Explicaciones para un recomendador de música basadas en datos enlazados
Explanations based on Linked Data for a Music Recommender System

Trabajo de Fin de Grado
Curso 2019–2020

Autor
Adrián Garrido Sierra
Diego Sánchez Muniesa

Director
Guillermo Jiménez Díaz
Marta Caro Martínez

Grado en Ingeniería Informática
Facultad de Informática
Universidad Complutense de Madrid

Explicaciones para un recomendador de música basadas en datos enlazados
Explanations based on Linked Data for a Music Recommender System

Trabajo de Fin de Grado en Ingeniería Informática
Departamento de Ingeniería del Software e Inteligencia Artificial

Autor
Adrián Garrido Sierra
Diego Sánchez Muniesa

Director
Guillermo Jiménez Díaz
Marta Caro Martínez

Convocatoria: *Septiembre 2020*
Calificación:

Grado en Ingeniería Informática
Facultad de Informática
Universidad Complutense de Madrid

DIA de MES de AÑO

Resumen

Explicaciones para un recomendador de música basadas en datos enlazados

Un resumen en castellano de media página, incluyendo el título en castellano. A continuación, se escribirá una lista de no más de 10 palabras clave.

Palabras clave

Máximo 10 palabras clave separadas por comas

Abstract

Explanations based on Linked Data for a Music Recommender System

An abstract in English, half a page long, including the title in English. Below, a list with no more than 10 keywords.

Keywords

10 keywords max., separated by commas.

Índice

Índice de figuras

Capítulo 1

Introducción

Desde el desarrollo de Internet como una red global, siempre ha albergado grandes cantidades de documentos e información escrita. A medida que fue creciendo, hubo que encontrar distintas maneras de organizar esa información para que pudiera ser utilizada como un servicio a cualquier usuario. La Web Semántica, cuyas características explicaremos más tarde, es uno de los métodos que toda la red global de Internet pudo adaptar a la hora de darle un sentido y una utilidad a esa información.

Básicamente, la Web Semántica trata de enlazar toda esa información con tal de darle contexto y accesibilidad. Para ello adopta el método de Linked Data, el cual haciendo referencia a su propio nombre, conecta o une distintos datos estableciendo una relación. El principal objetivo de todo esto es que los ordenadores puedan navegar y acceder a la mayor parte de los datos almacenados en Internet para que estos sean útiles y no queden perdidos en un mar de información.

Hay varias maneras de poder organizar y conectar todo ese contenido conocidas como sistemas ontológicos. La ontología es una rama filosófica que estudia los entes y sus conexiones y aquí es aplicada en su punto más técnico. El método ontológico que nosotros usaremos es el modelo RDF, uno de los más usados y extendidos.

Es mediante este concepto como los motores de búsqueda pueden ser ayudados por esta tecnología, ya que dotamos a las máquinas de la capacidad para navegar sobre esas relaciones y entender datos que nosotros requerimos o que podrían sernos útiles.

1.1. Motivación

La razón de ser de este proyecto consiste en dar explicaciones a un recomendador de música. Partimos de un recomendador de música que proporciona dos canciones, las cuales tienen algún tipo de relación entre ellas. Utilizando esa premisa como base, queremos explicar **por qué** esas dos canciones están relacionadas o, dicho de otra forma, en qué se parecen. Para alcanzar esa meta haremos uso de la web semántica.

Como ya hemos comentado, la Web Semántica es una manera de mejorar la conexión entre los datos guardados en la red, haciendo que Internet sea más útil y accesible para las

personas. Se trata de un método que permite que navegar por la red sea más intuitivo y reduce la cantidad de información que acaba volviéndose inaccesible para aquellos usuarios para los que resulta relevante.

El problema es que es una práctica que aún no está lo bastante extendida para que suponga un verdadero avance en nuestra forma de relacionarnos con la red, pues requiere hacer cambios en la forma en que se almacena la información. En este trabajo intentaremos mostrar el potencial de la Web Semántica.

La idea principal es poder recoger todas las conexiones que aparezcan entre toda la información que podamos recoger sobre cada uno de los dos temas musicales o canciones. Para ello, deberemos estudiar ese ente y todos los relacionados: artista que interpreta la canción, su género musical, los integrantes de su grupo, su sello discográfico, etc. y a su vez las relaciones entre estos, hasta disponer de suficientes conexiones que nos puedan asegurar que esas dos canciones tienen o no algo que ver.

1.2. Objetivos

Los objetivos principales de este trabajo son los siguientes:

- Hacer un estudio sobre la Web Semántica y los modelos de datos que la permiten, desarrollando más tarde una aplicación que ejemplifique las aplicaciones prácticas que puede tener esta Web Semántica.

- Establecer una relación entre dos canciones que, a priori, una persona no podría obtener sin hacer un estudio amplio de ello, empleando para ello explicaciones propias que compartan esas dos canciones. Y, sobre todo, intentar relacionar canciones que no sean tan parecidas musicalmente pero compartan otros elementos que puedan causar más curiosidad.

- Determinar qué propiedades pueden llegar a ser más útiles para establecer explicaciones para la relación entre dos canciones proporcionadas. Esta es la premisa de nuestro trabajo y, gracias a nuestro estudio, podremos determinar qué propiedades son más importantes a la hora de obtener explicaciones significativas en el dominio.

- Estudiar diferentes tecnologías que nos ayuden en la elaboración de la aplicación mencionada, además de ampliar nuestros conocimientos en áreas de la informática que no hemos explorado necesariamente durante el grado.

- Crear distintos diseños para la interfaz de nuestra aplicación hasta llegar al diseño final, utilizando un desarrollo iterativo que nos permita obtener un resultado satisfactorio.

- Una vez terminado, extraer conclusiones sobre el trabajo realizado tanto a nivel del estudio de las canciones como sobre el proceso de desarrollo y la utilidad de la Web Semántica.

1.3. Plan de trabajo

El punto de partida para el proyecto es estudiar todas las nuevas tecnologías y herramientas que vamos a necesitar, al menos, para el proceso de obtención y estudio de los datos.

Partimos de un dataset de canciones determinado, así que antes de proseguir debemos hacer una pequeña limpieza del mismo ya que contiene valores sucios y nulos. También decidimos hacer unas pequeñas agrupaciones por popularidad y por género. La popularidad se debe a que las canciones más populares son las que tienen más probabilidades de estar mejor documentadas en Wikidata y el género es porque queremos saber qué tipo de canciones predominan en nuestra muestra.

Aparte del estudio del dataset y como parte de la exploración de tecnologías, ejecutaremos pequeñas consultas con formato RDF en la herramienta propia de Wikidata. Gracias a esta experimentación podremos hacernos una idea de en qué se diferencian las consultas que más tarde usaremos en nuestra librería de SPARQL, además de la cantidad de información que seremos capaces de obtener con cada una de esas consultas.

El siguiente paso tras acumular cierta experiencia con Wikidata, SPARQL y el formato RDF es buscar una librería lo suficientemente completa y potente que pueda servirnos para ejecutar las consultas deseadas en un tiempo aceptable. Esto también se verá afectado por el lenguaje de programación que decidamos utilizar para el código de nuestra aplicación.

Una vez tomadas las decisiones previas, será el momento de comenzar el desarrollo del código de nuestra aplicación. Esto abarca la estructura de clases, el desarrollo de las librerías necesarias para la obtención de datos y la creación de algoritmos que sirvan para establecer relaciones entre canciones (que llamamos explicaciones) y representarlas gráficamente para comprensión del usuario. Este será un proceso iterativo en el que vayamos agregando las funcionalidades necesarias además de ir depurando todos los errores que vayan apareciendo hasta llegar a una versión final.

Por último, haremos una reflexión sobre el trabajo realizado teniendo en cuenta lo aprendido a lo largo del proceso, las dificultades que puedan surgir durante su desarrollo y aquellas cosas que pueden mejorarse o incluir en una hipotética versión actualizada del proyecto.

De forma simultánea a todo lo anteriormente explicado, elaboraremos gradualmente la memoria recogida en este documento. Esta memoria será compuesta en LaTeX para facilitar el trabajo de edición y beneficiarnos de sus funciones, y en ella se explicará en detalle todo el proceso de este TFG.

La memoria está dividida en varios capítulos que exploran distintas facetas del proyecto:

- El primero es un capítulo de introducción. Aquí es donde nos encontramos ahora y donde se recogen las previsiones y objetivos a cumplir en el trabajo.

- El segundo capítulo, "Estado de la Cuestión", representa el estudio realizado sobre el ámbito de nuestro trabajo. Aquí se recogerá la información relacionada con conceptos como la Web Semántica, Linked Data o el formato RDF, entre otros.

- El tercer capítulo será una recopilación de todas las tecnologías utilizadas a lo largo del proyecto con sus principales características y ventajas. De la misma forma, aquí se recogerán tecnologías o herramientas que estudiemos pero decidamos descartar o sustituir por algún motivo.

- El capítulo cuatro será uno de los más importantes del documento. En él definiremos qué entendemos como una "explicación" en el ámbito de nuestro trabajo, además de enumerar todas las explicaciones diferentes que usaremos en la versión final de la aplicación. Se proporcionarán todas las descripciones y ejemplos necesarios para que se entiendan correctamente estos conceptos.

- A continuación seguiremos hablando de la aplicación con un capítulo dedicado al diseño de la interfaz. En él documentaremos el proceso de desarrollo para la interfaz, examinando las iteraciones más importantes y explicando las decisiones que se hayan tomado en cada caso. Finalmente se dedicará un apartado a examinar la versión final, que será con la que se podrá interactuar en la aplicación terminada.

- El sexto capítulo será dedicado a la implementación de la aplicación. Servirá para exponer la arquitectura elegida para organizar el código y otros aspectos como la parte técnica del estudio del dataset o la extensibilidad de la aplicación.

- El último capítulo está reservado para hablar de conclusiones y trabajo futuro. Después de terminar el desarrollo de la aplicación, reflexionaremos sobre el mismo y también sobre el ámbito tratado. Hablaremos sobre contratiempos que hayamos podido encontrar a lo largo del camino y también de aquellas funciones o mejoras que nos gustaría incorporar al proyecto si tuviéramos la oportunidad de seguir trabajando en él en un futuro.

Capítulo 2

Estado de la Cuestión

Actualmente Internet aloja una cantidad ingente de información de todo tipo. A pesar de los beneficios evidentes que supone tener tanta información subida a la web, este hecho también trae consigo problemas debido a dificultades semánticas, un más que elevado número de fuentes de información y el propio tamaño de los datos. Como consecuencia de esto, para poder hacer consultas a información y navegar de un contenido a otro que esté conectado es necesario tener un sistema de organización para toda esa cantidad de datos.

A continuación exponemos tecnologías cuyo objetivo es acceder a datos con un contexto y a facilitar el proceso de búsqueda al proveer información verdaderamente relevante para los usuarios.

2.1. Conceptos sobre el ámbito

La llamada **Web Semántica** es una extensión de la World Wide Web propuesta por Tim Berners-Lee [2], cuyo objetivo es proveer a los datos estructura y significado. Con esto se consigue que la información de la web sea más fácilmente comprensible para las máquinas (o agentes de software). De esta forma se desea lograr que los agentes no solo analicen gramaticalmente, sino que comprendan la gran cantidad de información contenida en la web. Gracias a este entendimiento, los agentes podrán integrar datos de diversas fuentes, inferir hechos ocultos y responder a consultas complejas fácilmente. Finalmente lo que se ha creado es un sistema que permite navegar por la información mediante enlaces o hipervínculos, navegando así entre datos conectados [8].

Para poder hacer realidad esta Web Semántica es necesario que la información de la web sea accesible en un formato estandarizado, accesible y manejable por las herramientas de la Web Semántica. Además, no basta con tener acceso a los datos sino también a la relación entre la información, lo cual marca la diferencia entre la Web Semántica y una simple colección de datasets. Esta colección de datos interrelacionados es lo que llamamos **Linked Data** (o datos enlazados) y será la base de nuestro trabajo en el dominio de los temas musicales.

Hay dos tecnologías que sirven como modelo a la hora de configurar y organizar toda esta información. La primera es el modelo XML “Standard Generalised Mark-up Langua-

ge". Este sistema se basa en la utilización de etiquetas para la descripción y organización de los datos. Un documento XML es capaz de marcar y dotar de significado diferentes partes de un texto que se necesiten nivelar o etiquetar. Se utilizar tanto para representar información en una página HTML hasta organizar registros de una base de datos.

El segundo sistema estructural y actualmente el que nos interesa para este proyecto es el **Esquema RDF** (Resource Description Framework), que es un modelo de datos basado en declaraciones sobre recursos web mediante expresiones sujeto-predicado-objeto. Dichas expresiones se llaman "tripletas", donde el sujeto representa al recurso, el objeto representa el valor del recurso y el predicado supone los rasgos o aspectos del recurso que relacionan sujeto y objeto [8]. Un ejemplo a grandes rasgos podría ser el siguiente:

:Juan es :Persona
:Juan hijoDe :María

Con este simple ejemplo ya hemos representado el hecho de que existe algo llamado Juan, que es una persona y además es hijo de María. Aquí hemos juntado dos conjuntos de tuplas o tripletas relacionadas entre ellas de una forma que podemos representar mediante un grafo.

Es necesario señalar que este es un ejemplo muy básico y sencillo. En la web real, existen millones de tripletas relacionadas entre sí por lo que en nuestro ejemplo tanto juan como María tendrían un número muy alto de tripletas más que representan toda la información que se tiene sobre ellos en la web.

A la hora de llevar esta explicación a un nivel más técnico y realista, estas expresiones no se formulan con sus nombres en lenguaje natural que hemos usado para el ejemplo. Estos tres argumentos que conforman la unidad de tripleta, se representan mediante un Identificador de Recursos: una URI (Uniform Resource Identifier) [8, 1].

Una URI (Uniform Resource Identifier) es una cadena de caracteres que puede identificar una entidad o recurso por su nombre o bien por su ubicación, la cual muestra dónde podemos acceder a él. Siguiendo esta división, una URI puede comprender el URL (Uniform Resource Locator) y el URN (Uniform Resource Name). La URN identifica la entidad por su nombre, lo cual no nos proporciona su ubicación y no asegura que esta entidad esté disponible. Sin embargo, la URL nos proporciona la localización directa de la entidad y esta es la principal razón de que sea mucho más usada en la web [3, 7, 8].

Volviendo a lo que respecta al modelo RDF, hemos determinado el hecho de que si empleamos este lenguaje seremos capaces de relacionar un sujeto y un objeto mediante un predicado, de forma que obtendremos un grafo similar al de la Figura 2.1.

