportan transacciones ACID(***A****tomicity,* ***C****onsistency,* ***I****solation and* ***D****urability*) [36], lo que mejora aun más la escalabilidad, esto puede ser un punto negativo en contextos de negocios, banca... pero puede ser un detalle de poca importancia en casos como redes sociales, o el escenario del proyecto, donde si alguna operación no se ha relizado o no se ha acabado no tiene una gran importancia.

2. Las BD NoSQL ofrecen modelos heterogeneos de datos, así pues en caso de tener una ontología invariable y clara desde un inicio es interesante trabajar con BD relaciones mientras que si los campos de los datos pueden ir variando en el tiempo, o se modifican las estructuras constantemente o simplemente no tienen una estructura homogénia las bases de datos NoSQL cobran más sentido.

Entonces, por que NoSQL?

1. Este proyecto apuesta por el largo plazo, esto quiere decir que se espera que las Bases de datos crezcan mucho, pues hay mucha información sobre el mundo de la cocina en la red. En este escenario del largo plazo pues, se premia la escalabilidad, donde la NoSQL ganan por mucho.

2. Otra razón consiste en los modelos de datos, queremos trabajar con un modelo de datos claro, pero que a la vez sea flexible y sin sacrificar que pueda existir heterogeneidad en los datos, pues en el mismo contexto de largo plazo, el proyecto puede crecer en unas direcciones u otras, lo que afectara al modelo de datos.

3. Por otro lado, el hecho de que no todas las bases de datos NoSQL garanticen transacciones ACID en este escenario no es un inconveniente, pues con la gran cantidad de datos si falta algún ingrediente en la transacción no es un problema grave.
4. Y por último la forma de grafo de algunas bases de datos como Neo4j resulta muy útil en este contexto, ya que esta es la forma más natural almacenar los ingredientes. Esta estructura de grafo facilitarà la manipulación del grafo al igual que el análisis y operaciones sobre el grafo.

Como se explica en los dos siguientes apartados, las bases de datos elegidas han sido MongoDB para los datos no procesados y Neo4j para los ya procesados.

Mongo para los no procesador, ya que como estos no amntienen relaciones entre ellos, una base de datos basada en documentos resula idonea.

Mientras que Neo4j ha sido la base de datos elegida para los datos procesados, ya que queremos moderlarlos como grafo, oportunida que nos da Neo4j.

4.5 Web

En un contexto como el actual, donde los servicios en red son cada día más populares es normal que la tecnología vaya avanzando en la direcciones de las aplicaciones distribuidas, las cuales a poco a poco van desbancando las aplicaciones locales. A poco a poco vamos viendo aplicaciones distribuida que pueden llegar a susbtituir de emólogo ofline, como el caso del editor de UML creately[37] que es más sencillo, intuitivo y manegable que sus equivalentes locales.

Para el dessarrollo web, se han utilizado las tecnologías Html, Css, Js, Jquery, Ajax y las propias herramientas de Django, el framework de python.

Django ha resultado muy util de ulitizar puesto que facilita el control de url, maneja bien los ajustes internos en un solo archivo y por encima de todo, permite la introduccion de código python en los templatess, así como los jsp en java.

Se ha decidido utilizar Django por encima de otros sistemas, puesto que la filosofia python encaja muy bien con este proyecto.

4.4 Visualización

La visualización es uno de los cinco objetivos del proyecto, es además una herramienta muy importante, no solo por las facilidades de análisis y comprensión que ofrece, sino porque debido a su naturaleza resulta muy visual y atractiva de cara al usuario, a diferencia de otros módulos como la limpieza de datos o las bases de datos NoSQL.

La primera aproximación consistio en utilizar d3.js, pero despues de ver que era poco escalable en grafos muy grandes, además de algo compleja de utilizar decidi utilizar sigma.js, librería js de visualización de grafos, que pese a no estan muy bien documentada ni tener demasiados ejemplos, es muy práctica y tiene plugins de compatibilidad con Gephi, el programa de visualizaciones.

5.0 Resultados y testeo

Como resultado, el proyecto ha logrado cumplir sus cinco objetivos, pese a que algunos requisitos no han sido cumplidos como la petición de credenciales en la página oculta del administrado (www.foodnet.no-ip.com/admin) que no ha dado tiempo a realizarla:

Por otro lado, las bases de datos de recetas de Neo4j, en el momento de la entrega contiene unas 900 recetas, que no son demasiadas, pero se puede seguir llenando con facilidad por el crawler.

