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

Los sistemas de recomendación proporcionan a los usuarios opciones seleccionadas en base a sus preferencias, aportándoles una forma cómoda y rápida de encontrar elementos de interés. Para que este proceso sea efectivo, sin embargo, el usuario debe comprender las recomendaciones y confiar en el sistema para encontrar lo que busca. Esta confianza puede lograrse mediante el uso de explicaciones, que son una forma intuitiva de justificar la elección de las recomendaciones de forma comprensible para el usuario.

Este trabajo busca enriquecer las explicaciones para un recomendador de música gracias a los datos enlazados. Se hará un estudio de esta tecnología y las formas en que puede aplicarse al ámbito de los sistemas de recomendación y se demostrará de forma práctica con el desarrollo del prototipo de una aplicación que obtenga y muestre explicaciones para el recomendador mencionado.

Palabras clave

Datos enlazados, web semántica, explicaciones, RDF, SPARQL.

Abstract

Explanations based on Linked Data for a Music Recommender System

Recommender systems provide users with choices based on their preferences, providing a convenient and quick way to find items of interest. For this process to be effective, however, the user must be able to understand the recommendations and trust the system to find what they are looking for. This confidence can be achieved through the use of explanations, which are an intuitive way to justify the choice of recommendations in a way that is understandable to the user.

In this project we want to enrich the explanations for a music recommender using Linked Data. A study of this technology and the ways in which it can be applied to the scope of the recommendation systems will be made and demonstrated in a practical way with the development of the prototype of an application that obtains and shows explanations for the above-mentioned recommended.

Keywords

Linked Data, semantic web, explanations, RDF, SPARQL.

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.

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.

En este trabajo trataremos de emplear la tecnología de Linked Data para enriquecer un sistema de recomendación de música, proporcionándole explicaciones que justifiquen sus recomendaciones.

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 una canción al usuario en base a otra canción que haya escuchado, supuestamente porque 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. 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.

Las conexiones obtenidas serán explicaciones, que justifican la relación entre dos entidades establecida por el sistema.

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, con el objetivo de desarrollar más tarde una aplicación que ejemplifique las aplicaciones prácticas que puede tener esta Web Semántica.

- Determinar qué propiedades pueden llegar a ser más útiles para establecer explicaciones para la relación entre dos canciones proporcionadas, incluso si no son muy parecidas musicalmente pero comparten otros elementos interesantes. 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.

- Diseñar una interfaz visual de explicaciones, utilizando un desarrollo iterativo que nos permita obtener un resultado satisfactorio. Este desarrollo incluye la implementación del diseño en nuestra aplicación.

- Una vez terminado, extraer conclusiones sobre el trabajo realizado y establecer tareas a realizar en un desarrollo futuro.

1.3. Plan de trabajo

Tras haber explicado la motivación de este trabajo y los objetivos propuestos para el mismo, pasamos a detallar las etapas atravesadas en el proceso de desarrollo.

Inicialmente nos hemos familiarizado con la web semántica y Linked Data, así como con algunas de las tecnologías relacionadas. Este estudio se ha detallado en el Capítulo 2.

A continuación establecimos la lista de explicaciones que utilizaríamos en la versión final de la aplicación, la cual viene recogida y explicada en el Capítulo 3.

Posteriormente iniciamos el desarrollo de la aplicación, empezando con el diseño de la interfaz de explicaciones (documentado en el Capítulo 4) y continuando con la implementación, incluyendo su arquitectura y funcionalidad. Este proceso está explicado en el Capítulo 5.

Por último, dedicamos un tiempo a reflexionar sobre el trabajo realizado y el ámbito estudiado a lo largo del proyecto. Estas conclusiones están anotadas en el Capítulo 6, al igual que el trabajo futuro que deberíamos realizar si continuásemos con el desarrollo de la aplicación.

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 [10], 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.

Actualmente, un navegador puede ejecutar una búsqueda de una palabra en la web de forma que el resultado que obtendremos será producto del análisis de los caracteres de nuestra palabra, es decir, buscará una serie de documentos que contengan esa cadena de caracteres. Sin embargo, el concepto principal de la Web semántica es que todas estás búsquedas relacionen conceptos parecidos a lo que refiere esa palabra. Esa búsqueda de conceptos no tiene por qué coincidir con la sintaxis de la palabra buscada si no con sus diferentes significados. De esta forma no nos limitamos a una búsqueda simple, si no que muchos conceptos de distintos contextos son relacionados dotando de una mínima inteligencia a un sistema [12].

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 hiper-

vínculos, navegando así entre datos conectados [16].

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.

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. Normalmente cuando queremos referenciar una página web o un documento, tenemos un identificador único que simplemente nos proporciona un acceso al sitio web pero no nos da opción a saber nada más de el. El principal motor o sentido del concepto de linked data es poder acceder a un documento pero también poder dotarlo de un concepto o significado para después poder referenciar otros documentos con este mismo concepto.

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 Language". 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 como para organizar registros de una base de datos [13].

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 [16]. 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.

Cabe señalar que el sistema RDF es un modelo de representación, pero existen muchos formatos comúnmente usados que siguen la filosofía de este modelo: RDF/XML, RDFa, Turtle o N-triples. Todos ellos son eficaces en cuanto al manejo de datos enlazados.

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) [16, 9].

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 [11, 15, 16].

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.

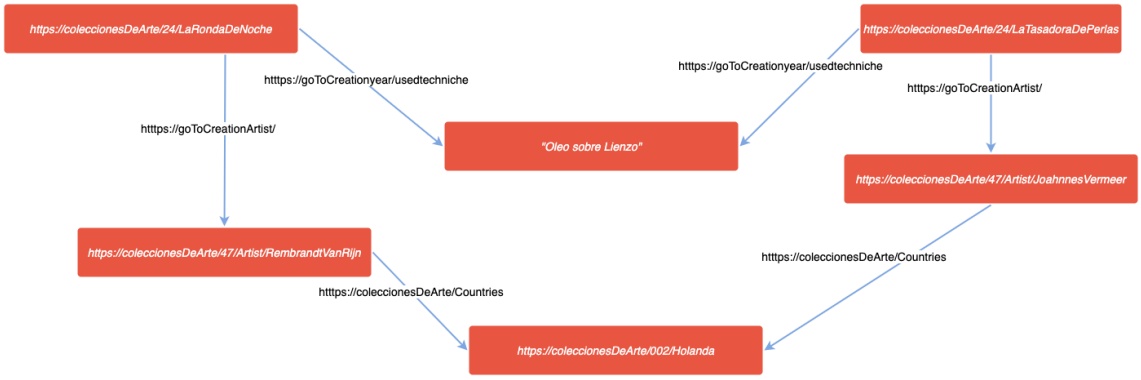

Figura 2.1: Ejemplo de grafo siguiendo el modelo RDF

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

2.2. Herramientas con uso de Linked Data

2.2.1. 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.

2.2.2. 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 [8]. 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

En el recomendador de música que tomamos como punto de partida para nuestro trabajo, el sistema recomienda una canción A y proporciona una explicación de esa recomendación en forma de una segunda canción B que ya le gusta al usuario y que es similar a la canción recomendada. Nuestro objetivo es enriquecer esas explicaciones para que el usuario tenga un mejor entendimiento de por qué se le recomienda la canción A a partir de la canción B, mostrando las características que tienen ambas canciones en común.

El uso de Linked Data nos permite aportar contexto a estas relaciones entre canciones. En nuestro caso consultaremos la web Wikidata para obtener las distintas propiedades de una canción y las compararemos con las propiedades de la canción con la que está relacionada, de tal forma que obtendremos una justificación fácil de entender de la similitud de ambas canciones.

De esta manera, llamaremos **explicación** a cada factor común que compartan ambas canciones, siendo principalmente propiedades que poseen las propias canciones. Estas propiedades son datos como, por ejemplo, el género al que pertenece la canción, su fecha de publicación o el artista que la interpreta. A lo largo de este capítulo concretaremos todas las explicaciones que vamos a estudiar, detallando cómo las obtenemos y lo que significa cada una.

Las relaciones que recogemos en cualquier caso de uso forman un grafo siguiendo el modelo RDF, ya que consideramos que una estructura de grafo permite una interpretación intuitiva de las explicaciones. Hemos decidido, además de seguir ese formato, dotarlas de un nivel que escala a medida que el estudio se aleje de la canción inicial. Esta distancia o alejamiento viene porque en el contexto del formato RDF y Linked Data la creación de relaciones se establece navegando por distintos objetos relacionados con el inicial secuencialmente. En ciertas explicaciones la propiedad que conecta las dos canciones no es una propiedad directa perteneciente a las canciones, sino que es la propiedad de los artistas que los interpretan, por ejemplo. La Figura 3.1 es útil para entender esta distinción.