Para una persona normal, esta puede resultar una nomenclatura poco amigable ya que las URIs pueden llegar a ser mucho más engorrosas y complejas. Más adelante explicaremos cómo hemos resuelto este problema en nuestro proyecto para que cualquiera que haga uso de nuestra interfaz pueda reconocer todos los objetos y, sobre todo, para darnos mucha más comodidad a la hora de trabajar con ellos.

Podríamos decir que estamos estableciendo una relación entre dos objetos unidos me-

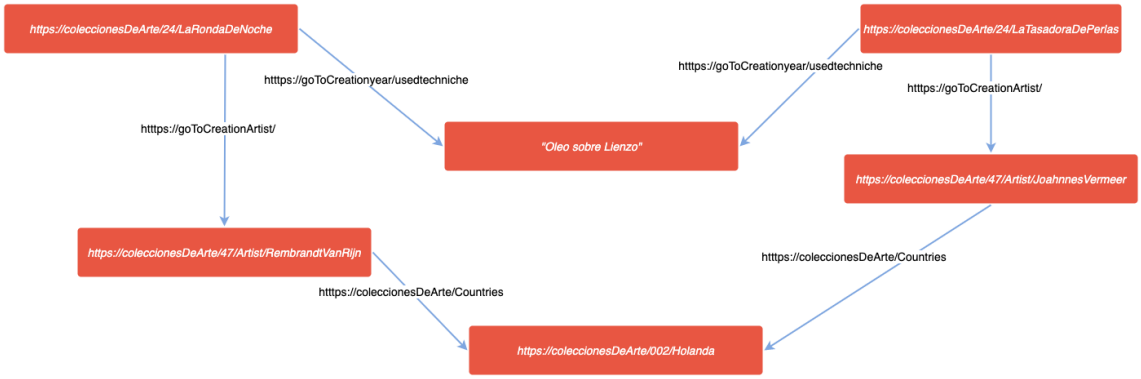

Figura 2.1: Ejemplo de grafo siguiendo el modelo RDF

diante una propiedad. Esta propiedad es una "explicación" que nos sirve para relacionar objetos. En nuestro proyecto, estos objetos son canciones.

2.2. SPARQL

Finalmente, para poder empezar a trabajar con la información hemos usado el lenguaje SPARQL, el cual nos permite consultar los datos enlazados de la web. Este es un lenguaje especializado para buscar y consultar datos RDF. Básicamente facilita el acceso a las URIs para después poder aplicar todo tipo de selección de datos, restricciones, condiciones y límites a las entidades con los esquemas explicados anteriormente. La estructura de una consulta SPARQL suele ser la siguiente:

```
PREFIX EX: Uri de una web
SELECT * variables que se desean conseguir
WHERE
{
    Operaciones con tripletas
}
OFFSET X intervalo de desplazamiento
LIMIT X limite de los datos obtenidos
```

La cadena PREFIX hace referencia a la URI o URIs donde se encuentran localizados las entidades a las que hacen referencia las tripletas. Su objetivo es no tener que escribir la URI entera, la cual es una cadena de texto poco amigable para un usuario, cada vez que lo referenciáramos en una relación de las tripleta dentro de la cláusula WHERE.

A la hora de hacer una consulta compleja, es posible que lleguemos a un código que referencie muchas URIs y entidades de forma que no sea un lenguaje de uso fácil. Es por esto por lo que pudimos encontrar variedad mediante la librería que nos dispone Wikidata que simplifica algo más la nomenclatura y los caracteres de este lenguaje.

2.3. WikiData

Wikidata es una base de datos libre, colaborativa, multilingüe y secundaria, que recopila datos estructurados para dar soporte a Wikipedia, Wikimedia Commons, así como a otras wikis del movimiento Wikimedia y a cualquier persona en el mundo [4]. El proyecto comenzó en 2012 liderado por Wikimedia Alemania. Gracias a Wikidata, tenemos una gran cantidad de información relevante acerca de toda nuestra cultura reunida en un mismo sitio. La usaremos para consultar e investigar la información relacionada con nuestro dataset (o conjunto de datos), esencialmente datos acerca de artistas y sus temas musicales. Tiene numerosos servicios, uno de los que probablemente hayamos usado y no nos hemos dado cuenta, es el de servir información para rellenar las fichas de los artículos de Wikipedia.

Wikidata está estructurada de tal manera que resulta sencillo acceder a su información utilizando el lenguaje de consultas SPARQL. Cada objeto tiene ciertas propiedades a las que podemos acceder seleccionando el identificador correcto de la propiedad.

Una de las ventajas de Wikidata es que nos va a eliminar toda la nomenclatura de URIs propias de SPARQL. En estas consultas, las entidades serán referenciadas con los caracteres wd y las propiedades con los caracteres wdt. Un ejemplo de una consulta sencilla podría ser:

-Obtén todos los Artistas cuyo género musical sea Rock:

```
SELECT ?singer ?singerLabel ?genre ?genreLabel
WHERE
{
    ?singer wdt:P31 wd:Q215380;
    wdt:P136 wd:Q7749;
    SERVICE wikibase:label { bd:serviceParam wikibase:language &quot;en&quot; . }
}
```

En esta consulta estamos accediendo a todos los objetos que contengan las propiedades (wdt) P31 (Tipo Instancia) y P136 (Género) con el valor (wd) que nosotros estamos seleccionando, Q215380 y Q7749, forzando así a que las dos propiedades sean Grupo Musical y Rock and Roll respectivamente.

SPARQL es un lenguaje muy potente que puede abarcar gran cantidad de datos en función de la web (se pueden crear consultas mucho más complejas para obtener datos más específicos).

Capítulo 3

Explicaciones

El objetivo de nuestro trabajo consiste en explicar las relaciones entre dos canciones dadas por un recomendador de música que trabaja con un dataset concreto. De esta manera, llamaremos **explicación** a cada factor común que compartan ambas canciones, siendo principalmente propiedades que poseen las propias canciones o los valores de otras propiedades intermedias. A lo largo de este capítulo concretaremos todas las explicaciones que vamos a estudiar y el sistema que une todas ellas para poder relacionar dos canciones en su totalidad.

Las relaciones que recogemos en cualquier caso de uso forman un grafo siguiendo el modelo RDF. Hemos decidido, además de seguir ese formato, dotarlas de un nivel que escala a medida que el estudio se aleje del objeto canción inicial. Al expresar alejamiento, nos referimos a que, como en el contexto del formato RDF y Linked Data, la creación de relaciones se establece navegando por distintos objetos relacionados con el inicial secuencialmente. La idea de los niveles viene dada por dos motivos:

El primero es debido a la cercanía de niveles. Cuanto menor sea el nivel, significará que está mas próximo al objeto canción inicial y por ello tendrá más peso a la hora de relacionarlas.

El segundo es porque a la hora de representar gráficamente todas las relaciones, queremos dividirlas por objetos de estudio de una forma nivelada y ordenada.

En una primera fase, generaremos explicaciones básicas para relacionar directamente las dos canciones u objetos principales, por ejemplo: género, artista, álbum, etc. Asignaremos una complejidad de k=1 a las explicaciones directas. Estas explicaciones son una relación directa entre dos canciones relacionadas por una propiedad, es decir, solo tenemos en cuenta propiedades ligadas a cada objeto en un primer nivel.

Una vez obtenemos las relaciones básicas, es necesario seguir avanzando en el estudio, ya que dos canciones que a priori no son muy similares musicalmente, no compartirán casi ninguna de estas relaciones directas. Es por esto por lo que indagamos más en la información que podemos obtener a partir de las relaciones que hemos sacado en el paso anterior. Aquí aparecen las relaciones indirectas, aquellas que surgen del estudio de otras obtenidas en niveles anteriores, siempre que coincidan en ambas canciones. La idea es poder represen-

tar toda la información compartida, de forma que se pueda representar con un grafo. Estas explicaciones pueden tener un nivel de complejidad diferente (k= 2, 3, 4...) dependiendo de cuántos niveles se hayan estudiado. La idea principal es ejecutar un estudio de ambas canciones obteniendo varios niveles de cada una de ellas, para finalmente unirlas enlazando únicamente las que coincidan.

Hemos recopilado a lo largo de investigación y consultas un conjunto de posibles relaciones a obtener, que suman un total de 24 explicaciones. Es importante añadir que la mayoría de ellas pueden presentarse en distintos niveles. Para explicar este caso más fácilmente lo representamos con el ejemplo de la Figura 3.1.

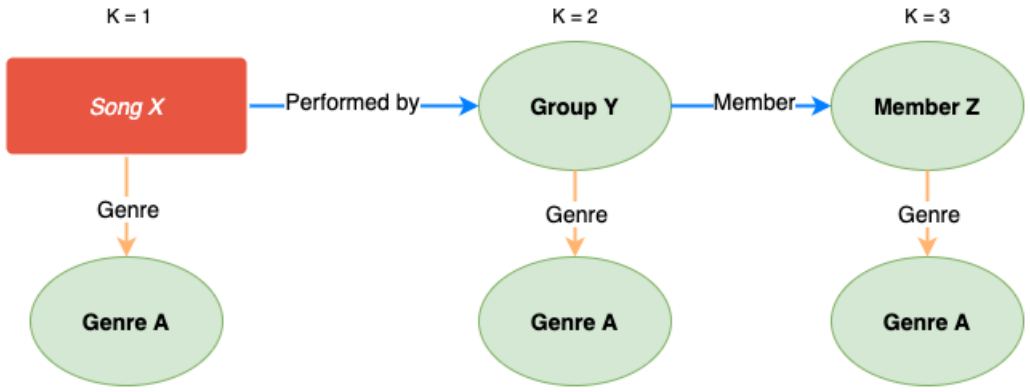

Figura 3.1: Ejemplo de misma relación en distintos niveles

Para nuestro caso, en el cual tratamos con dos objetos iniciales que representan a dos canciones, hemos seleccionado una serie de propiedades a investigar para obtener los distintos niveles. El primer nivel es el más intuitivo: obtener relaciones directamente de ambas canciones. Entre esas relaciones se encontraría la relación "Interpretado por", la cual nos proporciona el artista o grupo que interpreta la canción. Este artista es el que vamos a usar como objeto de estudio del nivel k = 2, obteniendo así otro conjunto de relaciones del artista y sus respectivos valores. También obtendremos el género de la canción, que usaremos para generar el nivel k = 3 mediante las propiedades que obtengamos del género. Por último, como nivel final hemos seleccionado investigar las propiedades de los miembros que conforman la banda de música en caso de que el intérprete de la canción esté formado por varios artistas, y denominamos a este último nivel k= 4.

Queremos señalar que estos niveles podrían alargarse más a medida que seguimos investigando nuevas propiedades y generando un grafo más completo. Hemos decidido poner un límite en este punto debido a que, a medida que vamos avanzando en el estudio de los niveles de las propiedades, las explicaciones son menos significativas debido a que están más alejadas del primer nivel, que es el que está relacionado con la canción en sí.

En esta sección vamos a recopilar las principales propiedades de la lista total y cómo relacionarían dos canciones objeto iniciales. Algunas de ellas podrán aparecer en distintos niveles mientras que otras serán propias de un solo nivel:

3.1. Explicaciones de la canción

Aquí se encuentran las explicaciones basadas en propiedades del objeto canción que no pueden aparecer en el resto de niveles:

Artista

El artista es una de las explicaciones más obvias pero también es una de las más importantes. Esta explicación hace referencia a cuando dos temas son interpretados por el mismo artista, por lo que tienen una clara relación directa ya que suele existir similitud entre las canciones de un artista. Además, a partir del artista podemos obtener otras relaciones más complejas en función de los datos de este mismo.

Como ejemplo podemos tomar los temas *One More Time* y *Something About Us*. Ambos son interpretados por el dúo musical **Daft Punk**, así que podemos explicar su relación gracias a este dato.

Para esta explicación utilizamos la propiedad "performer" (intérprete) de Wikidata. Siguiendo el modelo RDF, el sujeto es la canción (*One More Time*, por ejemplo), el predicado es la propiedad intérprete y el objeto es el artista (**Daft Punk**).

Figura 3.2: Ejemplo de explicación por artista

Álbum

Una explicación muy potente será que ambas canciones pertenezcan al mismo álbum. A menudo esta explicación aparecerá acompañada de la explicación del artista y, en cualquier caso, la relación que existe entre dos temas del mismo álbum suele ser más estrecha debido a que poseen más puntos en común, como puede ser el género, la fecha o la temática.

Tomemos como ejemplo *Today* y *Disarm*. Estas dos canciones son muy cercanas porque tienen varios puntos en común, pero una de las explicaciones que podemos ofrecer es que ambas pertenecen al álbum *Siamese Dream*, de **The Smashing Pumpkins**. *1979* es otro tema que se podría recomendar a raíz de *Today*, pero es una peor elección que *Disarm* porque pertenece a un álbum diferente.

Volvemos a utilizar Wikidata para obtener esta explicación, concretamente la propiedad "part of" (parte de). Esta propiedad se utiliza para varias cosas, entre ellas indicar los álbumes a los que pertenecen las canciones. Así obtenemos que *Today* es "parte de" *Siamese Dream*.

Figura 3.3: Ejemplo de explicación por álbum

Fecha de publicación (Década)

Creemos que hay una mayor probabilidad de que exista una relación entre dos canciones que pertenezcan a la misma década. Esto se debe a que a lo largo del tiempo ha habido periodos marcados por uno o varios géneros musicales. Esto también ayuda a estudiar cómo estos distintos géneros están relacionados entre sí, lo cual es otro punto importante a tener en cuenta ya que hay géneros íntimamente relacionados entre sí: techno y house, heavy metal y thrash metal, etc.

Gracias a esta explicación, podemos relacionar dos canciones que a priori son muy distintas, como es el caso de *Womanizer* de **Britney Spears** e *Hysteria* de **Muse**. Estos temas no comparten algo tan básico como el artista o el género, pero ambos fueron publicados en los años 2000 y por ello pueden resultar interesantes para un usuario que busque escuchar los ritmos de esa época.

Para obtener esta explicación, emplearemos la propiedad "publication date" (fecha de publicación) de las canciones en Wikidata y comprobaremos si las dos fechas se sitúan dentro de la misma década.

Figura 3.4: Ejemplo de explicación por década

Singles

Existe también una cierta relación entre aquellos temas que sean singles (o sencillos), así que consideramos esto como una explicación más. El razonamiento para esta decisión es que los singles son canciones que se publican de forma independiente por razones promocionales, por lo que suelen convertirse en los temas más populares y representativos del trabajo del artista. Por ello, pueden poseer más valor para el recomendador que otras canciones porque los singles tienen más probabilidades de coincidir con los gustos del usuario.

El tema *Hung Up* de **Madonna** es un single, así que podemos establecer cierta relación con *Feel Good Inc.* de **Gorillaz**, que también es un single.

Para esta explicación emplearemos la propiedad "instance of" (instancia de) en Wikidata y su valor "single". De esta forma comprobaremos que tanto *Hung Up* como *Feel Good Inc.* son "instancia de" "single".