Es recomendable para el testeo de las bases de datos usar Robomongo, un cliente de MongoDB que se puede conectar de forma remota a traves de ssh: usuario: Marc password: Andromina.

Especialmente se ha testeado el crawler que funcione correctamente, se han probado múltiples webs y la gran amyoria, unas 8 de cada 10, eran parseadas correctamente.

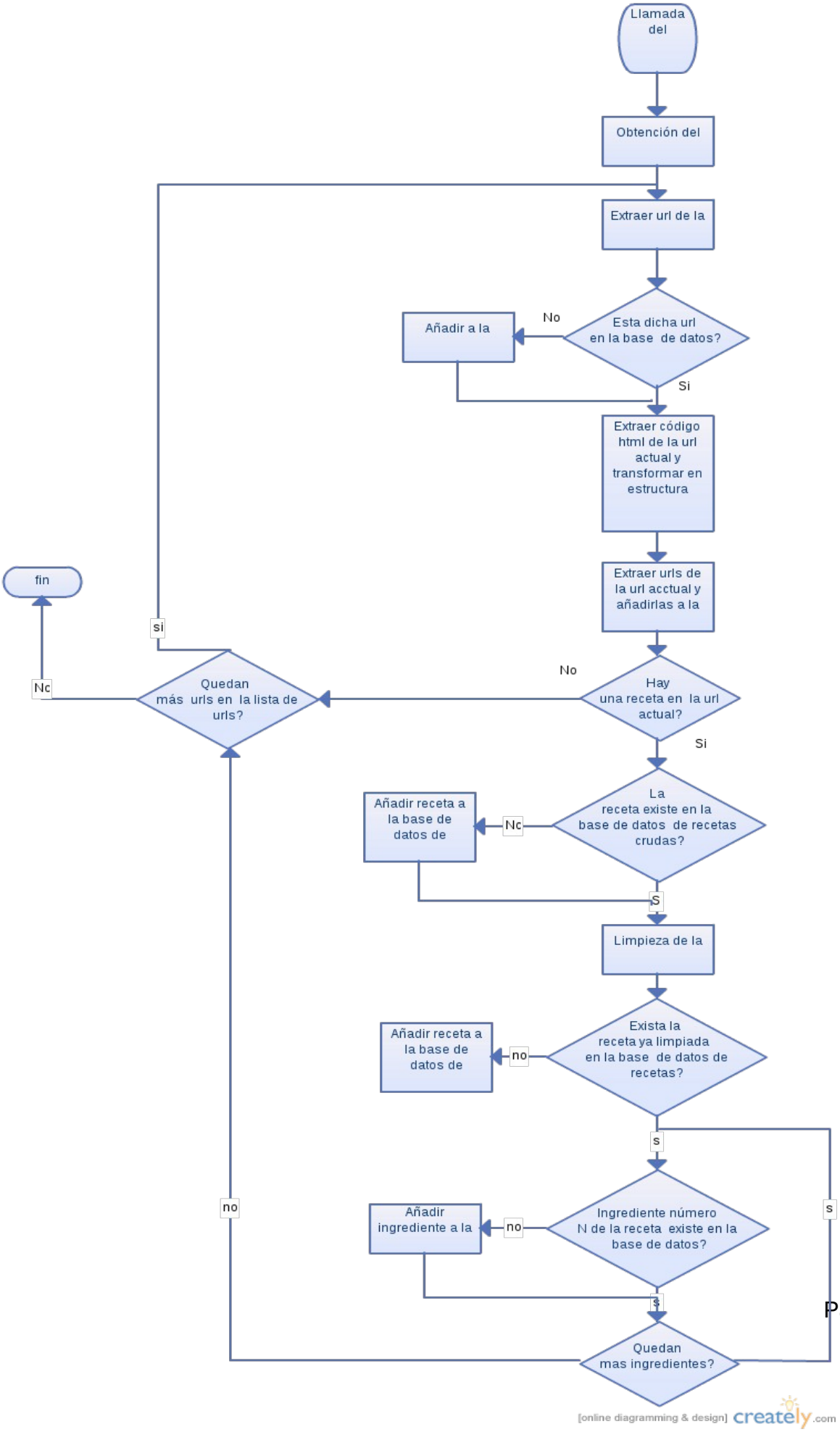
6.0 Posibles ampliaciones y largo plazo

Como se ha comentado constantemente este proyecto pretende ser una base para que pueda puedaser desarrollado posteriormente por otros.