La distinción de niveles nos permite identificar rápidamente las explicaciones más importantes. Cuanto menor sea el nivel, significará que está mas próximo a la canción inicial y por ello tendrá más peso a la hora de relacionarlas. Además, esta forma de separar las explicaciones hace que los grafos resultantes sean más ordenados y fáciles de entender.

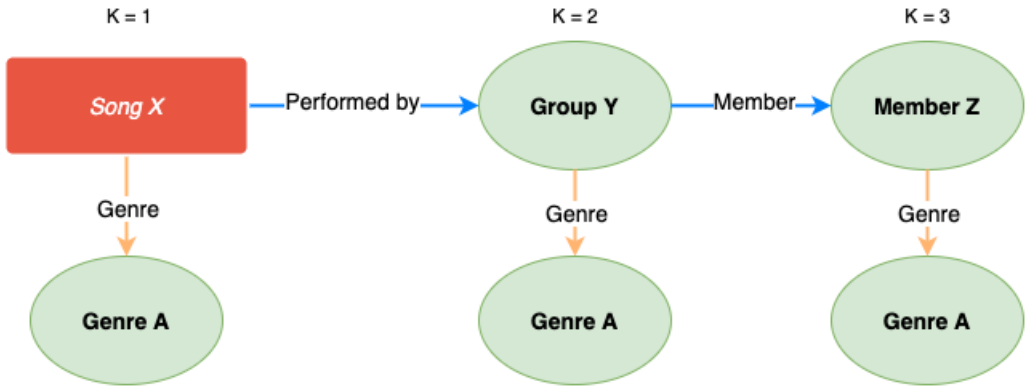

Figura 3.1: Ejemplo de misma relación en distintos niveles

En una primera fase, generaremos explicaciones directas para relacionar directamente las dos canciones u objetos principales, por ejemplo: género, artista, álbum, etc. Asignaremos un nivel 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 directas, 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 representar toda la información compartida, de forma que se pueda representar con un grafo. Estas explicaciones pueden tener un nivel 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.

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 compartirá el nivel $k = 2$ y usará 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 $k = 3$.

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. Por ello nos limitaremos a estudiar los 4 elementos mencionados: canción, artista, género y miembro (del artista).

A continuación vamos a recopilar las propiedades que hemos seleccionado como parte de nuestro estudio y cómo relacionarían dos canciones 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**). Para hacer más fácil de entender esta explicación y mantener una terminología coherente con el resto del proyecto, en las aristas aparecerá la palabra "artist" en lugar de "performer".

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".

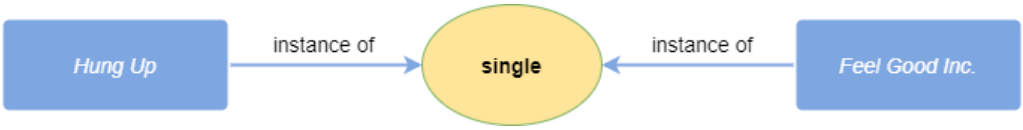

Figura 3.5: Ejemplo de explicación por single

3.2. Explicaciones del artista

Aquí se exponen las 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**.

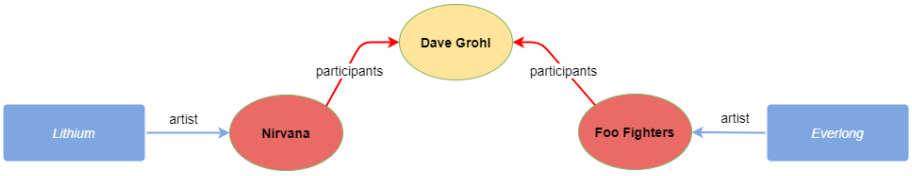

Figura 3.6: Ejemplo de explicación por integrantes

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*.

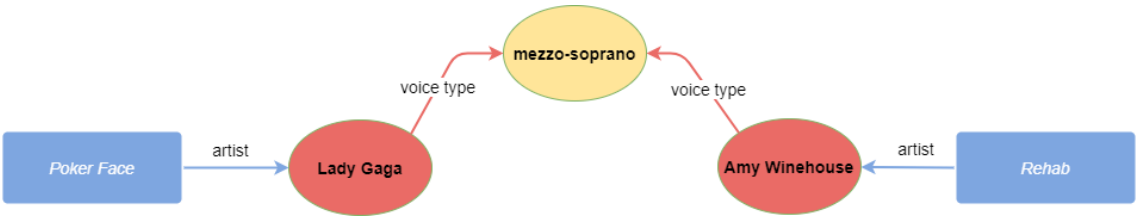

Figura 3.7: Ejemplo de explicación por tipo de voz

Fecha de fundación

Con esta propiedad intentamos tener en cuenta la historia de los artistas o su cronograma. Hay grupos que comenzaron su etapa musical conjuntamente a lo largo del tiempo. Un ejemplo de este caso se da 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.

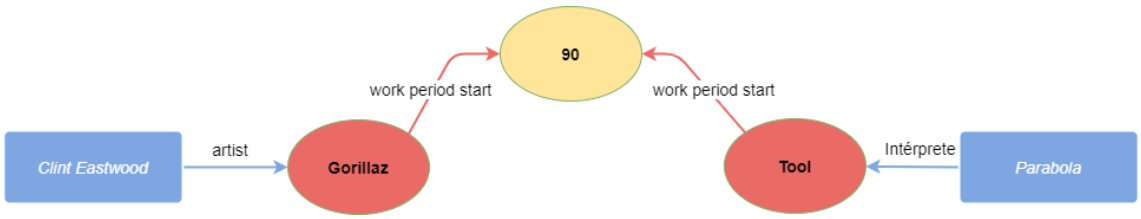

Figura 3.8: Ejemplo de explicación por fecha de fundación

3.3. Explicaciones del 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**.

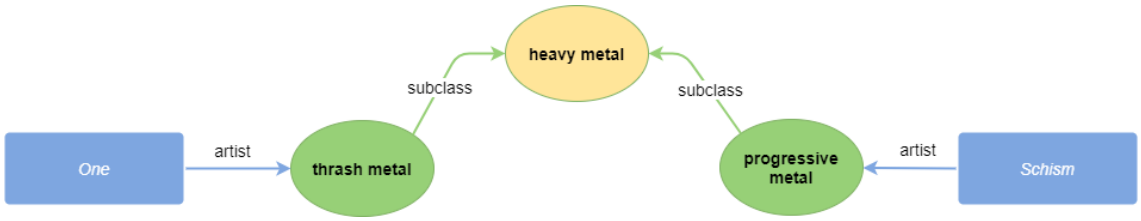

Figura 3.9: Ejemplo de explicación por Subgénero

3.4. Explicaciones 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 3.10: 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** y *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 3.11: 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 3.12: 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 3.13: 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.

3.5. Explicaciones 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**.

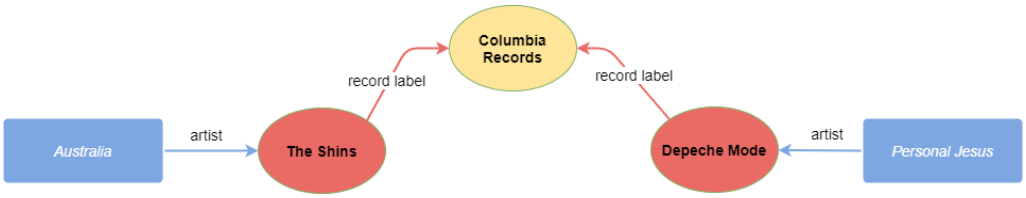

Figura 3.14: Ejemplo de explicación por compañía discográfica

3.6. Explicaciones 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.

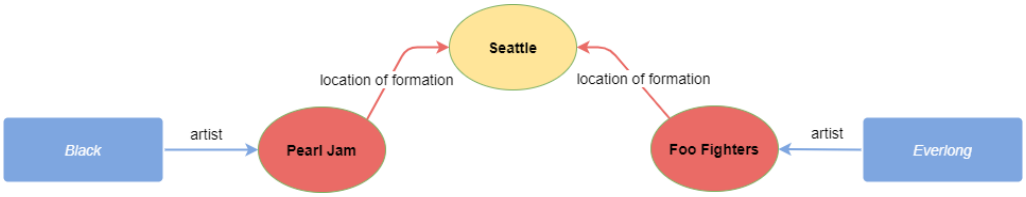

Figura 3.15: Ejemplo de explicación por lugar de formación

3.7. Explicaciones 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 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**.