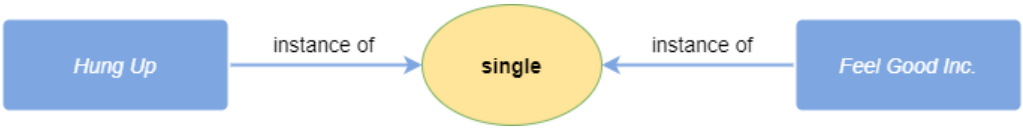

Figura 3.5: Ejemplo de explicación por single

3.2. Explicaciones compartidas por Artista y Canción

En esta sección exponemos propiedades que pueden ser establecidas tanto en el nivel de estudio de la canción como en el nivel de estudio del artista.

Premios

La siguiente explicación son los premios recibidos. Existe una variedad de premios compartidos por diferentes canciones. Algunos de estos premios son más específicos que otros, pero en cualquier caso pueden resultar una relación útil para nuestras explicaciones ya que son un reflejo de la repercusión de las canciones. El mismo caso se da para los artistas, aquellos que suelen optar al mismo premio tienen más posibilidades de tener un estilo muy similar y por ende se puede establecer una relación de cercanía.

En esta sección hemos recogido dos propiedades: "award received" (premio recibido) y "nominated for" (nominado a). Estas categorías hacen referencia a los premios que han recibido los artistas o canciones y a los premios a los que han sido nominados respectivamente. Normalmente los premios se reparten según distintas categorías en función al estilo de música, pero también hay otros que nos aportan otro tipo de información al estudio como Grammy al Mejor Videoclip, Grammy Award a la Mejor Canción Escrita, Grammy Award a la Mejor Interpretación Rap/Sung etc...

Basándonos en el nivel de canción tenemos como ejemplo *Single Ladies (Put a Ring on It)* de **Beyoncé** y *Rolling in the Deep* de **Adele**, ya que ambos temas han recibido el Premio Grammy a la canción del año. Además, como es lógico, ambas canciones fueron nominadas a ese mismo premio, así que podemos mostrar ambas propiedades en este ejemplo.

Usaremos la propiedad "award received" (premio recibido) y "nominated for" (nominado a) encontradas en Wikidata. De esta forma, *Rolling in the Deep* sería el sujeto, "award received" y "nominated for" serían los predicados y *Grammy Award for Song of the Year* sería el objeto.

Figura 3.6: Ejemplo de explicación por premio

Género

La explicación Género es una de las más representativas ya que las personas se guían por el género que prefieren para escuchar canciones similares, aunque en ocasiones no deba ser así. Un usuario que sea fan del jazz querrá escuchar canciones de ese mismo género o géneros similares porque todas los temas de un mismo género comparten una serie de puntos en común como la elección de los instrumentos, temáticas similares en sus letras o patrones de composición.

Gracias al género, podemos explicar la relación entre canciones como *A-Punk* de **Vampire Weekend** e *Brianstorm* de **Arctic Monkeys**, que pertenecen a un subgénero muy concreto del rock conocido como **surf music**.

Para esta relación utilizamos la propiedad "genre" (género) de Wikidata. Así tenemos que la canción *A-Punk* pertenece al género **surf music**.

Figura 3.7: Ejemplo de explicación por género entre canciones

En este caso podríamos crear una relación del mismo tipo con los artistas, debido a que ambos artistas de las dos canciones previas también cuentan con un registro de **alternative rock** (entre otros géneros) en su propiedad de género.

Figura 3.8: Ejemplo de explicación por género entre artistas

Idioma

La explicación Idioma denota, como su nombre sugiere, que ambas canciones están en el mismo idioma o lenguaje. Existen numerosos ejemplos de temas musicales que están escritos en un idioma que no se corresponde con la lengua materna del artista que los publican, o sencillamente están escritas en un idioma que se habla en varias partes del mundo distintas, como el inglés o el español. En ciertos casos el compartir idioma es un indicativo de otras relaciones adicionales, como puede ser el caso de canciones que pertenecen a un folclore regional determinado. En cualquier caso el compartir lenguaje ya es una relación directa en sí misma.

Como ejemplo podemos tomar el tema *Talk* de **Coldplay** y *Light My Fire* de **The Doors**, los cuales están escritos en inglés.

Para esta explicación emplearemos la propiedad "language of work or name" (idioma de la obra o del nombre) de Wikidata. Así tenemos que la canción *Talk* tiene idioma "inglés".

Figura 3.9: Ejemplo de explicación por idioma

La misma explicación podría para relacionar dos artistas con una misma propiedad de idioma, lo que significa que ambos artistas han trabajado interpretando canciones en ese idioma.

3.3. Explicaciones compartidas por Canción, Artista y Miembro

En esta sección solo aparecen las principales propiedades que por naturaleza son exclusivas de la canción, el artista o un miembro del grupo (en caso de que el artista sea una banda compuesta por varias personas).

Compañía discográfica

Record Label (compañía discográfica) hace referencia a la compañía por la que ha firmado el artista para publicar un tema concreto. Hemos podido observar que una misma compañía puede firmar a artistas que a veces no tienen relación ninguna en cuanto al estilo musical, pero hay otras causas que sí los pueden relacionar indirectamente como la tendencia del momento, el target del público que generan, etc. También se puede dar el caso en el que un miembro del grupo firme o haya firmado con una compañía individual, ya sea por conceptos relacionados con marcas, representantes o contratos. Además también tenemos en cuenta la compañía discográfica bajo la que se ha publicado cada canción, lo cual nos puede aportar relaciones adicionales.

Para ilustrarlo con un ejemplo, las canciones *Personal Jesus* de **Depeche Mode** y *Australia* de **The Shins** pertenecen a bandas que han trabajado con el sello discográfico **Columbia Records**.

Podemos alcanzar esta explicación gracias a la propiedad "record label" (sello discográfico) de Wikidata. *Personal Jesus* tiene un "record label" cuyo nombre (o valor) es **Columbia Records**.

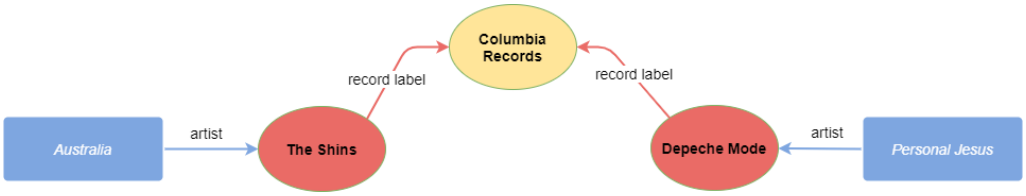

Figura 3.10: Ejemplo de explicación por compañía discográfica

3.4. Explicaciones compartidas por Género, Artista y Miembro

Estas explicaciones son compartidas por géneros, artistas y miembros, excluyendo solamente el nivel de canción.

Lugar de formación

La propiedad "location of formation" (lugar de formación) que hace referencia a la localidad de formación del grupo o artista, puede parecer muy similar a la explicación "country of origin", pero esta ejerce una selección más precisa porque nos da un núcleo de población más pequeño. Un ejemplo de ello podrían ser dos míticas bandas de la ciudad de Seattle: **Pearl Jam** y **Foo Fighters**. Creemos que la unión de la ciudad natal con la música puede hacer a un usuario el escuchar grupos de su misma ciudad o estado. En cuanto a la relación referente a un miembro del grupo, se trataría del lugar donde ese miembro se inició como músico ya sea como artista en solitario o en su grupo original.

De esta forma podemos explicar la relación existente entre sus temas *Black* y *Everlong*, respectivamente.

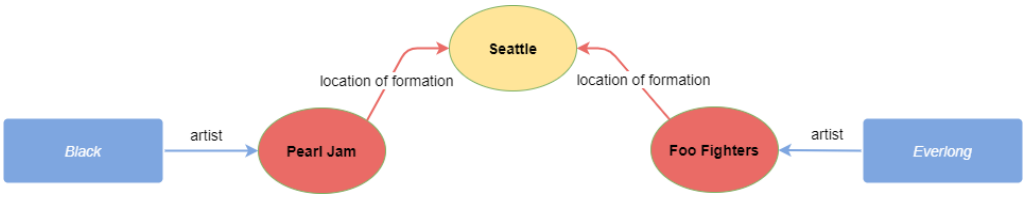

Figura 3.11: Ejemplo de explicación por lugar de formación

3.5. Explicaciones compartidas por Artista, Miembro, Género y Canción

Estas explicaciones son compartidas en los cuatro niveles y resultan de gran interés porque nos permitirán enlazar los cuatro mismos en el caso en el que coincidan en el objeto con las relaciones de la canción opuesta.

País de origen

Esta explicación puede ser especialmente útil para países relativamente pequeños donde la música nacional presenta unos patrones claros. Además, a lo largo de la historia en un país se genera una tendencia o nuevo género musical el cual es propio de ese país en el que se forma un artista y esto nos daría una relación muy específica que definitivamente acertaría por completo en la relación 1 a 1 de dos canciones.

En países que son significativamente grandes o con una población muy alta perdería algo de especificidad, pero aún así hay más posibilidades de que dos un artistas del mismo país sean escuchados por un mismo usuario.

La canción *Black Hole Sun* de la banda **Soundgarden** y *Closer* de **Nine Inch Nails** guardan cierta similitud porque ambos grupos son originarios de **Estados Unidos**.

Para comprobarlo, obtendremos la propiedad "country of origin" (país de origen) del artista que previamente ya hemos almacenado al estudiar las relaciones directas en el caso de las bandas o la propiedad "country of citizenship" (país de nacionalidad) en el caso de los artistas en solitario. *Black Hole Sun* tiene el "performer" **Soundgarden**, cuyo "country of origin" es **United States of America**.

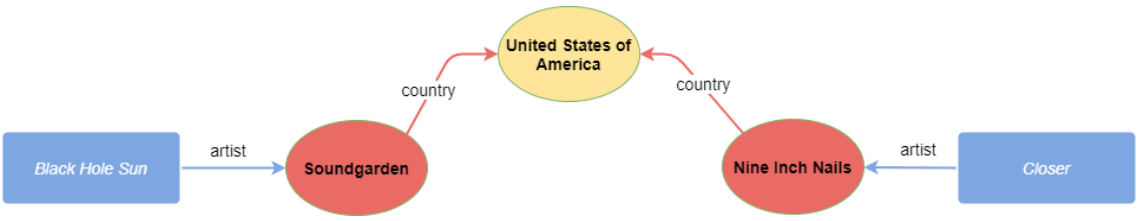

Figura 3.12: Ejemplo de explicación por país de origen

También podemos recoger la información del país donde se formó ese género. Con esta propiedad queremos recoger movimientos artísticos que surgieron en el mismo país. Los géneros que fueron formados en un determinado país están mucho más arraigados a este a pesar de que se hayan extendido por todo el mundo. Es por esto que si queremos relacionar una artista que cualquier parte del mundo, que tiene un estilo techno, será más propenso a estar relacionado con artistas o canciones holandesas, cuna de este estilo musical.

En el caso de que la propiedad se estableciese en la canción, significaría que ambas canciones han sido lanzadas originariamente en el mismo país.

Influenciado por

La **influencia de los artistas** es otra explicación importante. A menudo el trabajo de un artista se ve influenciado por otros artistas de formas que no siempre están ligadas a un género musical concreto, así que podemos encontrar una relación entre dos canciones examinando estas influencias.

Tomando como ejemplo las canciones *Billie Jean* de **Michael Jackson** y *Paint It Black* de **The Rolling Stones**, podemos establecer una relación entre ellas al ver que ambos

artistas fueron influenciados por la banda **The Beatles**.

Obtenemos esta explicación gracias a la propiedad "influenced by" (influenciado por) de Wikidata, además de la propiedad "performer" para relacionar el tema con su artista. Así averiguamos que *Billie Jean* tiene de "intérprete" a **Michael Jackson**, quien fue "influenciado por" **The Beatles**.

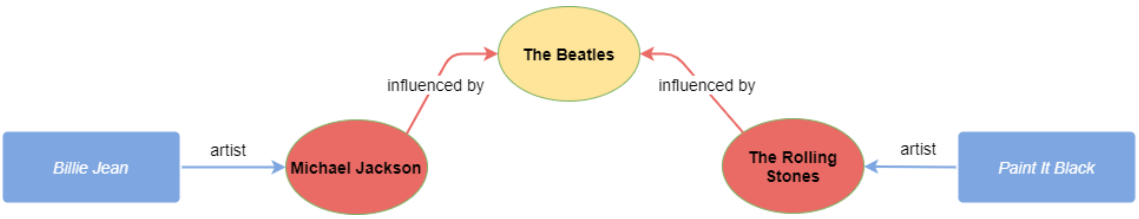

Figura 3.13: Ejemplo de explicación por influencia

Al igual que con el artista, los géneros también son influenciados por otros los cuales ha servido para el desarrollo de nuevos estilos a lo largo de la historia. Gracias a la evolución del Rock hay otros géneros que han marcado tendencia como el Rock and Roll, el Heavy Metal,Hard Rock, Garage Rock etc...

3.6. Propiedades propias del artista o grupo

Aquí se expones las principales relaciones que solo pueden ser propias del artista o grupo.

Integrantes

Los **integrantes** son cada uno de los miembros que conforman un grupo. En caso de ser un único artista, esta propiedad es irrelevante ya que haríamos referencia a la propiedad directa de artista. Esta propiedad resulta útil en casos en los que miembros de la banda han cambiado de grupo a lo largo de su carrera musical. Esto nos da acceso a otros grupos que por valores o gustos musicales del artista nos hacen pensar que pueden ser muy parecidos.

Si tomamos los temas *Lithium* de **Nirvana** y *Everlong* de **Foo Fighters** podemos observar que existe una relación entre ellos porque el músico **Dave Grohl** ha formado parte de ambas bandas.

Para comprobarlo usaremos la propiedad de Wikidata "performer" y después comprobaremos la propiedad "has part" (compuesto de) para ver todos los miembros de la banda y buscar coincidencias. De este modo, podemos ver que *Everlong* tiene un "performer" que es **Foo Fighters**, y a su vez **Foo Fighters** "has part" **Dave Grohl**.

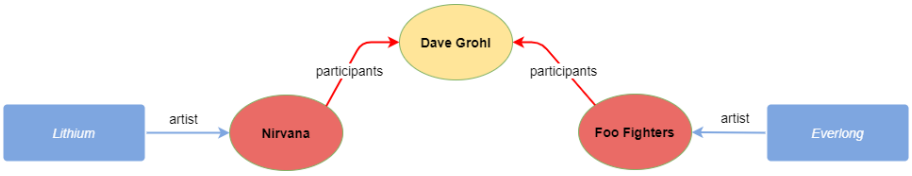

Figura 3.14: Ejemplo de explicación por integrantes

Tipo de voz

Siguiendo con el estudio de los artistas, el **tipo de voz** de los vocalistas es una buena explicación para relacionar dos canciones. El tipo de voz influye en el sonido general del tema y puede ser especialmente determinante en ciertos géneros musicales. Por limitaciones técnicas, solo aplicaremos esta explicación con artistas en solitario.