Asi pues los diversos puntos a dessarrollar pueden ser:

- Mejorar la liempieza de datos, para poder extraer valores nutritivos o cantidades.
- Mejorar la viualización con extracciones de caminos mínimos o clustererizar mejor la visualización.
- Ampliar las bases de datos para poner a prueba la escalabilidad del sistema.

Anexos


Llamada del
Obtención del
Extraer url de la
Esta dicha url en la base de datos?
No
Añadir a la
Si
Extraer código html de la url actual y transformar en estructura
Extraer urls de la url acctual y añadirlas a la
Hay una receta en la url actual?
No
Si
Quedan más urls en la lista de urls?
Nc
si
fin
La receta existe en la base de datos de recetas crudas?
Nc
Añadir receta a la base de datos de
S
Limpieza de la
Exista la receta ya limpiada en la base de datos de recetas?
no
Añadir receta a la base de datos de
s
Ingrediente número N de la receta existe en la base de datos?
no
Añadir ingrediente a la
s
Quedan mas ingredientes?
s
no
[online diagramming & design] creately.com

4.0 Referencias

1. Portal web Bullipedia

   http://www.bullipedia.com/

2. Concurso Hackingbullipedia

   http://hackingbullipedia.org/bullipedia3-2

3. Vídeo propuesta ganadora Hackingbullipedia

   http://vimeo.com/77458528

4.

5. Martin Hilbert and Priscila López (2011)

   *The World's Technological Capacity to Store, Communicate, and Compute Information, 1 April 2011:**332**(6025),60-65.Published online10 February 2011[DOI:10.1126/science.1200970]*

6.

7. Librería de visualizaciones d3.js

   http://d3js.org/

8. Explicación del interes de las estructuras de forma de grafo en la actualidad

   https://www.youtube.com/watch?v=2ElGO1P8v0c

9. Twitter NoSQL graph db

   https://github.com/twitter/flockdb

10. Page rank paper

    http://en.wikipedia.org/wiki/PageRank

11. Software de visualización Gephi

    https://gephi.org/

12. GitHub

    https://github.com/

13. Kwang-Il Goh, Michael E. Cusick, David Valle, Barton Childs, Marc Vidal, and Albert László Barabási (2007)

    The human disease network 2007 104 (21) 8685-8690; published ahead of print May 14, 2007,doi:10.1073/pnas.0701361104

14. The human disease network data set

    http://gephi.org/datasets/diseasome.gexf.zip

15. The human disease network, sigma.js visualization

    http://exploringdata.github.io/vis/human-disease-network/

16. Diseasome project

    http://diseasome.eu

17. Lxml

http://lxml.de/

18. Yong-Yeol Ahn, Sebastian E. Ahnert, James P. Bagrow, Albert-László Barabási (2011)

Flavor network and the principles of food pairing, Dec 15, 2011, Scientific Reports Nature Publishing Group

19. Base de datos BEDCA

http://www.bedca.net/bdpub/index.php

20. Scrum

http://en.wikipedia.org/wiki/Scrum_%28software_development%29

21. Python
https://www.python.org/

22. XPath

http://www.w3schools.com/XPath/

23. Scrapy

http://scrapy.org/

24. BeautifulSoup

http://www.crummy.com/software/BeautifulSoup/

25.

26. MySQL

http://www.mysql.com/

27. PostgreSQL

http://www.postgresql.org/

28. SQLite

http://www.sqlite.org/

29. MongoDB

http://www.mongodb.com

30. Cassndra apache

http://cassandra.apache.org/

31. Neo4j

http://www.neo4j.org

32. Google Cloud DataStore

https://developers.google.com/datastore/

33. Amazon NoSQL

http://aws.amazon.com/nosql/

34.

35. Lxml Benchmarks
http://lxml.de/performance.html
36. ACID
http://es.wikipedia.org/wiki/ACID
37. Creately
https://creately.com/