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

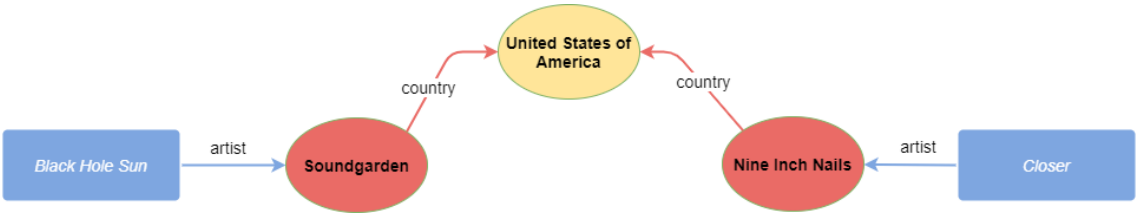

Figura 3.16: Ejemplo de explicación por país de origen

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**.

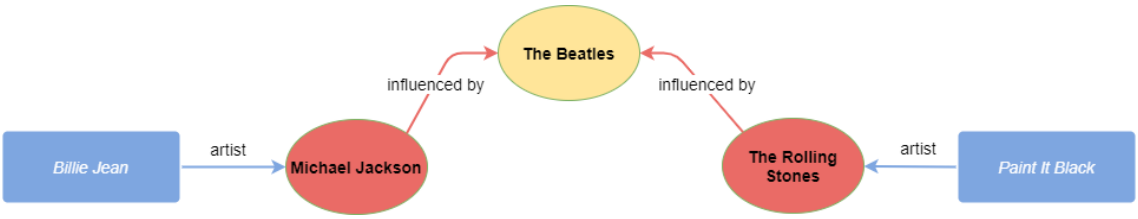

Figura 3.17: Ejemplo de explicación por influencia

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...

Una vez examinado el concepto de explicación y los distintos tipos que podremos encon-

trar en este proyecto, es el momento de pasar al desarrollo de la aplicación. En el siguiente capítulo se hará un repaso de todo el proceso de diseño de la interfaz, donde se utilizarán las propiedades y conceptos vistos en este capítulo para formar los grafos de explicaciones.

Capítulo 4

Diseño de la interfaz de explicaciones

El objetivo de nuestro proyecto, como ya hemos expresado anteriormente en este documento, consiste en enriquecer las explicaciones que justifican la relación existente entre dos canciones obtenidas de un recomendador de música, una de ellas es la recomendación para el usuario y la otra es la que se ofrece como explicación para justificar esa recomendación. Una vez realizado el estudio de las canciones y las distintas explicaciones posibles, es necesario mostrar el resultado de nuestro estudio de una forma comprensible para un usuario humano. Hay distintas maneras de lograr esto último, pero nosotros nos hemos decantado por una representación visual.

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.

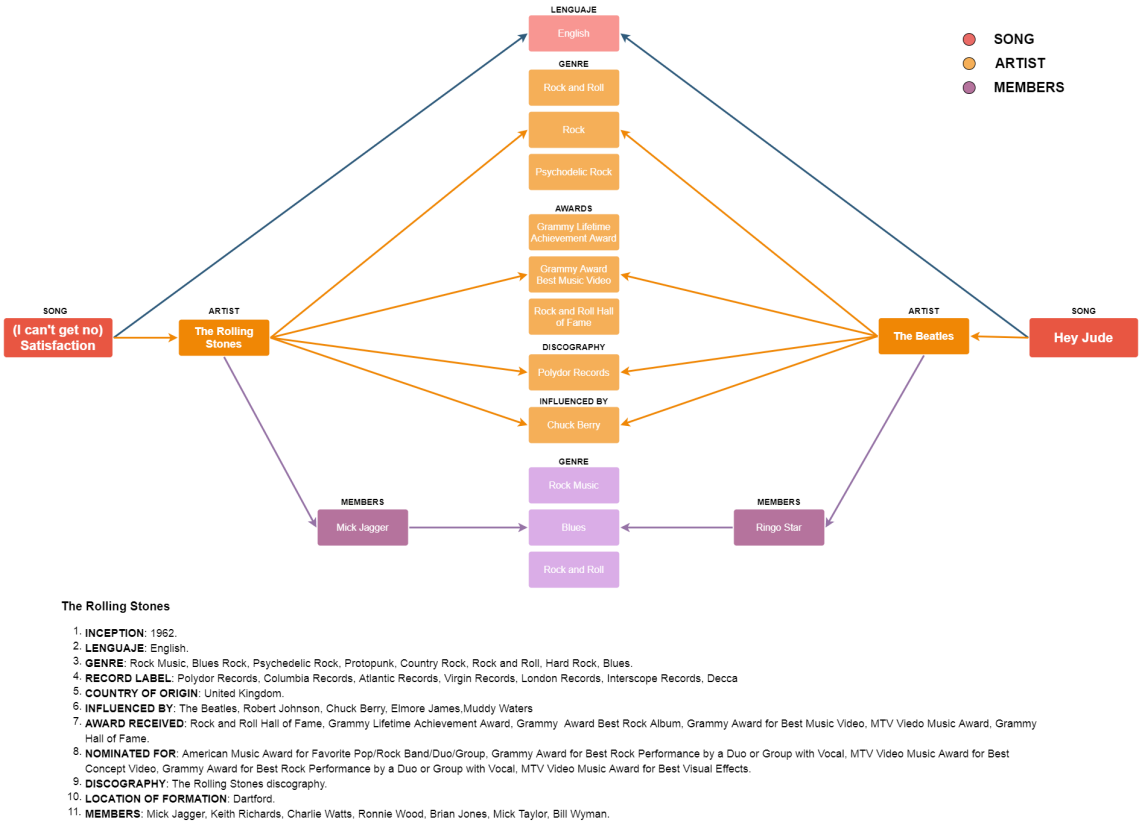

Figura 4.1: Primer diseño de la interfaz web

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 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 princi-

pales (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.

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. Para facilitar la tarea, 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

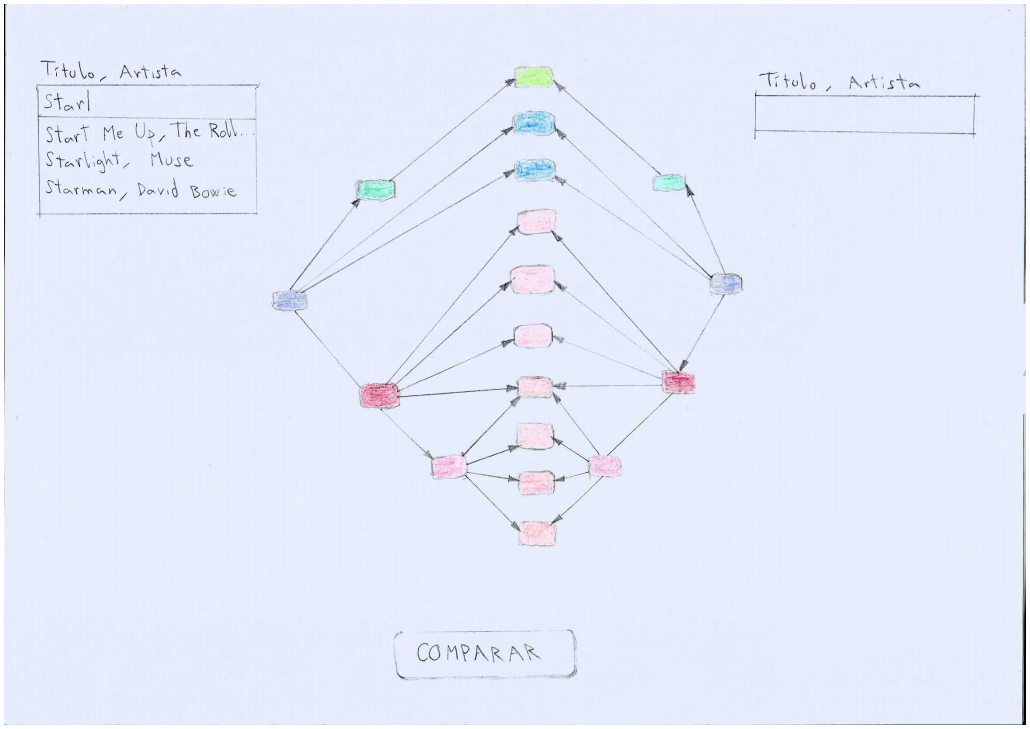

Figura 4.2: Segundo diseño de la interfaz web

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.