La voz de un artista puede encajar en más de un tipo, aunque en esta explicación buscamos que coincidan en al menos un tipo. La cantante **Lady Gaga** se asemeja a **Amy Winehouse** por su tipo de voz, ya que ambas son *mezzo-soprano*. Así podemos explicar la relación entre dos canciones de estas artistas como *Poker Face* y *Rehab*.

Para esta explicación resulta útil la propiedad "voice type" (tipo de voz) en Wikidata. Partiendo de *Poker Face*, utilizamos la propiedad "performer" para llegar a **Lady Gaga** y finalmente usamos su "voice type" para obtener *mezzo-soprano*.

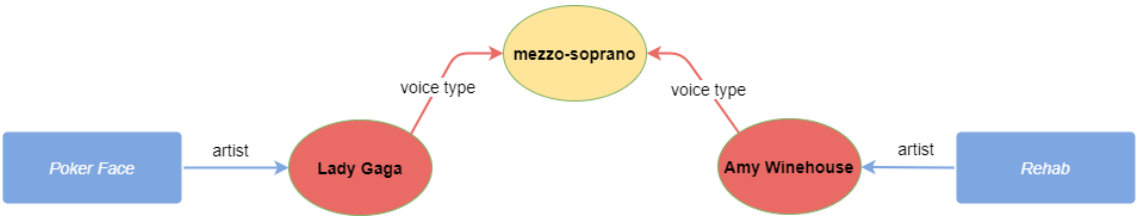

Figura 3.15: Ejemplo de explicación por tipo de voz

Fecha de fundación

Con esta propiedad intentamos tener en cuenta la historia de los o su cronograma. Hay grupos que comenzaron su etapa musical conjuntamente a lo largo del tiempo. Un ejemplo de este caso se en curiosamente en el technopop que tanto caracterizó a los años 80, pues convivió a veces con el soul y con el estilo afroamericano doo-wop.

Gracias a esta explicación podemos relacionar canciones como *Clint Eastwood* de **Gorillaz** y *Parabola* de **Tool**. Son canciones muy diferentes, pero ambos artistas comenzaron su carrera en la década de los 90, por lo que comparten un marco temporal.

Utilizaremos la propiedad "work period (start)" (inicio del periodo de actividad) de Wikidata para obtener esta explicación. *Clint Eastwood* tiene de intérprete a **Gorillaz**, que a su vez inició su periodo de actividad en la década de los 90.

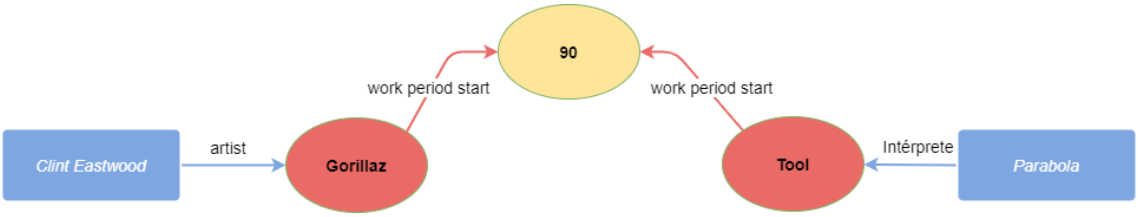

Figura 3.16: Ejemplo de explicación por fecha de fundación

3.7. Explicaciones basadas en el género

Estas propiedades vienen ligadas directamente al género de la canción. Es una de las propiedades más detalladas en Wikidata y por lo tanto una de de las que más información nos pueden proporcionar. Hay muchos géneros y subgéneros documentados, lo que nos facilita la obtención y clasificación de estos.

Subgénero

Cabe destacar la diferencia entre las propiedades "Influenciado por" y "Subgénero"; mientras que la primera solo indica que ese estilo musical ha marcado una cierta influencia tomando algunos aspectos, la segunda señala que es un Subgénero, una tendencia muy parecida y que toma como base ese estilo musical.

De esta forma podemos explicar la cercanía existente entre dos canciones como *One* de **Metallica** y *Schism* de **Tool**. La primera canción pertenece al género **thrash metal** mientras que la segunda es un ejemplo de **progressive metal** y, a su vez, ambos géneros son subgéneros del **heavy metal**.

Para esta explicación usamos la propiedad "subclass of" (subclase de) encontrada en Wikidata. Así tenemos que *One* pertenece al género **thrash metal**, el cual es "subclass of" **heavy metal**.

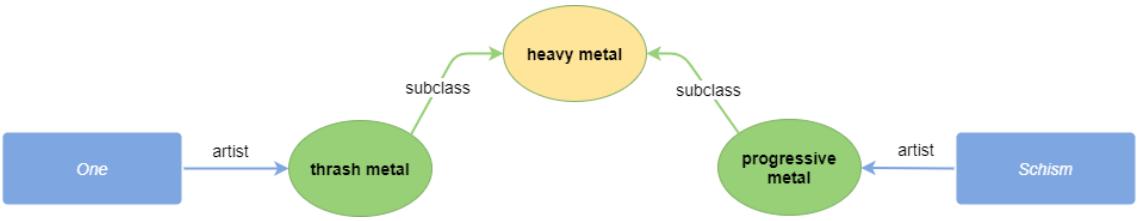

Figura 3.17: Ejemplo de explicación por Subgénero

Capítulo 4

Diseño de la interfaz de explicaciones

El objetivo de nuestro proyecto, como ya hemos expresado anteriormente en este documento, consiste en proporcionar explicaciones que justifiquen la relación existente entre dos canciones proporcionadas por un recomendador de música. Una vez realizado el estudio de las canciones y las distintas explicaciones posibles, es necesario mostrar el resultado de nuestro estudio de una forma gráfica, comprensible para un usuario humano.

Los datos que manejamos en nuestro estudio consisten en objetos y las relaciones que existen entre ellos, pues siguen el modelo de datos RDF. Necesitamos representar todos los caminos que se forman entre las dos canciones iniciales, mostrando las conexiones existentes entre los objetos intermedios. Por estos motivos hemos decidido llevar a cabo la visualización mediante un grafo porque consideramos que se adecúa mejor a nuestras necesidades.

Es importante que la interfaz sea fácil de entender y usar, así que siempre trataremos de mantenerla lo más sencilla posible. La base de nuestro diseño es el grafo de explicaciones ya mencionado, pues es el elemento más importante debido a la gran cantidad de información que aporta y, por lo tanto, es también la parte central de la interfaz alrededor de la cual se posicionarán el resto de elementos.

4.1. Diseño del grafo

En la Figura 4.1 se puede observar la primera iteración de nuestro diseño. En ella se ve un ejemplo de grafo donde los nodos son los distintos sujetos y objetos (según el modelo RDF), mientras que las aristas representan los predicados que relacionan a esos nodos.

Los nodos están coloreados de manera diferente en función de su categoría. El color rojo está asociado a los nodos de las **canciones**, el naranja a los nodos de los **artistas** y el morado a los nodos de los **miembros** de esos artistas (en el caso de que estos artistas sean agrupaciones musicales). Los nodos centrales, correspondientes a los objetos de las explicaciones, tienen los mismos colores que los nodos de los que proceden pero con un tono más suave para diferenciarlos.

Los nodos rojos posicionados a ambos extremos son las canciones sobre las que estamos realizando el estudio. A partir de ellas parten todas las explicaciones. En el centro se

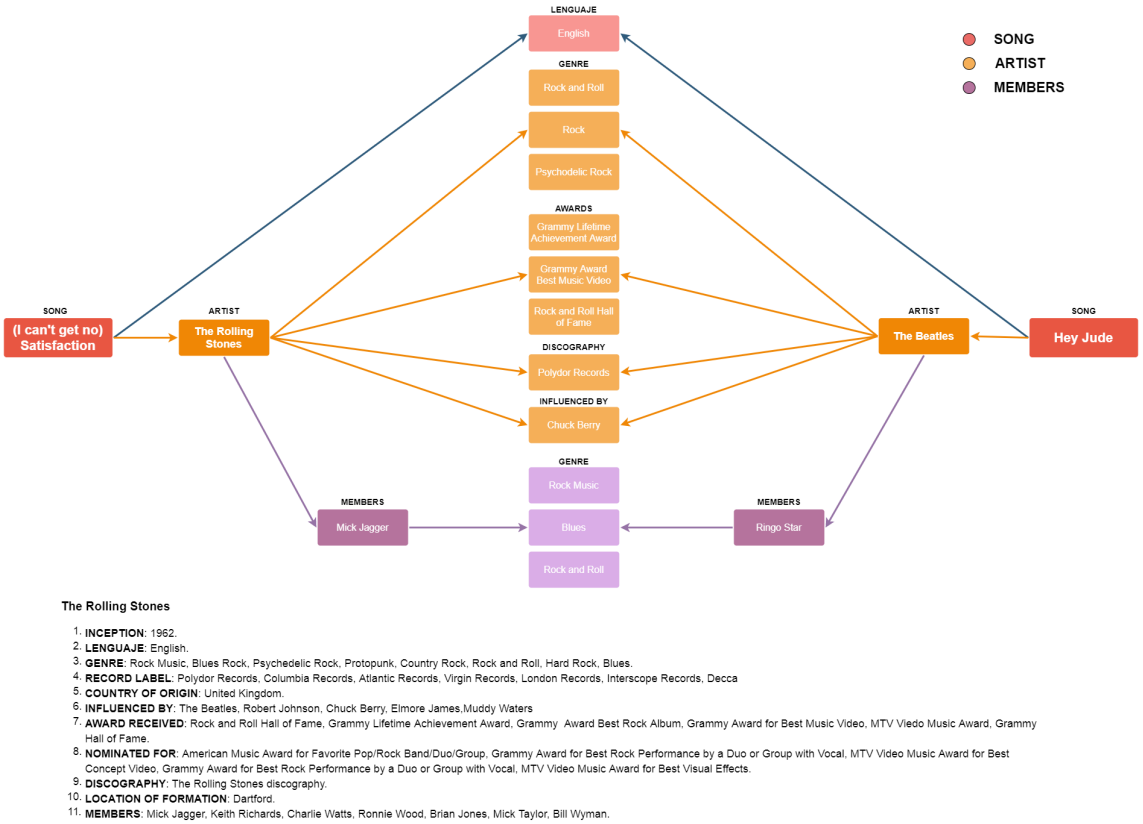

Figura 4.1: Primer diseño de la interfaz web

puede ver también un nodo coloreado de un rojo más suave, este representa al objeto de una explicación directa (en este caso la explicación "Idioma").

Los nodos coloreados de color naranja representan los artistas de ambas canciones y las relaciones que existen entre ellos, de la misma forma que en el anterior caso con las canciones y las explicaciones directas. Estas explicaciones obtenidas por el estudio de los artistas son indirectas, ya que requieren un estudio más profundo para relacionar las canciones originales.

Por último, tenemos nodos morados que representan a los miembros o integrantes de los artistas, pues en este caso dichos artistas son bandas formadas por varias personas. Su funcionamiento es el mismo que el descrito en el párrafo anterior, con la salvedad de que estos miembros se obtienen del artista y, por lo tanto, sus explicaciones presentan un nivel extra de profundidad. Las explicaciones obtenidas de estos miembros son indirectas, ya que no proceden de las canciones en sí, y se representan con nodos de un morado más claro.

En esta versión del diseño también se propone una funcionalidad que permite visualizar datos adicionales. Haciendo click en los nodos sobre los que se hacen los estudios principales (canciones y artistas), se despliega una sección inferior en la que se muestran todos los datos obtenidos en el estudio del elemento concreto, aparezcan en el grafo o no. Así podemos ver todos los sellos discográficos con los que trabajaron **The Rolling Stones**

aunque solo **Polydor Records** los relacione con **The Beatles**.

Esta función no es integral para el objetivo de nuestro proyecto, tan solo es un complemento que ofrece opciones al usuario, pero consideramos que puede ser un añadido interesante.

4.2. Diseño básico de la interfaz

Pasaremos ahora a explicar el resto de la interfaz con la ayuda de la Figura 4.2. En esta iteración, el diseño del grafo se mantiene intacto salvo un par de excepciones:

- Se han cambiado los colores que muestran las distintas categorías de los nodos. Ahora los nodos de las canciones y las explicaciones directas entre ellas son de color azul, los correspondientes a los artistas son de color rojo y los nodos de los miembros de los artistas son de color violeta. La decisión de cambiar estos colores se tomó para mejorar la visibilidad y obtener un grafo con mejor estética.

- La segunda novedad, más relevante que la primera, es que se ha ampliado la lista de explicaciones indirectas mostradas en el grafo. Ahora se incluyen las explicaciones obtenidas del estudio de los géneros musicales de la canción y se representan con nodos de color verde.

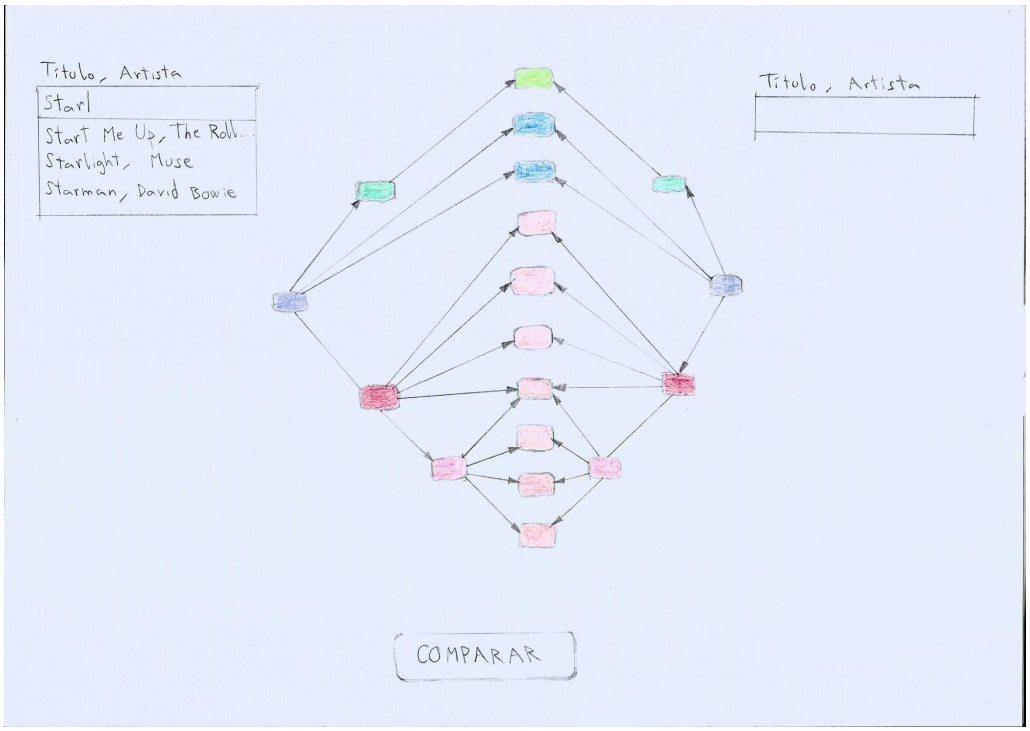

Figura 4.2: Segundo diseño de la interfaz web

El resto de la interfaz consiste en los elementos con los que interactuará el usuario antes de generar el grafo de explicaciones. A ambos lados de la pantalla se sitúan los campos a

rellenar con los datos básicos de las canciones. Para identificar la canción deseada es necesario escribir en ellos el título de la canción y el nombre de su artista, separados por una coma. En realidad este campo de entrada es un buscador que mostrará en un desplegable la lista de canciones que coinciden con el texto introducido y el usuario deberá seleccionar la que está buscando. Si intenta comparar una canción no contemplada por el sistema, aparecerá un mensaje de error.

La decisión de limitar las posibles entradas ha sido tomada por motivos prácticos. En Wikidata hay una cantidad ingente de temas musicales registrados, sin embargo los datos de cada canción concreta no siempre son tan completos como cabría esperar. A menudo falta información o la que hay no aparece siguiendo los mismos estándares que el resto, especialmente cuando se trata de canciones menos populares. Por ello hemos elaborado una lista de canciones que poseen información útil en Wikidata y que podemos tratar en nuestra aplicación sin problemas.

Por último, en la parte inferior de la pantalla hay un botón para "COMPARAR". Una vez rellenados los campos de las dos canciones con opciones válidas, el usuario deberá pulsar este botón para que se dibuje el grafo de explicaciones. La primera vez que se use la aplicación no aparecerá ningún grafo en el centro de la interfaz, pero en las comparaciones siguientes el grafo previo no se borrará hasta pulsar de nuevo este botón para dibujar un grafo nuevo.

4.3. Diseño avanzado

Partiendo de la base establecida en los apartados anteriores, hicimos un diseño más completo de la interfaz para nuestra aplicación. Aquí se puede ver el regreso de la sección inferior con datos adicionales de los artistas y las canciones, esta vez en forma de tabla para que sea más legible para el usuario.

También por motivos de visibilidad y comprensibilidad se ha desplazado el botón "COMPARAR" a la parte superior de la pantalla, además de mostrar una lista con todas las canciones disponibles bajo los campos de entrada. Esta decisión se ha tomado para tratar de evitar que el usuario se sienta perdido por la falta de opciones visibles al abrir la aplicación por primera vez. Cabe destacar que esta lista se sigue filtrando en función de la entrada que esté escribiendo el usuario, mostrando así cuáles son sus opciones en todo momento.

En esta iteración se hacen algunos cambios al grafo de explicaciones. Para empezar, el nombre de las explicaciones (el predicado según RDF) se traslada a las aristas. También se han unido todos los nodos que tengan las mismas aristas conectándolos a los mismos nodos padres, pues resultaba innecesario repetir todas esas aristas y entorpecía la lectura del grafo. En el ejemplo mostrado se aprecia este cambio en los premios recibidos por los artistas, que ahora están recogidos en un solo nodo.

De la misma forma se colapsan los nodos de los miembros de los artistas, reduciéndolos a un solo nodo por artista. Para expresar cuántos integrantes se ven involucrados en cada explicación, el grosor de la arista que los une varía en función del número de miembros que participan en esa explicación concreta. Además, al colocar el ratón sobre esas aristas

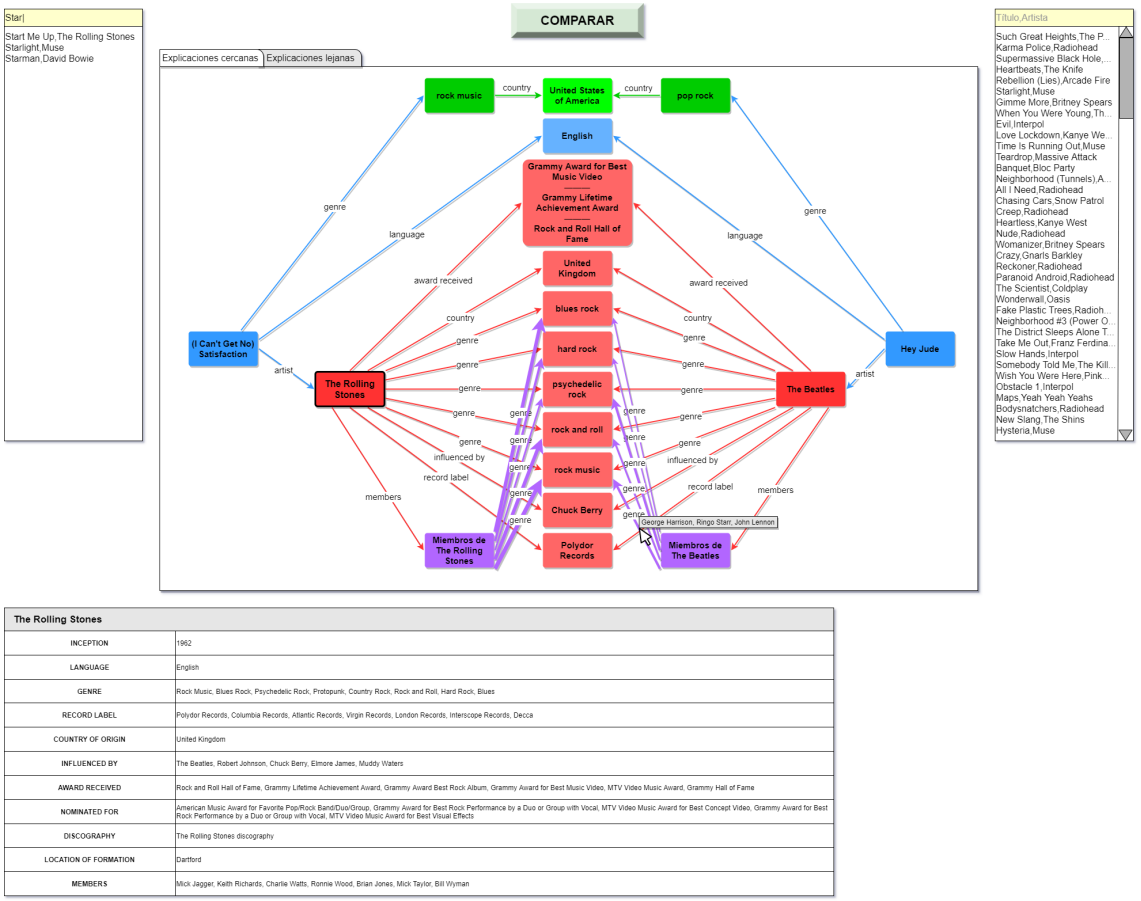

Figura 4.3: Tercer diseño de la interfaz web

aparece una etiqueta donde figuran los nombres de esos miembros. Este cambio se propuso para reducir el número de nodos y aristas, pues en ciertos casos podía ser abrumador a la vista.

Por último, cabe destacar las pestañas que se ven justo encima del marco para el grafo. Para tratar de mejorar la legibilidad lo máximo posible decidimos hacer dos grafos diferentes, siendo el que vemos en la Figura 4.3 el de "explicaciones cercanas". Este grafo muestra solamente las explicaciones existentes entre elementos del mismo "nivel de estudio". Esto quiere decir que, por ejemplo, el género de una canción solo podrá relacionarse con un género de la otra canción, pues corresponden al mismo estudio o categoría. Por el contrario, no se verán relaciones existentes entre ese género y otro elemento, como una canción, artista o miembro.

En la Figura 4.4 se ve el llamado "grafo de explicaciones lejanas", con el que se ve mejor la diferencia. Al contrario del "grafo de explicaciones cercanas", aquí solo se muestran las explicaciones existentes entre elementos que pertenecen a estudios diferentes. De esta forma, en este grafo se ve la relación entre el artista **The Rolling Stones**, autor de la primera canción, y el género **pop rock** obtenido de la segunda canción.

El usuario podrá cambiar de vista libremente para consultar ambos grafos haciendo click en las pestañas anteriormente mencionadas. Todas las funciones explicadas en esta

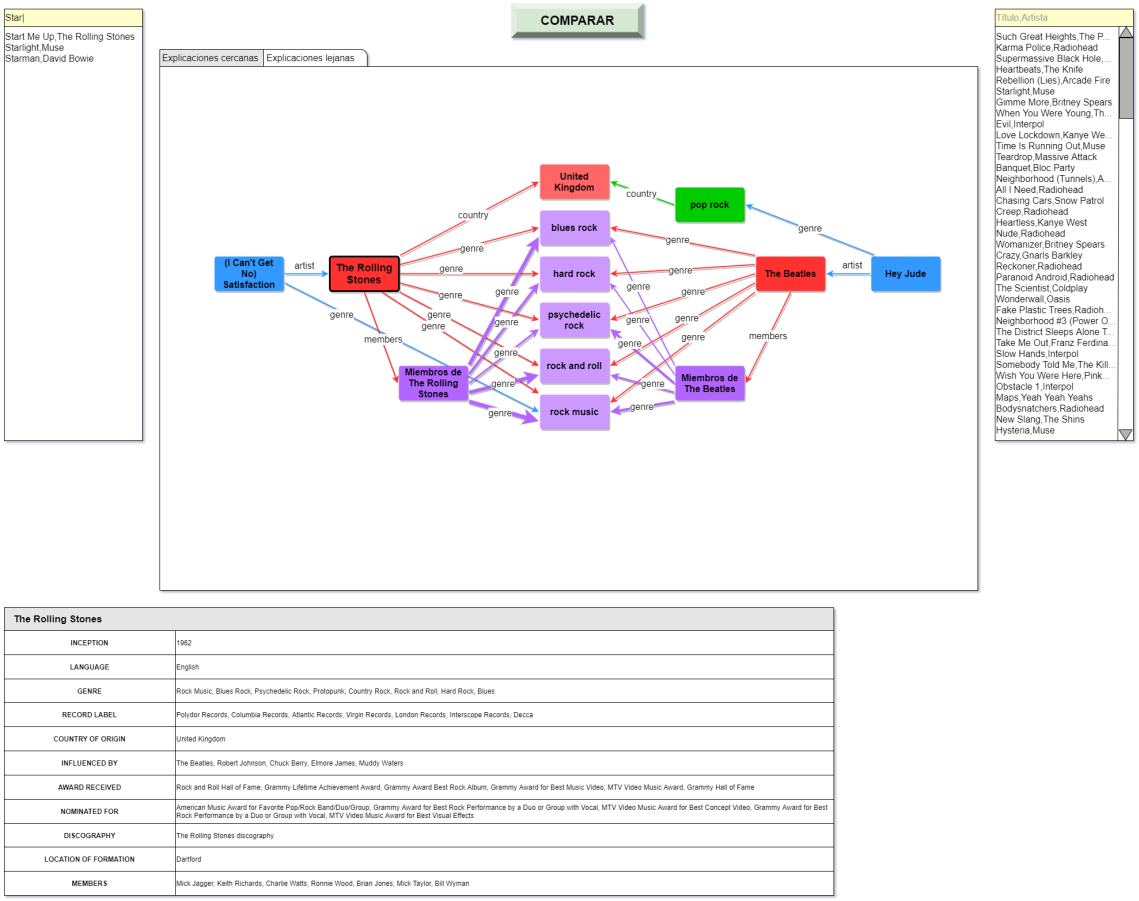

Figura 4.4: Grafo de explicaciones lejanas

sección aparecen en ambos grafos.

4.4. Estado final del sistema

Tras hacer una serie de cambios respecto a la versión anterior, alcanzamos la versión final de la interfaz. Aprovecharemos este apartado para hacer una demostración paso a paso del uso de la aplicación y comentaremos los cambios realizados a medida que vayan apareciendo.

Al abrir la aplicación, nos encontramos con la pantalla inicial que aparece en la figura 5.5. Al igual que en versiones anteriores, en este estado inicial tenemos dos listas a ambos lados para seleccionar las canciones a comparar, un botón en la parte superior para confirmar nuestra selección, un espacio central en blanco que será donde se dibuje el grafo y dos botones sobre dicho espacio para cambiar el grafo mostrado.

Aunque la estructura es la misma, el aspecto visual de casi todos los elementos ha cambiado con el objetivo de hacerlos más claros y agradables para los usuarios. Además, hemos decidido cambiar el idioma de todos los elementos al inglés, pues nos parece un lenguaje más universal.

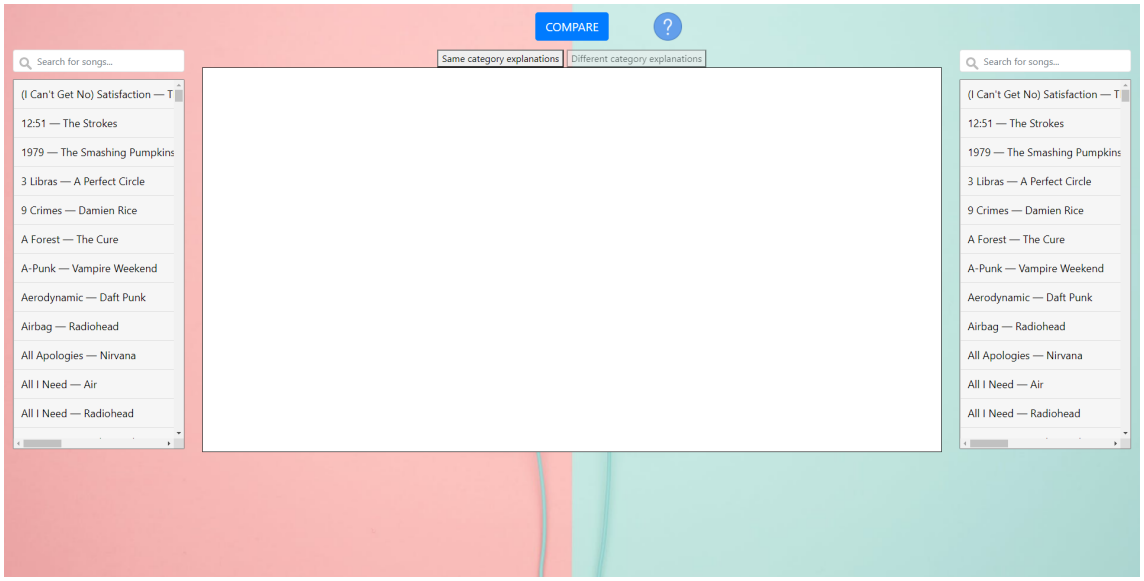

Figura 4.5: Estado inicial de la interfaz

La terminología utilizada para referirnos a los dos grafos mostrados también ha cambiado, como se puede ver en los botones relacionados. El que anteriormente llamábamos "grafo de explicaciones cercanas" ahora pasa a ser el "grafo de explicaciones de la misma categoría" (Same category explanations), mientras que el "grafo de explicaciones lejanas" se convierte en "grafo de explicaciones de categorías distintas" (Different category explanations). Su funcionalidad y forma de construirlos es la misma, pero decidimos cambiar los términos para intentar hacerlo más comprensible.