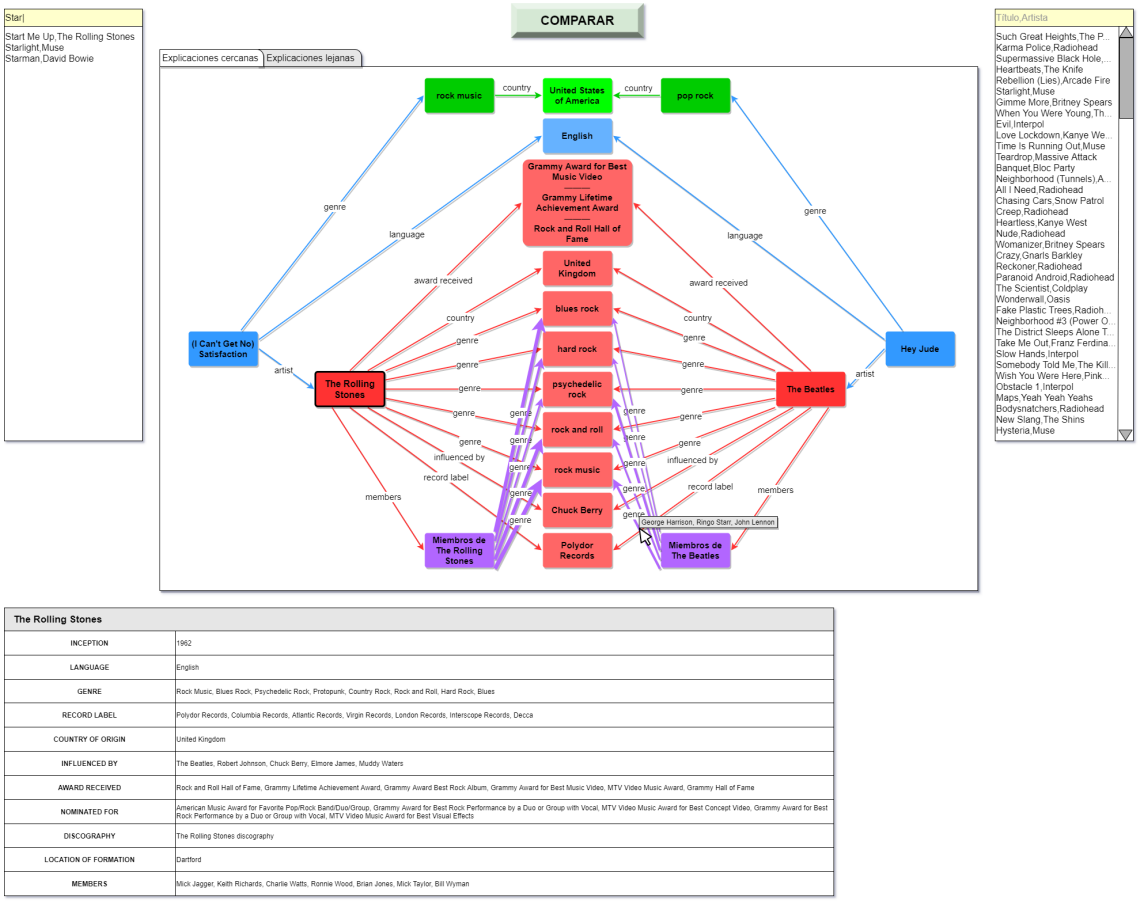

Figura 4.3: Tercer diseño de la interfaz web

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 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 de la misma categoría. Esto quiere decir que, por ejemplo, el género de una canción solo podrá relacionarse con un género de la otra canción, pues corresponden a la misma 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.

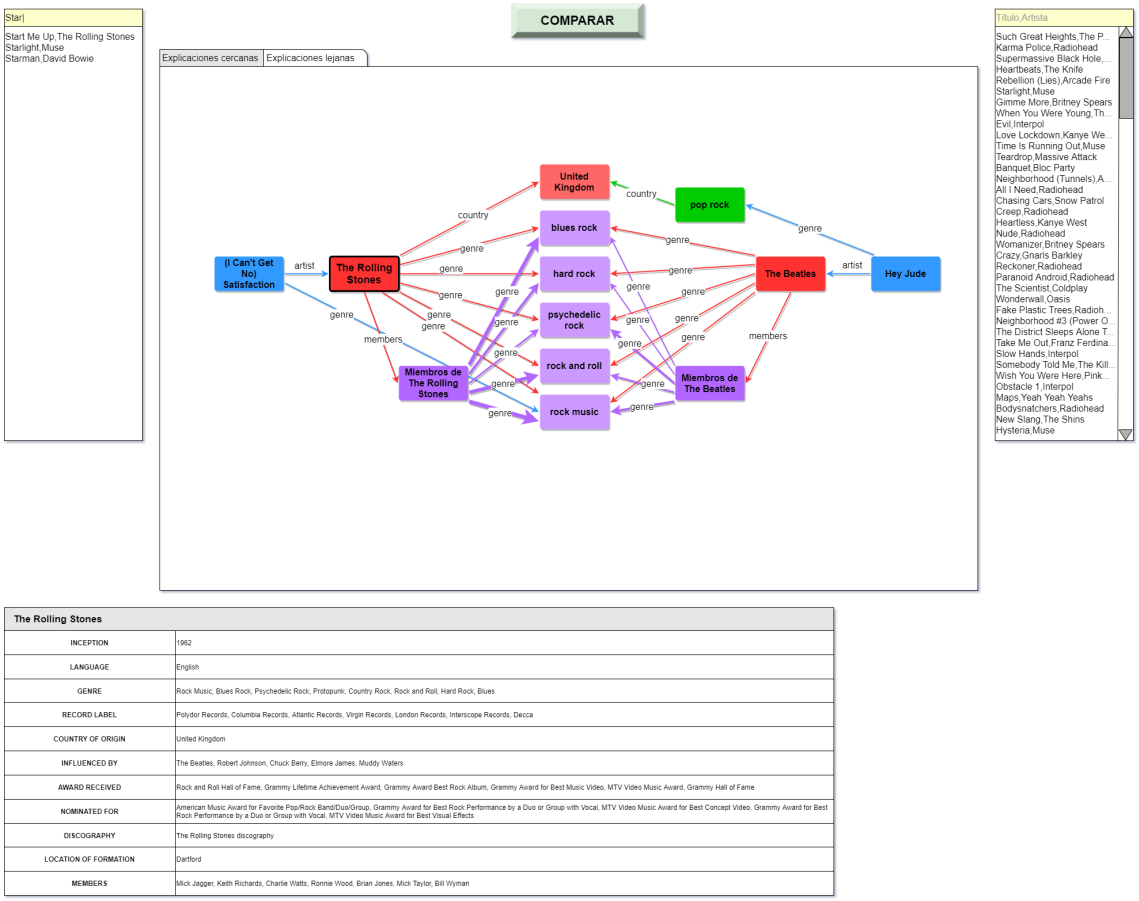

Figura 4.4: Grafo de explicaciones lejanas

El usuario podrá cambiar de vista libremente para consultar ambos grafos haciendo click en las pestañas anteriormente mencionadas. Todas las funciones explicadas en esta 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 4.5. 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,

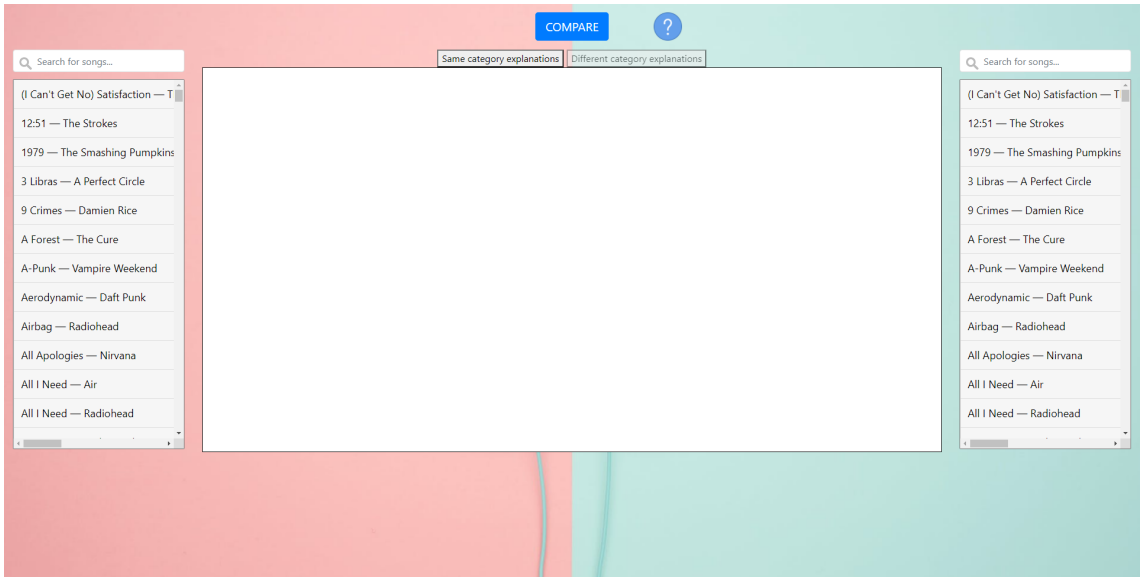

Figura 4.5: Estado inicial de la interfaz

hemos decidido cambiar el idioma de todos los elementos al inglés, pues nos parece un lenguaje más universal.

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.

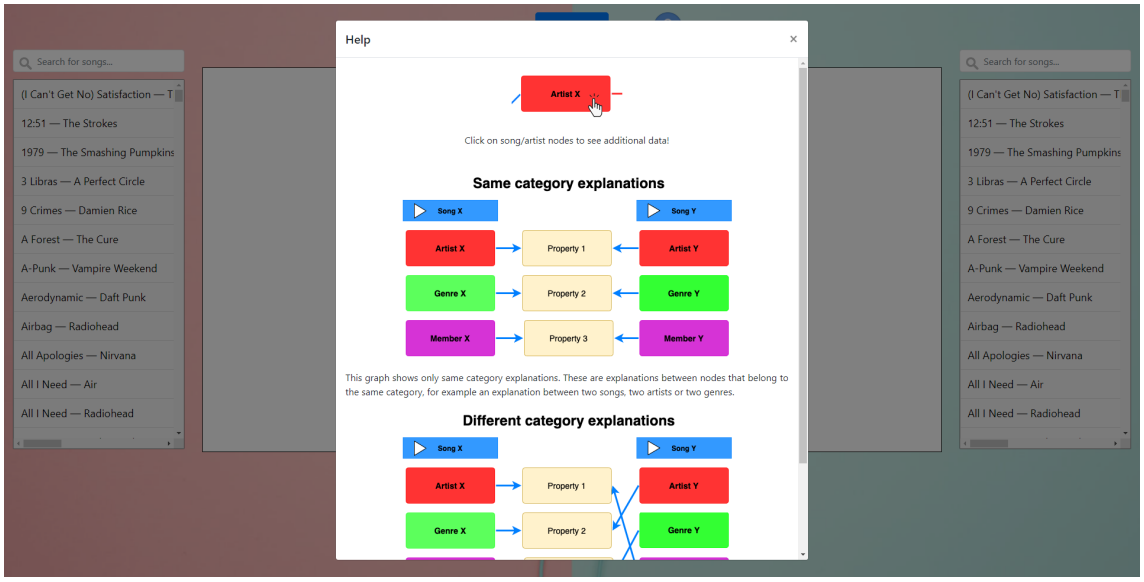

Figura 4.6: Ventana de ayuda

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.

Al pulsar ese botón, se despliega la ventana mostrada en la Figura 4.6. En ella viene explicada la diferencia entre los dos grafos con la ayuda de imágenes y texto. También viene un recordatorio sobre la función para obtener una tabla de datos adicionales al hacer click en los nodos correspondientes a canciones o artistas, la cual mostraremos más adelante.

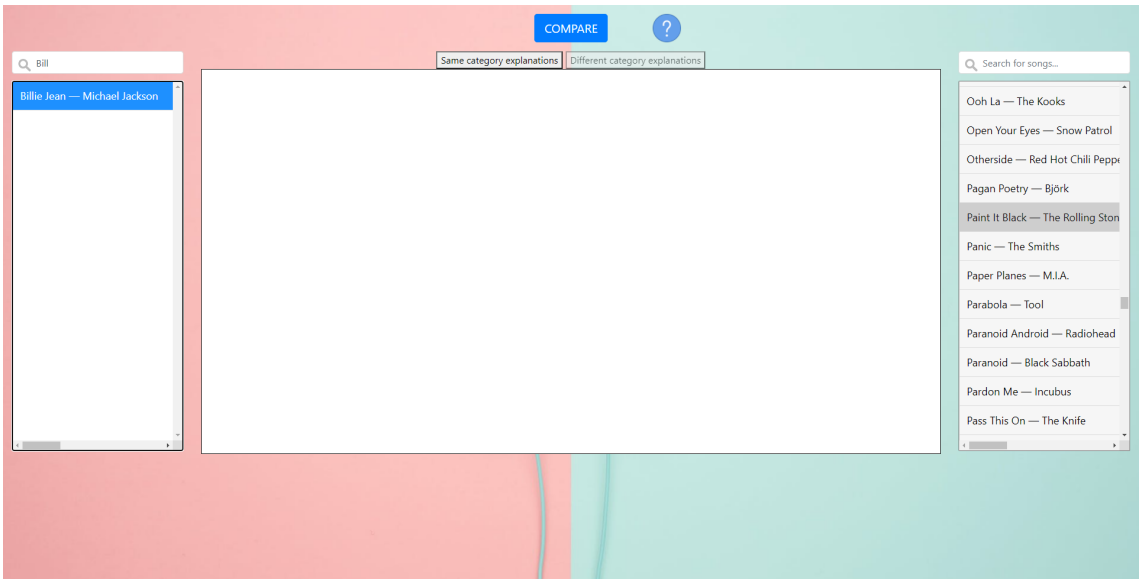

Figura 4.7: Ejemplo de selección de dos canciones

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 (Figura 4.7). 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, como se muestra en la Figura 4.8. Para esta y las figuras siguientes hemos seleccionado el ejemplo de "Influenciado por" utilizada en el Capítulo 3.7. Por defecto se muestra el "grafo de explicaciones de la misma categoría" (Same category explanations), aunque se puede alternar la vista entre ambos en cualquier momento. 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).

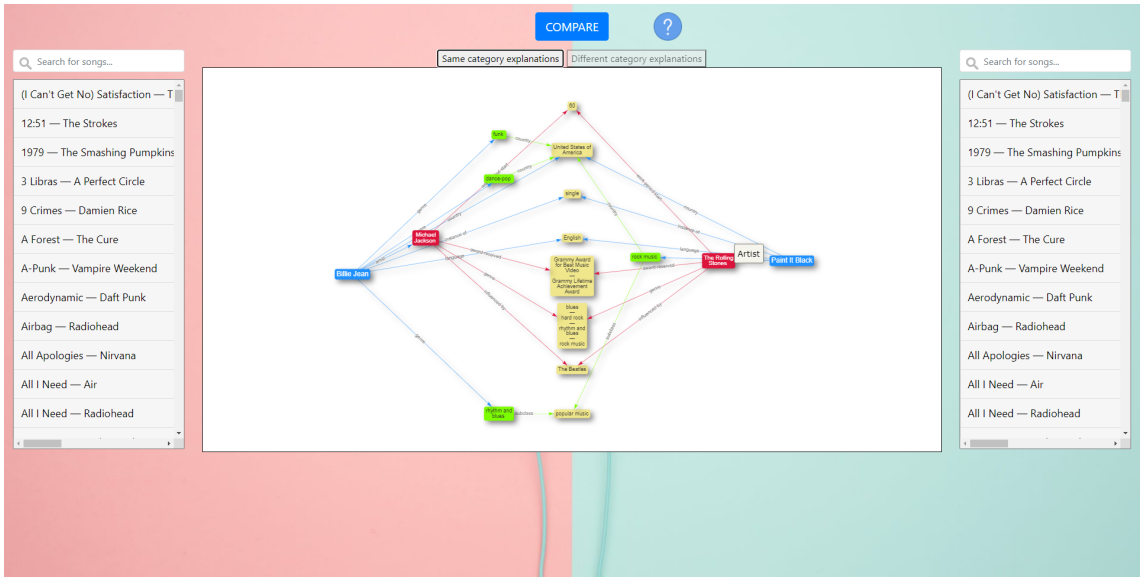

Figura 4.8: Grafo de explicaciones de la misma categoría

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.