Somos conscientes de que la diferencia entre ambos grafos puede ser confusa para un usuario recién llegado a la aplicación, por eso hemos incorporado un botón de ayuda en la parte superior, reconocible por su forma de interrogación.

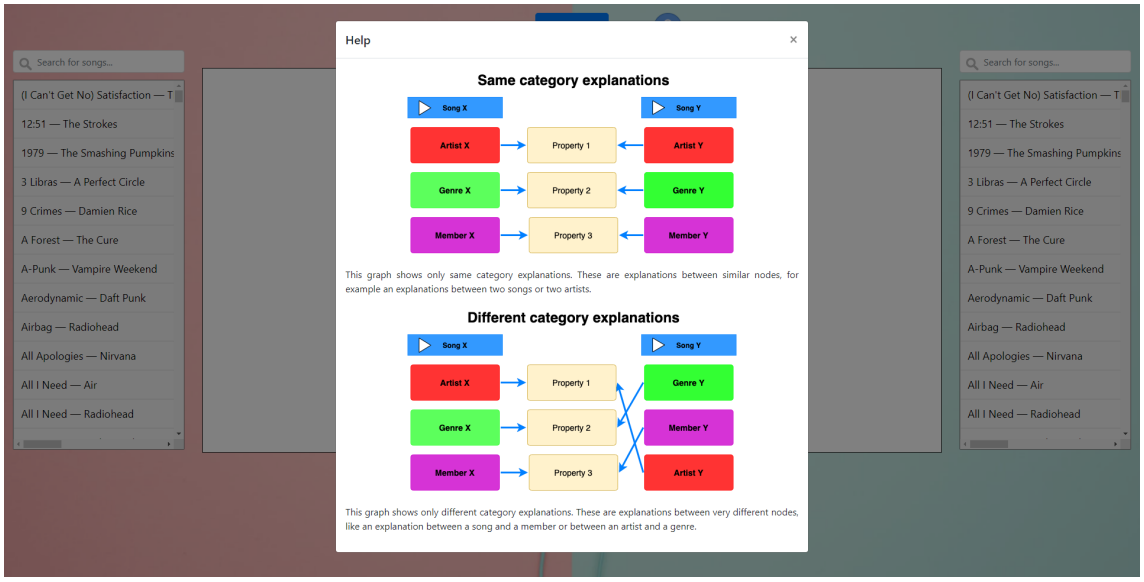

Figura 4.6: Ventana de ayuda

Al pulsar ese botón, se despliega la ventana mostrada en la Figura 5.6. En ella viene explicada la diferencia entre los dos grafos con la ayuda de imágenes y texto. Se puede cerrar esta ventana con la hélice de la esquina superior derecha o simplemente haciendo click fuera del marco.

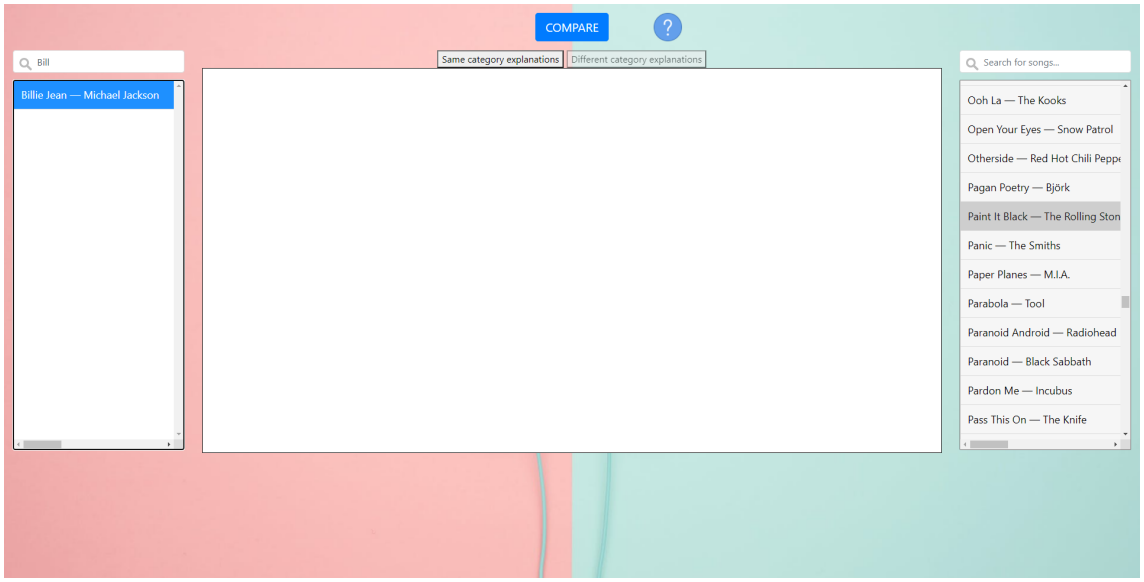

Figura 4.7: Ejemplo de selección de dos canciones

Para continuar es necesario seleccionar dos canciones, una de cada lista. Para ello es posible hacer scroll en las listas hasta encontrar la canción deseada o también se puede utilizar el buscador que hay sobre ellas para filtrarlas. Una vez se han seleccionado ambas canciones, se puede hacer click sobre el botón "COMPARE" para visualizar las explicaciones. Si se intenta proceder sin seleccionar dos canciones diferentes, el usuario recibirá un aviso.

Tras un pequeño tiempo de carga, el grafo de explicaciones que relaciona ambas canciones aparecerá en el espacio reservado del centro de la pantalla. Este grafo tiene la misma forma que en la última versión, con colores distinguiendo las distintas categorías (azul para las canciones, verde para los géneros, rojo para los artistas y morado para los miembros de los artistas).

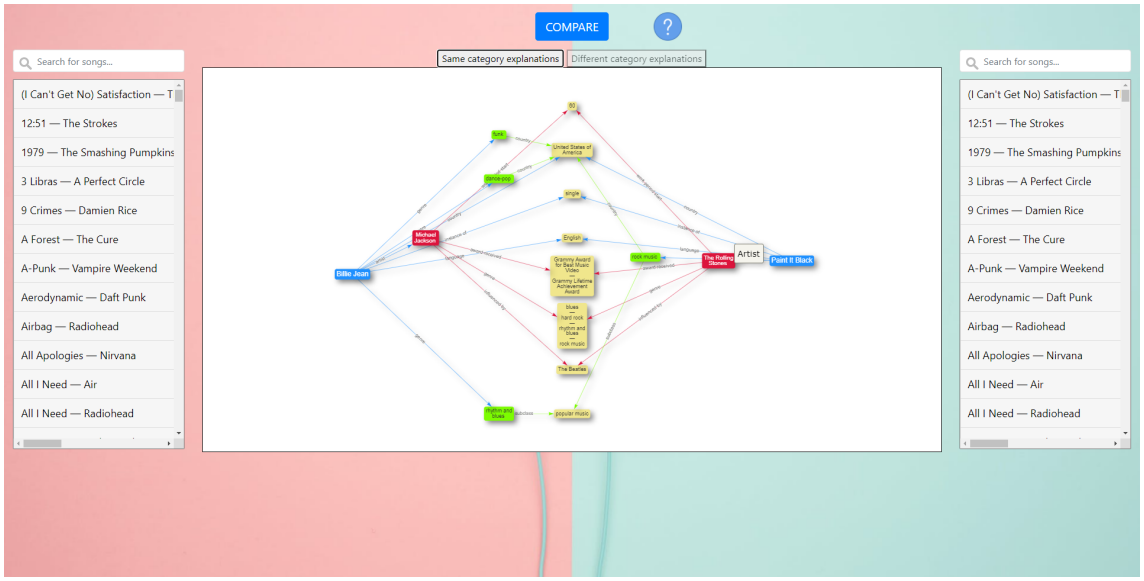

Figura 4.8: Grafo de explicaciones de la misma categoría

En esta versión final hemos cambiado el color de los nodos centrales por un tono amarillo para diferenciarlos del resto. En esta columna central aparecen los nodos que representan realmente las explicaciones que relacionan las canciones.

También hemos cambiado el tamaño y color de la fuente para los nodos de canciones y de artistas para que sea más fácil identificarlos a simple vista. Esto es importante debido a que esos son los datos que el usuario introduce a la aplicación y debería ser capaz de reconocerlos rápidamente.

Por último hemos añadido una etiqueta desplegable en los nodos, de forma que se puede colocar la flecha del ratón sobre ellos para saber qué tipo de elemento es.

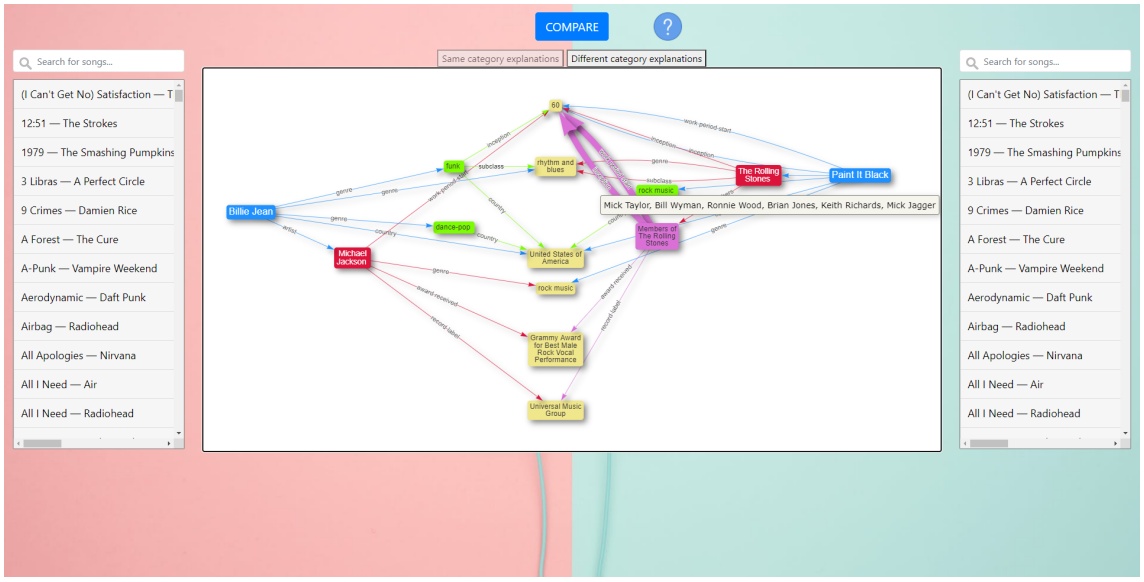

Figura 4.9: Grafo de explicaciones de distinta categoría

Al pulsar la otra pestaña, "Different category explanations", se muestra el grafo alternativo. Este grafo no ha sufrido cambios notables desde su última versión más allá de los ya comentados en este mismo apartado.

Hay que mencionar que se puede interactuar con los grafos en cualquier momento. Si se sitúa la flecha del ratón dentro del marco blanco, la rueda sirve para ajustar el zoom. Además, se puede pinchar y arrastrar cualquiera de los nodos para resaltarlos y moverlos verticalmente. Estas acciones pueden resultar de gran ayuda, especialmente cuando el grafo es muy grande y resulta difícil ver todas las conexiones al mismo tiempo.

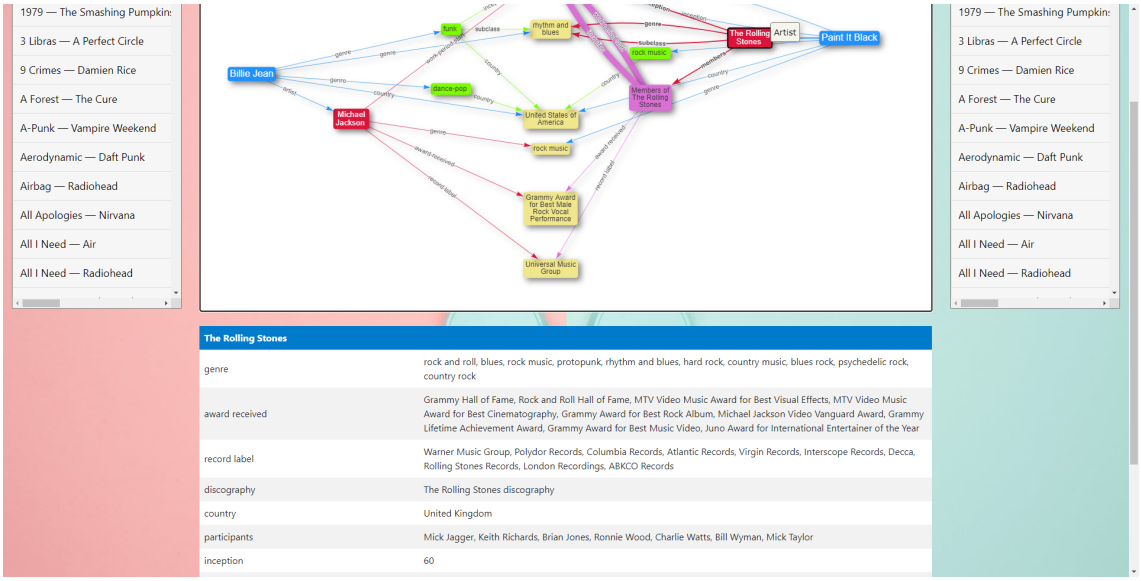

Figura 4.10: Tabla de datos adicionales

Por último, al hacer click en alguno de los nodos de canciones o artistas se despliega una tabla en la parte inferior de la pantalla. En ella se puede consultar una serie de datos adicionales que, si bien no todos forman parte de las explicaciones mostradas, pueden servir para que el usuario tenga un mejor entendimiento de ambas canciones y los artistas que las interpretan.

Capítulo 5

Implementación

5.1. Procesamiento y manejo de la muestra

Partimos de una muestra (o **dataset**) de alrededor de 19 millones de líneas, obtenido a partir de la API de **Last.fm** [5], una plataforma que almacena y proporciona mucho contenido musical. Constaba de los siguientes campos: <**user, timestamp, artist, song**>, los cuales hacen referencia al usuario que ha escuchado la canción, la fecha en la que fue escuchada, el nombre del artista y el título de la canción respectivamente.