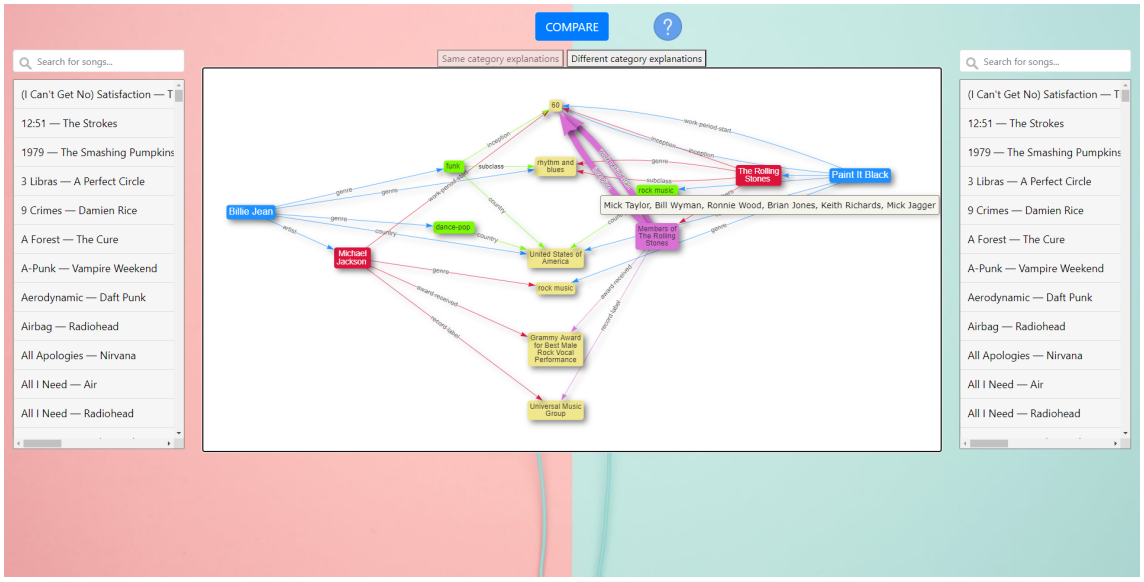

Figura 4.9: Grafo de explicaciones de distinta categoría

Al pulsar la otra pestaña, "Different category explanations", se muestra el grafo alternativo (Figura 4.9). 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.

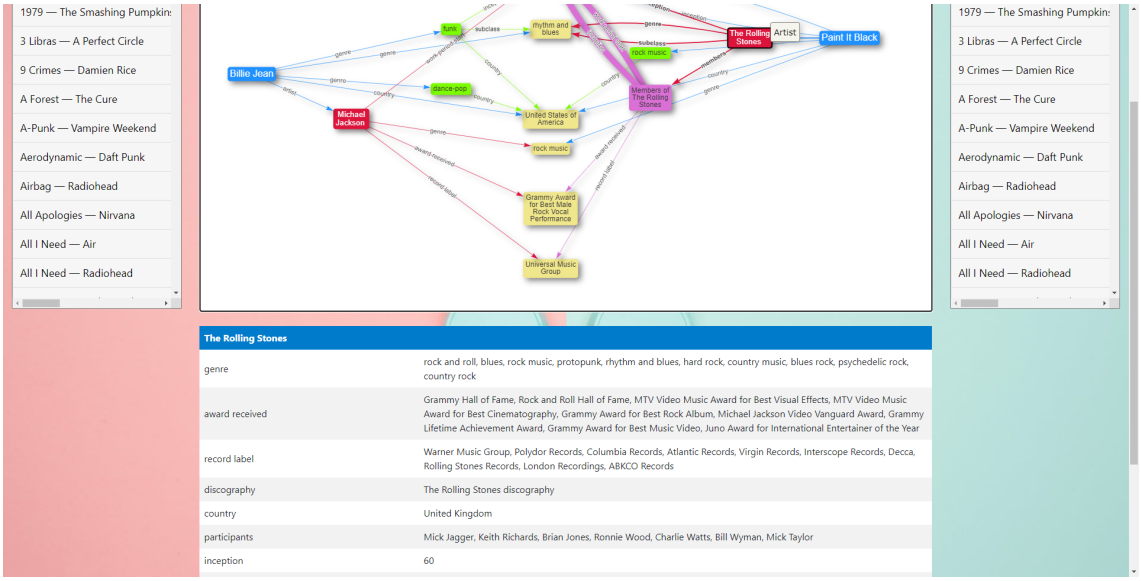

Figura 4.10: Tabla de datos adicionales

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 (Figura 4.10). 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.

Así concluye el desarrollo correspondiente con el diseño de la interfaz. En el siguiente capítulo pasaremos a examinar la implementación de la aplicación, por lo que trataremos aspectos más técnicos.

Capítulo 5

Implementación

Una de las partes más importantes de nuestro proyecto es el desarrollo del prototipo para una aplicación de explicaciones que haga uso de los diferentes conceptos que hemos estudiado en las fases iniciales del trabajo. En el capítulo anterior mostramos el proceso de diseño de su interfaz, así que ahora procederemos a detallar la parte de la implementación de la aplicación.

El código fuente de la aplicación está alojado en el siguiente repositorio de Github: `https://github.com/Adrigarr/TFG-2020-Code`. Sin embargo, para poder probar esta aplicación hay que usar este enlace: `https://explicaciones.herokuapp.com/`.

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** [2], 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. Arquitectura de la aplicación

Nuestra aplicación adopta una arquitectura cliente-servidor. En esta arquitectura, las tareas se reparten entre los proveedores de servicios, llamadas servidores, y los demandantes de esos servicios, llamados clientes. Por esto mismo, el servidor es el encargado de realizar todas las gestiones mientras que los clientes se limitan a hacer peticiones al servidor y recibir los resultados de esas gestiones [14].

Además, en nuestro caso el servidor no almacena todos los datos y recursos necesarios para el funcionamiento de la aplicación. Para poder gestionar apropiadamente las peticiones de los clientes, nuestro servidor debe ponerse en contacto con Wikidata, de donde obtendrá la información vital a procesar para obtener los resultados que se envían posteriormente al cliente que haya hecho la petición. En la Figura 5.1 se ve una representación de nuestra arquitectura.

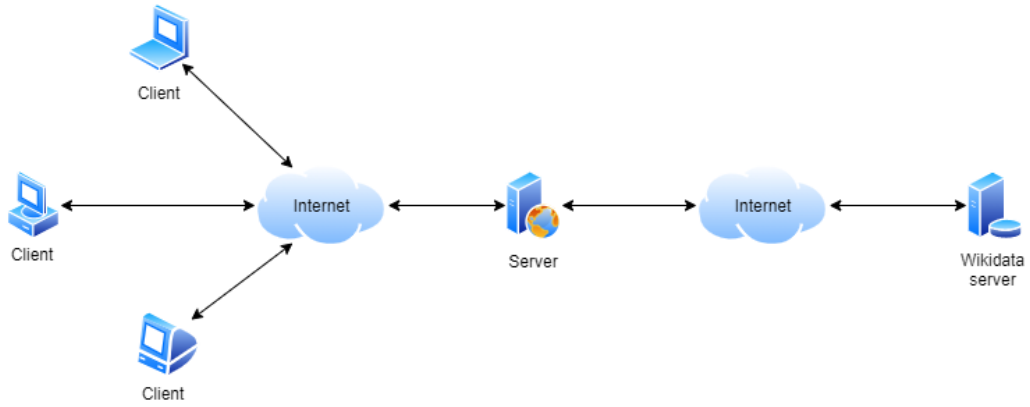

Figura 5.1: Diagrama de la arquitectura cliente-servidor de la aplicación

Para explicar mejor su funcionamiento, estudiaremos cada una de las partes de la arquitectura por separado.

5.2.1. Cliente

El cliente es la parte con la que interactúan los usuarios de la aplicación. Como la nuestra es una aplicación web, el cliente es la parte del navegador y la interfaz visual que se muestra en el mismo.

En la interfaz se muestra la lista de canciones a elegir, la cual recibe desde el servidor mediante un archivo de formato CSV. Una vez el usuario escoja dos de esas canciones y confirme la selección, el cliente enviará una petición al servidor. El servidor realizará una serie de operaciones para comparar las canciones y obtener los grafos de explicaciones deseados.

Cuando el servidor haya completado su parte, devolverá al cliente un archivo de formato JavaScript donde están contenidas las estructuras de datos necesarias para dibujar los grafos. Empleando la librería vis.js, el cliente dibujará dos grafos y los mostrará por pantalla para que el usuario pueda verlos.

5.2.2. Servidor

El servidor es la parte que escucha las peticiones del cliente y hace las gestiones necesarias para devolverle la información deseada. Esta es la parte responsable de la mayoría de procesos existentes en nuestra aplicación, desde gestionar consultas a Wikidata hasta tratar los datos obtenidos para dibujar los grafos de explicaciones.

Para explicar el funcionamiento del servidor debemos repasar los módulos que lo conforman, cuya relación se puede observar en la Figura 5.2.

El módulo “view” contiene las plantillas de la interfaz, es decir, lo que verá el usuario por pantalla una vez se haya renderizado en el navegador.

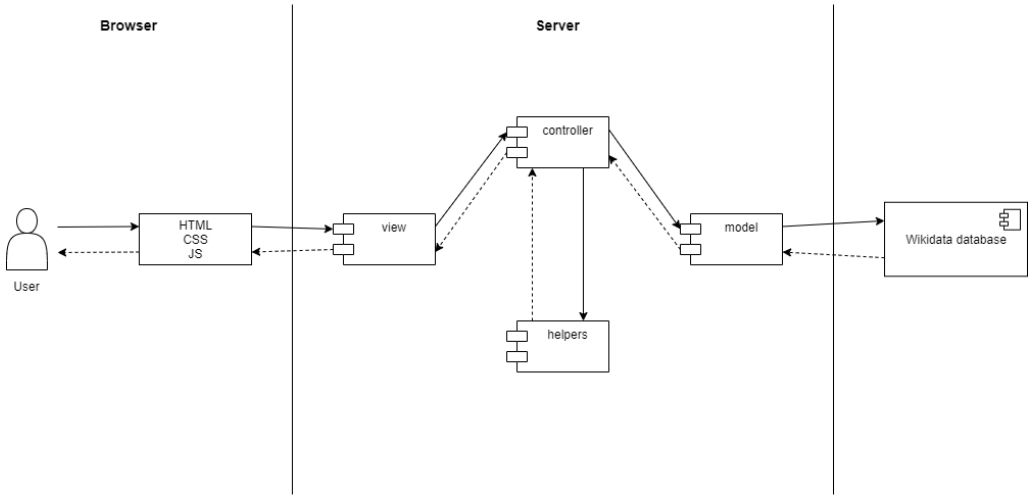

Figura 5.2: Diagrama de componentes de la aplicación

El módulo "controller" es el centro de control entre el resto de módulos. Todas las comunicaciones entre módulos pasan por aquí. Recibe las peticiones de "view", que se originan por las elecciones del usuario, y a su vez pide al módulo "model" que gestione esa petición y devuelva los resultados. Con los resultados de "model", "controller" le pide al módulo "helpers" que procese esos datos para poder dibujar los grafos, tras lo cual ya está listo para devolver los resultados a "view".

El módulo "model" es uno de los más importantes. Se encarga de organizar los datos con los que se trabaja en la aplicación en clases, además de comunicarse con el servidor de Wikidata para obtener la información en la que se basan las explicaciones del sistema. Hace las consultas SPARQL y recoge los resultados en estructuras de datos con las que pueda trabajar el resto de la aplicación.

Por último, el módulo "helpers" es un módulo auxiliar para el dibujo de los grafos. Recibe los datos obtenidos de las consultas y los procesa para generar un archivo JavaScript que contiene las instrucciones necesarias para generar los grafos. Este será el que utilice el cliente para mostrar los grafos de explicaciones al usuario.

5.2.3. Comunicación

La comunicación con el servidor de Wikidata, se hace mediante el módulo "model". Este módulo se encarga de establecer la conexión con el punto de consultas de Wikidata para poder iniciar las consultas y recoger los datos devueltos. Como punto principal, encontramos la librería que nos ha permitido todas estas funciones SPARQLWrapper. Esta librería hace las funciones principales de establecer queries, crear un canal de conexión, y recibir en formato JSON, XML o CSV los datos obtenidos a partir de las mismas.

La conexión será establecida cada vez que el usuario quiera hacer una comparación entre un par de canciones, es decir el servidor hace una llamada por cada caso de uso que se quiera probar.

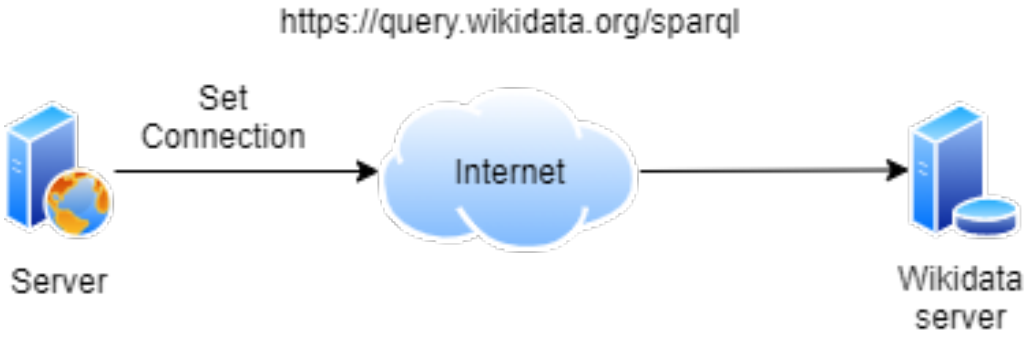

Figura 5.3: Diagrama para establecer una conexión

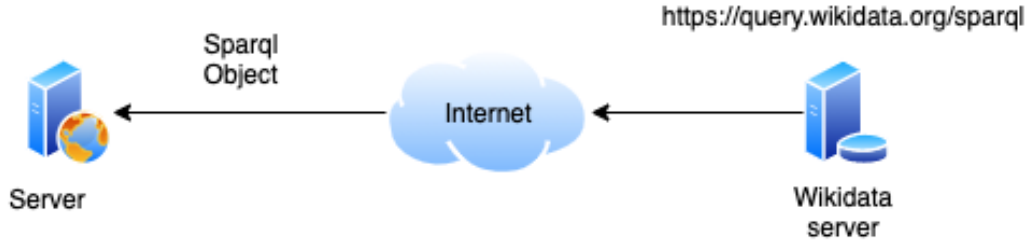

Figura 5.4: Respuesta del servidor de Wikidata

Una vez que obtengamos el objeto Sparql que devuelve el servidor de Wikidata ya podemos utilizarlo reiteradamente para poder mandar queries al servidor de Wikidata. Estas queries están predefinidas por las propiedades objeto de estudio y son las mismas para cada canción.

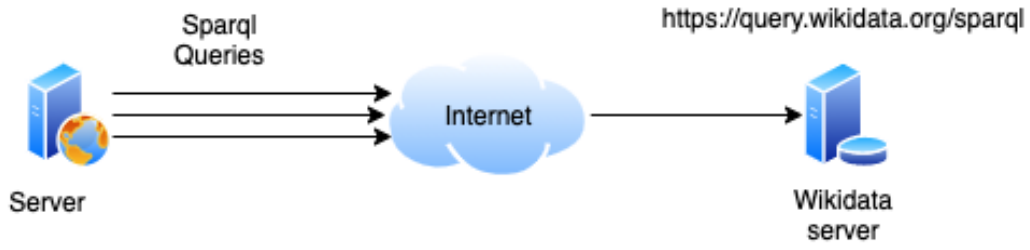

Figura 5.5: Lanzamiento de queries al servidor

La entrada común para todo caso de uso en nuestro modelo de la aplicación para un determinado caso de uso, siempre es la misma: El par de canciones y el conjunto de propiedades que se va a estudiar sobre ellas para poder hallar las explicaciones. Esta información está guardada en formatos JSON y CSV, los cuáles al inicio del proceso, serán convertidos en diccionarios para poderlos procesar con las clases del modelo en Python.

Contamos con una jerarquía de clases, las cuales completan el módulo de conexión. Se encargarán de procesar las repuestas del servidor y crear progresivamente a medida que nos llegan respuestas varios documento que contengan todas las propiedades de cada canción y las relacione entre sí para poder obtener un grafo representable.

5.3. 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.

5.4. Tecnologías y Herramientas utilizadas

Para la elaboración de este proyecto hemos tenido que estudiar numerosas tecnologías y herramientas diferentes. Algunas de ellas son viejas conocidas mientras que otras son completamente nuevas para nosotros y hemos tenido que familiarizarnos con ellas para poder trabajar con ellas.

A continuación haremos un repaso a las más relevantes para el desarrollo del proyecto, separándolas en distintos ámbitos para mayor comodidad.

5.4.1. 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 [5]. 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 [6]. 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 [7]. 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.

5.4.2. 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.

5.4.3. 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.

5.4.4. 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* [4]. 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 [1]. 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

En este capítulo exponemos nuestras reflexiones respecto al trabajo realizado a lo largo del proyecto y los siguientes pasos a tomar en caso de continuarlo.

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 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 la oferta de bases de conocimiento que aprovechen las ventajas

de la Web Semántica.

Centrándonos más en nuestro trabajo, consideramos que hemos logrado alcanzar la mayoría de nuestros objetivos propuestos. Gracias al estudio realizado de la Web Semántica y Linked Data, además de otras tecnologías, hemos alcanzado las conclusiones expuestas anteriormente. También hemos desarrollado un prototipo funcional de aplicación para mostrar el resultado de aplicar esas técnicas a las explicaciones de un recomendador. Todavía se podría continuar trabajando para mejorar y completar nuestro proyecto, pero estamos satisfechos con el resultado conseguido.