El primer paso fue hacer un pequeño estudio del dataset para hacernos una idea de cómo era nuestra muestra y qué podríamos sacar de ella. Se trataba de un dataset con muy pocos campos que no daba información ninguna sobre la canción o el artista, sino que solo proporcionaba un medio para poder obtenerla de una fuente externa. Hicimos una limpieza del dataset, ya que había numerosos valores que eran nulos en determinados campos o no tenían una codificación correcta.

Es en este punto donde comenzamos a pensar cómo queríamos mostrar la información que podíamos ofrecer previa al funcionamiento de la aplicación. Dudábamos entre permitir utilizar cualquier objeto del dataset o restringirlo solo a algunos.

En la librería de Wikidata, por motivos obvios, se debe poner como entrada de cualquier función de búsqueda el código identificador del objeto sobre el que se quiere obtener información. Estos identificadores son fijos y únicos para cada uno de los objetos que están registrados en la página. Con nuestra librería somos capaces de, a partir de un string que represente un título de canción o un artista, género, etc., obtener su respectivo identificador para posteriormente procesar las consultas.

El problema aquí es que para obtener el identificador del objeto, su nombre o título debía ser exacto al que aparecía en Wikidata, pues de cualquier otra forma se lanzaría una excepción. Por ejemplo, intentar obtener el identificador de la canción "Don't Stop me Now" sería incorrecto, porque en Wikidata figura como "Don't Stop Me Now". Debido a esto, es necesario hacer un parseo previo de cualquier String (o cadena de caracteres) que se vaya a utilizar como entrada.

Finalmente decidimos crear una lista preseleccionada y parseada de las canciones más

populares de todo el dataset. Para ello ordenamos las canciones del dataset por popularidad descendente, entendiendo como popularidad la cantidad de veces que aparecían en la muestra. Después elaboramos un script que recorría todas ellas, ejecutaba el parseo y finalmente comprobaba si era posible obtener los datos de Wikidata mediante una llamada a un método de la librería SPARQLWrapper que retornaba un objeto si se había encontrado información del artista o una excepción en caso contrario. Finalmente obtuvimos una lista con tuplas de canciones-artista con las que trabajar sabiendo que no íbamos a obtener fallos o excepciones(sin tener en cuenta la posible cantidad de datos que podríamos obtener de cada una de ellas).

Empezamos con un dataset de las 2500 canciones más populares y fuimos capaces de obtener la información de 1408 canciones con su determinado artista. Cabe señalar que en cada búsqueda de una canción se debe añadir su artista, pues hay varias canciones con el mismo título que no podrían diferenciarse de otro modo.

Nuestra aplicación final funciona con este dataset limitado pero que cuenta con la seguridad de que se pueden obtener datos fiables sin importar la canción que se elija. Además, al haber escogido las canciones con mayor popularidad nos vamos a encontrar con mayor cantidad de datos ya que estas eran las que más documentadas estaban. a diferencia de las que estaban en la parte inferior del dataset, que eran muy poco conocidas y apenas se podían sacar datos valiosos sobre ellas.

5.2. Estudio del dataset

Para poder comprender mejor el dataset obtenido, estudiamos el género de todas las canciones de la muestra con el objetivo de tener conocimiento de cuán diferente o similar era el dataset, musicalmente hablando. Objetivamente no es igual de complejo poder relacionar dos géneros dispares como podrían serlo Folk y Electrónica que otros dos géneros que pueda ser parejos en algunos aspectos musicales, sociales etc... como lo son el Rock y el Heavy Metal.

Con el género queremos demostrar la variedad de la muestra y cómo de posible es relacionar canciones de distinto género entre sí. Si hubiésemos obtenido muy pocos géneros similares entre sí, la muestra no tendría un gran valor desde el punto de vista de esta explicación, pues no habríamos llegado a un resultado realmente interesante. Cabe destacar que no es un conteo exacto, ya que una canción puede tener más de un género o ninguno (por falta de documentación, no es posible catalogarlo en una categoría exacta, etc.). Sin embargo, el componente de variedad y representación musical no se ve afectado.

En total hemos obtenido 81 géneros distintos para las canciones, entre los que destacan en gran principalmente subgéneros o géneros derivados del rock. El estilo más popular es **alternative rock**, el cual cuenta con un 22 % sobre el total, lo cual es una cantidad bastante alta en comparación con el resto. Le siguen **indie rock** y **rock music** con un 8,3 % y 3,9 % respectivamente.

Sin embargo, también encontramos otros géneros más distantes a los anteriores, como por ejemplo **rhythm & blues**, **soul**, **reggae**, **jazz** o **house music**. Todos estos son estilos dispares entres sí, lo que nos da la posibilidad de ver cómo de potentes pueden ser las relaciones y explicaciones que hemos obtenido y probar nuestro sistema para ver si es capaz de relacionar canciones muy diferentes o incluso opuestas.

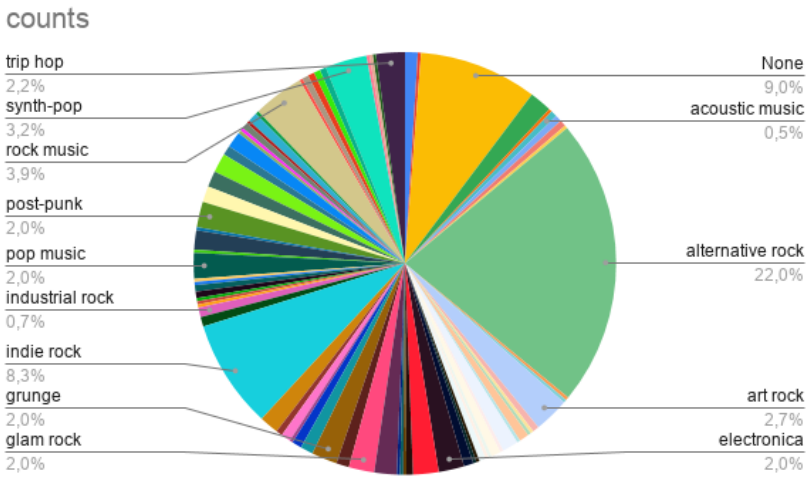

Figura 5.1: Gráfico resultado del estudio de géneros

Por último hemos querido representar una red de toda nuestra lista. Para ello hemos establecido las canciones como nodos y las aristas como relaciones que se establecen en función de si podemos encontrar al menos 3 explicaciones de relación (musical) entre dos nodos de nuestra lista. El objetivo es conocer mejor nuestro dataset de canciones y ver cómo de relacionadas están entre ellas y cuáles son los nodos centrales de toda la lista.

(AQUÍ FALTA AÑADIR UNA IMAGEN. ESTARÁ LISTA EN LOS PRÓXIMOS DÍAS)

5.3. Arquitectura de la aplicación

Como punto inicial hemos utilizado varios scripts que nos ayudaron a organizar y manejar mucho mejor los datos iniciales que nos fueron proporcionados, como por ejemplo el script de limpieza y organización del dataset que, como ya hemos explicado antes, elimina cualquier tipo de valor nulo y lo ordena por popularidad, o el script de parseo, que nos permite acercarnos más a la sintaxis gramatical de Wikidata.

Como punto troncal, hemos desarrollado una librería que nos permite trabajar con la API de Wikidata. Hace uso de SparQL Library, la cual nos permite hacer uso del lenguaje SPARQL y hacer todo tipo de consultas a un punto. Esta librería se encarga de crear un punto de conexión a la API de Wikidata y de crear y ejecutar queries de consulta u obtención de datos hacia ese punto, el cual debe permitir el intercambio de datos en formato RDF. Además nos proporciona un objeto respuesta, el cual puede devolverse en distintos formatos.

A la hora de organizar el código de nuestra aplicación, nos hemos decantado por el **patrón MVC (Modelo-Vista-Controlador)** debido a su utilidad en el desarrollo y mantenimiento de aplicaciones web. Por ello, la lógica de nuestra aplicación se encuentra dividida en diferentes carpetas según el patrón, además de un par de carpetas auxiliares.

La carpeta **model** contiene todo lo necesario para el Modelo, incluyendo la librería anteriormente mencionada y las distintas clases empleadas en la aplicación, que pasaremos a explicar a continuación.

La clase principal del modelo de la aplicación es **getInfoSongs**. Esta clase hace uso de una instancia de la librería SPARQL e inicia una cadena de métodos los cuales obtienen información de distintos ámbitos de una canción proporcionada como entrada.

La clase **Properties** se encarga de establecer y configurar todas las propiedades que se van a estudiar sobre cada canción. Al crear una instancia de la clase, importa una serie de diccionarios que son almacenados. Cabe destacar que estos diccionarios son inicialmente estáticos ya que en un principio cuentan con las propiedades que hemos estudiado y probado aunque tienen la posibilidad de expandirse añadiendo nuevas propiedades, siempre que respeten la sintaxis de Wikidata.

Para terminar con el modelo tenemos el módulo **dataOperations**, el cual es utilizado a modo auxiliar para poder convertir y hacer operaciones con los distintos dataFrames de información que se van creando continuamente al hacer un estudio de una canción.

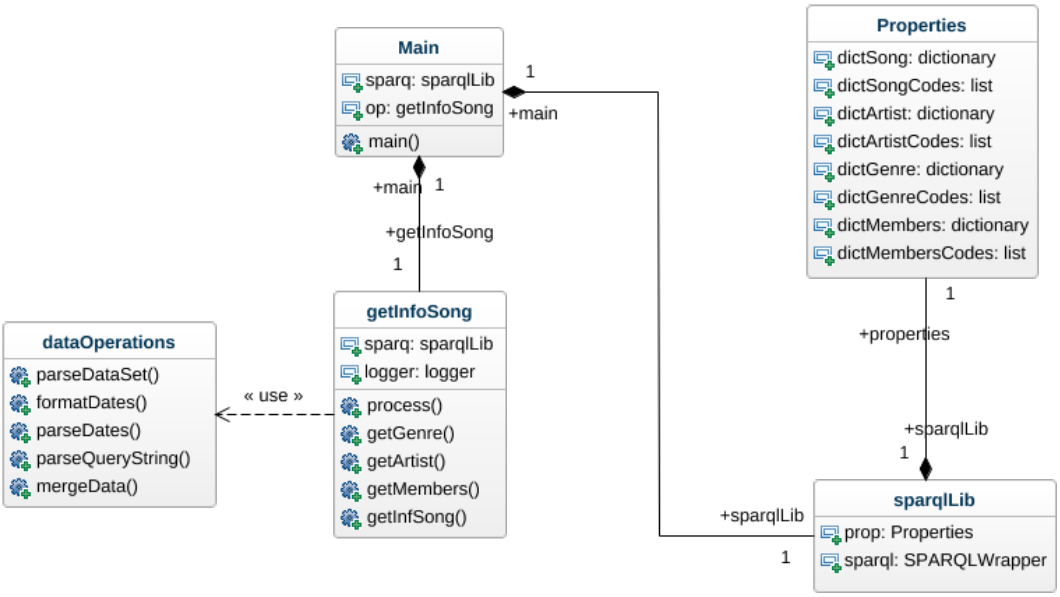

Figura 5.2: Diagrama de clases sobre el modelo de datos

Por otra parte tenemos la carpeta **view**, donde se encuentran las Vistas de la aplicación. Aquí es donde almacenamos el código HTML de la interfaz, aunque esta no estaría completa sin el contenido de la carpeta **static**, que es donde se recogen todos los recursos estáticos de la aplicación, como lo son el código CSS para dar formato a la vista o el JavaScript para aportarle funcionalidad.

A continuación está la carpeta **controller**, que contiene todas las rutas de la aplicación. El controlador actúa como intermediario entre el modelo y la vista, es el encargado

de reaccionar a las acciones del usuario en la interfaz y hacer peticiones al modelo para obtener la información que se necesite mostrar en cada momento.

Por último hay una carpeta adicional llamada **helpers**. En ella se recogen las funciones necesarias para tratar los datos obtenidos del modelo y elaborar a partir de ellos la representación gráfica que se muestra en la interfaz. Dicho de otra forma, es la parte responsable de generar los grafos de explicaciones. Para esto hacemos uso de vis.js, una librería de JavaScript para visualización de datos.

5.4. Extensibilidad de la aplicación

Se ha intentado dar un enfoque lo más aproximado posible a una estructura MVC (Modelo-Vista-Controlador), con tal de poder hacer una aplicación lo más reutilizable posible y atenta a posibles cambios y modificaciones.

Para empezar, todas las propiedades y sus códigos para consultas están incluidas en diccionarios que son usados por el modelo. En un inicio creábamos queries con unos valores estáticos predeterminados pero nos dimos cuenta de que ese código no iba a poder ser reutilizable y era muy poco práctico. Poniendo un ejemplo, imaginemos que Wikidata implementa nuevas propiedades o nueva información en la base de datos de los artistas que añadir a nuestro estudio, contamos con métodos de adición con tan solo obtener el nombre y el código de esa nueva propiedad. El modelo no sufriría cambios en el código, ya que importa en cada lanzamiento todos estos diccionarios estáticos. De hecho, Wikidata es una base de datos que cambia constantemente por acción de diversos usuarios que completan y actualizan documentos constantemente, así que no es de extrañar que en cierto tiempo las explicaciones y por ende las propiedades activas, se queden obsoletas.

También como resultado final del modelo se proporciona un dataset en forma de archivos .json que contienen todas las relaciones con sus respectivos objetos y 4 datasets que representan información extra de las dos duplas canción/artista, de forma que son archivos estáticos y únicos en cada lanzamiento, nunca va a enviarse un número distinto de archivos.

En cuanto a la extensibilidad del DataSource de canciones y artistas, se trata de un módulo separado el cual utiliza como fuente el DataSet original de lastFM. El módulo se encarga de parsear los strings y procesarlo como hemos explicado en la sección de Preprocesamiento. No es una función que se pueda lanzar desde el main de la aplicación porque no está pensada para añadir canciones a voluntad por motivos de compatibilidad con la sintaxis y el manejo de Wikidata.

5.5. Tecnologías y Herramientas utilizadas

5.5.1. Programación

Python

Ha sido el lenguaje de programación principal, al menos en la parte del backend tanto para el núcleo del código que obtiene y relaciona los datos como para la creación de distintos

scripts de ayuda, además de numerosas funciones para tratar los datos a representar en los grafos finales. Lo hemos escogido debido a su versatilidad y a su gran adaptación debido a la cantidad de librerías con las que cuenta.

Pandas y Numpy

Dos librerías propias de Python, especializadas en el manejo y procesamiento de datos. Han sido de una utilidad vital ya que en gran parte del trabajo trabajamos con numerosos datasets y estas librerías nos proporcionan todo lo necesario para tratarlos, pudiendo así trabajar con ellos más fácilmente.