6.2. Trabajo futuro

Teniendo en cuenta que uno de nuestros objetivos es justificar de forma comprensible las recomendaciones de un sistema de recomendación, la evaluación con usuarios sería muy útil para determinar si se ha logrado alcanzar esta meta. No nos ha sido posible realizarla en esta versión del trabajo por falta de tiempo y complicaciones relacionadas con la situación extraordinaria de este año, pero sería el primer paso para mejorar nuestro proyecto.

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 establecer 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** [3] 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

Since the development of the Internet as a global network, it has always hosted large amounts of documents and written information. As it grew, different ways had to be found to organize that information so that it could be used as a service to any user. The Semantic Web, whose characteristics we will explain later, is one of the methods that the entire Internet was able to adapt when it came to giving a meaning and a utility to that information.

The Semantic Web tries to link all that information in order to give it context and accessibility. To do this, it adopts the method of Linked Data, which, referring to its own name, connects or links different data establishing a relationship. The main objective of this is to give computers the ability to navigate and access most of the data stored on the Internet so that they are useful and not lost in a sea of information.

It is through this concept that search engines can be helped by this technology, as we provide machines the ability to navigate over those relationships and understand data that we require or could be useful to us.

In this project we will try to use Linked Data technology to enrich a music recommender system, providing explanations that justify its recommendations.

7.1. Motivation

The goal of this project is to give explanations to a music recommender. We start from a music recommender that provides a song to the user based on another song he has listened to, allegedly because there is some kind of relationship between them. Using that premise, we want to explain **why** those two songs are related or, in other words, how they resemble each other. To achieve this goal we will use the semantic web.

As we mentioned, the Semantic Web is a way to improve the connection between the data stored in the net, making the Internet more useful and accessible to people. The problem is that it is a practice that is not widespread enough to represent a real advance in our way of relating to the network yet, because it requires making changes in the way information is stored. In this paper we will try to show the potential of the Semantic Web.

The main point is to be able to collect all the connections that appear between all the information that we can get from each of the two musical themes or songs. For this, we must study that entity and all related: artist who performs the song, its musical genre, the members of its band, its record label, etc. and in turn the relations between them, until we have enough connections that can guarantee that those two songs have some relationship.

The connections obtained are explanations, which justify the relationship established by the system between two entities.

7.2. Objectives

The main objectives of this project are:

- To make a study on the Semantic Web and the data models that allow it, with the aim of developing later an application that exemplifies the practical applications that this Semantic Web can have.

- To determine which properties may be most useful to establish explanations for the relationship between two provided songs, even if they are not very similar musically but share other interesting elements. This is the premise of our project and, thanks to our research, we will be able to determine which properties are most important when obtaining meaningful explanations in the domain.

- To study different technologies that help us in the elaboration of the aforementioned application, in addition to expanding our knowledge in areas of computer science that we have not necessarily explored during the degree.

- To design a visual interface of explanations, using iterative development that allows us to obtain a satisfactory result. This development includes the implementation of the design in our application.

- Once completed, draw conclusions about the work done and set tasks to be performed in a future development.

7.3. Work plan

After explaining the motivation of this project and the objectives proposed for it, we proceed to detail the stages crossed in the development process.

Initially we became familiar with the Semantic Web and Linked Data, as well as some of the related technologies. This research has been detailed in Chapter 2.

Then we established the list of explanations that we would use in the final version of the application, which is collected and explained in Chapter 3.

Later we started the development of the application, starting with the design of the explanation interface (documented in Chapter 4) and continuing with the implementation, including its architecture and functionality. This process is explained in the Chapter 5.

Finally, we reflected on the work done and the subject studied throughout the project. These conclusions are noted in Chapter 8, as well as the future work we should do if we continued to develop the application.

Chapter 8

Conclusions and Future Work

In this chapter we expose our reflections on the work done throughout the project and the next steps to take if we were to continue working on it.

8.1. Conclusions

After working with the Semantic Web and Linked Data, we have been able to verify that they are methods of data management with lots of potential that we could all benefit from if we explore beyond its current state. Its main drawback is perhaps precisely that its usefulness depends to a large extent on how widespread its use is and, unfortunately, it is currently not as standardised as we would like it to be.

Taking as an example the domain on which we have focused our project we can see these limitations. Thanks to Linked Data and the RDF format we have been able to obtain a good amount of information that allowed us to generate explanations for the relationship between many different songs, however this work is not enough to apply the same treatment to all existing musical works, not even on the platform we have chosen for data collection.

We chose the Wikidata website to study the songs because of its size and because it uses the linked data model effectively. However, this platform is far from perfect because it does not host the same amount of information from all its elements even though that information exists, so there are songs that we have been able to explore in more depth than others. On the other hand there are also cases of unreliable or outright incorrect information, a problem that we have encountered on occasion and which causes that our application cannot be used correctly in such cases.

Despite the problems mentioned, Wikidata is still the best option we have found for hosting music information. Other platforms do not use Linked Data or do so in such a way that it is very difficult to establish relationships between its elements because they are excessively concrete, to the point of having many different entries for elements that are identical in essence. Projects like ours would benefit from a greater variety in the supply of knowledge bases that take advantage of the advantages of the Semantic Web.

Focusing more on our project, we believe that we have achieved most of our set objectives. Thanks to our research in the Semantic Web and Linked Data, in addition to other technologies, we have reached the conclusions mentioned above. We have also developed a functional application prototype to show the result of applying those techniques to a recommender's explanations. We could still continue working to improve and complete our project, but we are satisfied with the result achieved.

8.2. Future work

Given that one of our objectives is to provide an understandable justification for the recommendations of a recommender system, user evaluation would be very useful in determining whether this goal has been achieved or not. We have not been able to do it in this version of the project due to lack of time and complications related to the extraordinary situation of this year, but it would be the first step to further improve our project.

The RDF scheme and the use of SPARQL for its use and analysis can be very powerful and encompass large amounts of data. In our case we have focused on working with the Wikidata tool and data for a matter of amount of information, reliability and results obtained. With Wikidata we have been able to establish connections between songs with as much information as possible on each of their respective descriptive pages. However there are more options to gather from more information about songs and artists.

We originally tried to get information that was a little bit further away from what we created, which is a musical analysis. There are other options and data that can be taken from songs, such as movies or series in which they appear as soundtrack, or also songs that have been broadcast during major events such as the Superbowl. The problem we had was that the project at that point deviated from its original direction, because on Wikidata there was no data so specific for the vast majority of songs and we had to resort to other sources.

The problem with other alternative databases is that they do not have such detailed information either, do not follow the RDF data model or have the information only in very specific cases and without an effective way to access it from our dataset, as we found when studying the possibility of using **MusicBrainz** [3] to obtain the soundtracks in which a song appears.

Another aspect in which we can work to get a more complete work and be able to get an explanation of the relationship of two songs would be a user-level study. In our project we have made an advanced analysis of the relationships that the entities have in a musical field. However, there is another possible study that is the analysis of users who have listened to those songs. We start from the basis that two people who listen to the same song can share part of their musical preferences, and we can recommend and create relationships according to their history.

Each user has a listened tracks history, in fact one of the datasets we got at the beginning of the project was of this type. If we were to do an individual study of a user, we could analyze their musical tastes, most popular songs, latest popular genres, etc.

As an example we can select a user whose most listened songs belong to the pop genre, but also has some belonging to the indie genre during the last months. If a similar trend is found among a significant number of users, we could draw an explanation that establishes a relationship between a pop track and another indie track because of the frequency in which these genres appear together among the lists of listened songs from these users mentioned.

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] Alchemy.js. Accessed 11-September-2020.

[2] Last.fm. Accessed 02-June-2020.

[3] Musicbrainz. Accessed 02-June-2020.

[4] RDFstarTools. Accessed 11-September-2020.

[5] Sanic. Accessed 11-September-2020.

[6] SPARQLWrapper. Accessed 11-September-2020.

[7] vis.js. Accessed 11-September-2020.

[8] Wikidata: Introduction. Accessed 13-April-2020.

[9] T. Berners-Lee, R. Fielding, and L. Masinter. Rfc 2396: Uniform resource identifiers (uri): Generic syntax, august 1998. *Status: Draft Standard*, 2007.

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

[11] T. Berners-Lee, L. Masinter, and M. McCahill. Uniform resource locators (url)(rfc 1738). *Network Working Group*, 1994.

[12] L. Codina and C. Rovira. La web semántica. In *Tendencias en documentación digital*. Trea, 2006.

[13] T. Heath and C. Bizer. Linked data: Evolving the web into a global data space. *Synthesis lectures on the semantic web: theory and technology*, 1(1):1–136, 2011.

[14] G. Reese. *Database Programming with JDBC and JAVA*. .ºReilly Media, Inc.", 2000.

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

[16] 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.