Sanic

El framework seleccionado para desarrollar nuestra aplicación. Es un framework web asíncrono para Python cuyo objetivo es proporcionar una forma de crear un servidor que sea rápido y fácil de usar. Su sencillez para empezar a utilizarlo es uno de los principales factores que nos hizo decantarnos por él en lugar de otras opciones.

Jinja2

Un motor de plantillas para Python basado en el sistema de plantillas de Django. Permite trabajar con documentos HTML con marcadores de posición que son llenados por lo que se le indica desde el código Python, permitiendo así utilizar variables en las vistas.

Esta función se utilizaba en una versión antigua de nuestro proyecto para tratar correctamente documentos cuyo título depende de la fecha y hora de su creación. Más adelante se cambió la forma en que se trataban estos documentos, pero seguimos utilizando Jinja2 para el formato de uso de las plantillas.

SPARQLWrapper

Un wrapper para Python que permite ejecutar consultas SPARQL de forma remota. Proporciona una funcionalidad esencial para nuestro trabajo, pues es lo que utilizamos para obtener la información de Wikidata desde nuestra aplicación.

HTML y CSS

Dos lenguajes fundamentales para la programación web. Debido a la naturaleza de nuestra aplicación hemos recurrido a estos lenguajes para darle forma y estilo a la interfaz de explicaciones, aquella parte de nuestro proyecto con la que interactuarán los usuarios.

Javascript

Al igual que los anteriores, este es un lenguaje de programación muy importante para el desarrollo web. Javascript es una parte integral de nuestro proyecto, pues es el lenguaje en el que se ha desarrollado la parte encargada de dibujar los grafos de explicaciones, además de otras funciones necesarias para el funcionamiento de la interfaz.

vis.js

Esta es una librería de Javascript para visualización de datos. Permite diversas representaciones gráficas, pero nosotros hemos empleado el componente Network para dibujar nuestros grafos de explicaciones. Es una librería bastante completa y con una documentación bien organizada, así que fue muy útil a la hora de plasmar en pantalla el resultado de nuestra investigación.

Jupyter-Notebooks

Es un entorno informático interactivo basado en la web. Fue un entorno apropiado para realizar la prueba de scripts que nos ayudaron a limpiar y probar el dataset original.

5.5.2. Organización

Github

Para poder almacenar y organizar todo el código en el que hemos trabajado conjuntamente. Nos ha permitido llevar un historial de versiones y actualizaciones de cada módulo del código. Github ha sido una herramienta apropiada no solo para llevar un control del código de la aplicación, sino también para la construcción de este mismo documento.

Google Drive

Empleamos el servicio de alojamiento de archivos en la nube de Google para recoger y poner en común todos los documentos relacionados con la investigación previa al inicio del trabajo, además de para compartir recursos durante la realización del mismo. Su importancia quedó relegada a un segundo plano a medida que avanzaba el proyecto debido a que comenzamos a utilizar Github, pero cabe resaltar su utilidad durante las primeras fases.

Google Meet

La aplicación de videoconferencias de Google fue una herramienta clave para mantener el contacto tanto con los directores del TFG como entre los miembros del equipo. Cuando las reuniones presenciales dejaron de ser posibles por motivos ajenos a nuestro control, se hizo necesario el uso de un servicio como este.

5.5.3. Memoria

LaTeX

Este ha sido el procesador de textos elegido para la realización de este documento: la Memoria del TFG. Nos decantamos por este en lugar de otros procesadores como Word debido a las muchas posibilidades que tiene para generar documentos de calidad. Puntos como la estructura de capítulos, la estandarización de títulos o la forma de mostrar figuras (tanto imágenes como fragmentos de código), han hecho que nos resulte más sencilla la tarea de desarrollar esta memoria.

TeXiS

TeXiS es una plantilla de LaTeX para Tesis, Trabajos de Fin de Máster y otros documentos desarrollada por Marco Antonio y Pedro Pablo Gómez Martín. Es la plantilla que se ha usado como base para construir este documento y ha resultado de gran ayuda tanto para cuestiones de organización como para aprender a usar varias funcionalidades de LaTeX, lo cual ha sido muy importante debido a nuestra falta de experiencia previa a este proyecto.

5.5.4. Tecnologías y herramientas descartadas

RDFstarTools

Esta es una colección de librerías Java y herramientas de línea de comandos para procesar datos RDF* y consultas SPARQL*. Proporciona varias funcionalidades, pero la verdaderamente relevante para nuestro proyecto es SPARQL* Parser, que sirve para hacer consultas SPARQL. Este parser está implementado sobre el framework Apache Jena.

Esta fue una de las opciones que barajamos para hacer consultas SPARQL desde nuestro código, pero acabamos decidiendo usar SPARQLWrapper en su lugar debido a que RDFstarTools es una colección de librerías de las cuales solo nos haría falta una pequeña parte. Tomando en consideración ambas opciones, nos pareció más adecuado utilizar Python junto con SPARQLWrapper debido ya que era más conciso y sencillo de implementar.

Java

Uno de los lenguajes de programación de programación más extendidos y populares, especializado en la programación orientada a objetos. Al principio del proyecto nos planteamos utilizarlo como lenguaje principal para el backend de nuestra aplicación, sin embargo finalmente nos decantamos por Python.

Como ya hemos comentado antes, al decidir usar SPARQLWrapper para nuestras consultas SPARQL en lugar de RDFstarTools también nos pareció más adecuado descartar Java en favor de Python, aunque ambos habrían sido elecciones viables.

Alchemy.js

Alchemy.js es una aplicación de visualización de grafos para la web. Está escrita en JavaScript con la librería D3.js como base y ofrece una manera sencilla y rápida de generar grafos. La mayoría de su personalización se lleva a cabo sobrescribiendo sus configuraciones por defecto, por lo que no requiere apenas implementar código JavaScript adicional.

Fue la primera opción contemplada para representar los grafos de nuestro proyecto debido a su sencillez de manejo y a su uso de archivos JSON para aportar los datos, algo que resultaba atractivo en un principio. Tras trabajar con ella durante un tiempo fue necesario descartarla por ciertas limitaciones a la hora de personalizar la representación según nuestro diseño, además de la falta de soporte por tratarse de un proyecto actualmente abandonado.

Capítulo 6

Conclusiones y Trabajo Futuro

6.1. Conclusiones

Tras trabajar con la Web Semántica y Linked Data, hemos podido comprobar que son métodos de organización de datos con mucho potencial y del que todos podríamos beneficiarnos si se explora más allá de su estado actual. Su mayor inconveniente quizás sea precisamente que su utilidad depende en gran medida de lo extendido que esté su uso y, por desgracia, actualmente no está tan normalizado como nos gustaría.

Tomando como ejemplo el dominio en el que hemos centrado nuestro proyecto podemos ver estas limitaciones. Gracias a Linked Data y el formato RDF hemos sido capaces de obtener una buena cantidad de información que nos ha permitido generar explicaciones para la relación existente entre muchas canciones diferentes, sin embargo este trabajo no es suficiente para aplicar el mismo tratamiento a todas las obras musicales existentes, ni siquiera en la plataforma que hemos elegido para la obtención de datos.

Escogimos la web Wikidata para estudiar las canciones debido a su tamaño y a que utiliza efectivamente el modelo de datos enlazados. No obstante, esta plataforma dista mucho de ser perfecta debido a que no alberga la misma cantidad de información de todos sus elementos incluso aunque esa información exista, por lo que hay canciones que hemos podido explorar en más profundidad que otras. Por otra parte también hay casos de información poco fiable o directamente incorrecta, un problema con el que nos hemos topado en alguna ocasión y que provoca que no se pueda utilizar nuestra aplicación correctamente en esos casos.

A pesar de los problemas mencionados, Wikidata sigue siendo la mejor opción que hemos encontrado para alojamiento de información de música. Otras plataformas no hacen uso de Linked Data o lo hacen de tal forma que resulta muy difícil establecer relaciones en sus elementos porque sus elementos son excesivamente concretos, llegando a tener muchas entradas diferentes para elementos que son iguales en esencia. Trabajos como el nuestro se beneficiarían de una mayor variedad en las ofertas de bases de conocimiento que aprovechen las ventajas de la Web Semántica.

6.2. Trabajo futuro

El esquema RDF y el uso de SPARQL para su uso y análisis puede ser muy potente y abarcar grandes cantidades de datos. En nuestro caso nos hemos centrado en trabajar con la herramienta y los datos de Wikidata por una cuestión de cantidad de información, fiabilidad y resultados obtenidos.

Con Wikidata hemos sido capaces de crear conexiones entre canciones con toda la información posible en cada una de sus respectivas páginas descriptivas. Sin embargo hay más opciones de donde recopilar más información sobre canciones y artistas.

En un principio intentamos obtener información que se alejaba un poco más de lo que hemos creado, que es un análisis musical. Hay otras opciones y datos que se pueden sacar sobre canciones, como por ejemplo las películas o series en las que aparecen como banda sonora, o también canciones que han sido emitidas durante grandes eventos como la Superbowl. El problema que tuvimos es que el proyecto en ese punto se desviaba de su rumbo original, ya que en Wikidata no había datos tan específicos para la gran mayoría de canciones y debíamos recurrir a otras fuentes.

El problema de otras bases de datos alternativas es que tampoco poseen ese tipo de información tan detallada, no siguen el modelo de datos RDF o tienen la información tan solo en casos muy concretos y sin una forma efectiva de acceder a ella a partir de nuestro dataset, como comprobamos al estudiar la posibilidad de utilizar **MusicBrainz** [6] para obtener las bandas sonoras en las que aparece una canción.

Otro aspecto en el que se puede trabajar para llegar a un trabajo mucho más completo y poder hacer una explicación de la relación de dos canciones, sería un estudio a nivel de usuario. En nuestro proyecto hemos hecho un análisis avanzado de las relaciones que tienen las entidades en un ámbito musical. Sin embargo, hay otro estudio posible que es el análisis de los usuarios que han escuchado esas canciones. Partimos de la base de que dos personas que escuchan una misma canción, pueden compartir parte de sus gustos musicales, y se pueden recomendar y crear relaciones conforme a su historial.

Cada usuario tiene un historial de temas escuchados, de hecho uno de los datasets que obtuvimos al principio del proyecto era de este carácter. Si se hiciese un estudio individual de un usuario, se podrían analizar sus gustos musicales, canciones más escuchadas, últimos géneros más populares, etc...

Como ejemplo podemos seleccionar un usuario cuya mayoría de temas escuchados pertenecen al género pop, pero también cuenta con algunos pertenecientes al género indie durante los últimos meses. Si se encuentra una tendencia similar entre una cantidad significativa de usuarios, podríamos sacar una explicación que establezca una relación entre un tema de género pop y otro de género indie por la frecuencia con que aparecen estos géneros juntos entre las listas de temas escuchados de estos usuarios mencionados.

Chapter 7

Introduction

Introduction to the subject area. This chapter contains the translation of Chapter 1.

Chapter 8

Conclusions and Future Work

Conclusions and future lines of work. This chapter contains the translation of Chapter 6.

Bibliografía

Y así, del mucho leer y del poco dormir, se le secó el celebro de manera que vino a perder el juicio.
(modificar en Cascaras\bibliografia.tex)

Miguel de Cervantes Saavedra

[1] T. Berners-Lee, R. Fielding, L. Masinter, et al. Uniform resource identifiers (uri): Generic syntax, 1998.

[2] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific american*, 284(5):34–43, 2001.

[3] T. Berners-Lee, L. Masinter, M. McCahill, et al. Uniform resource locators (url). 1994.

[4] W. contributors. Wikidata: Introduction, 2020. Accessed 13-April-2020.

[5] M. B. T. W. R. J. Felix Miller, Martin Stiksel. Last.fm. Accessed 02-June-2020.

[6] R. Kaye. Musicbrainz. Accessed 02-June-2020.

[7] P. Saint-Andre and J. Klensin. Uniform resource names (urns). *Internet Engineering Task Force (IETF), RFC*, 8141, 2017.

[8] S. Sakr, M. Wylot, R. Mutharaju, D. Le Phuoc, and I. Fundulaki. *Linked Data: Storing, Querying, and Reasoning*. Springer, 2018.

Apéndice A

Reparto de trabajo

La toma de decisiones en las distintas áreas del proyecto, así como la distribución de tareas, ha sido un esfuerzo consensuado entre los miembros del equipo. Para asegurar esta cooperación, ambos miembros han hecho una serie de reuniones informales (generalmente cada una o dos semanas) para poner en común el progreso individual realizado y acordar el curso a seguir a partir de ahí.

A.1. Adrián Garrido Sierra

En lo que respecta a la realización de esta memoria, Adrián se ha encargado de realizar las siguientes tareas:

- Formato y cohesión mediante el uso de LaTeX.
- Referencias bibliográficas.
- Revisión y colaboración en el Capítulo 1: Introducción.
- Revisión y colaboración en el Capítulo 2: Estado de la Cuestión.
- Capítulo 3: Tecnologías y Herramientas utilizadas.
- Aportación principal del Capítulo 4: Explicaciones.
- Capítulo 5: Diseño de la interfaz de explicaciones.
- Colaboración en el apartado 6.3 Arquitectura de la aplicación.
- Sección 7.1 Conclusiones

Pasando a la implementación, ha sido el responsable principal de las siguientes partes del trabajo:

- Montaje del framework y el servidor. Esto incluye el alojamiento en Heroku.
- Organización de la estructura de carpetas.
- Programación de las vistas, incluyendo HTML, CSS y JavaScript relacionado.

- Programación del controlador.
- Programación de funciones auxiliares para el algoritmo encargado de generar los grafos de explicaciones a partir de los datos obtenidos del modelo.

A.2. Diego Sánchez Muniesa

Los siguientes puntos de la memoria han sdo elaborados por parte de Diego Sanchez Muniesa:

- Formato y cohesión mediante el uso de LaTeX.
- Redacción del Capítulo 1 del proyecto: Introducción.
- Redacción del Capítulo 2: Estado de la Cuestión.
- Revisión y aportaciones del Capítulo 4.
- Aportación principal del Capítulo 4: Explicaciones.
- Capítulo 6: Redacción del Capítulo 6: Implementación.

En implementación, ha sido el responsable principal de las siguientes partes del trabajo:

- Creación del dataset limpio de canciones que se toma como source en la aplicación mediante su preprocesamiento, parseo y adaptación a los estándares de Wikidata.
- Estructura del sistema de unión y relación de los datos mediante un modelo linked data.
- Codificación y montaje de queries mediante la librería SPARQLWrapper.
- Estructura y codificación del modelo de la aplicación.
- Programación de scripts auxiliares a modo de pruebas.
- Diseño y elaboración de imágenes utilizadas en la interfaz de la aplicación.