Sistema de recomendación de música

Lothar Soto Palma Daniel López García Ignacio Cordón Castillo

24 de marzo de 2017

Índice

1. Descripción

Nuestro sistema pide al usuario un determinado número de grupos afines (grupos 'modelo'). A partir de dichos grupos obtenemos una predicción, basándonos en grupos afines obtenidos a partir de las APIs de Youtube y de Spotify. Para los grupos sugeridos además, proporciona información biográfica, álbumes de dichos grupos, etc.

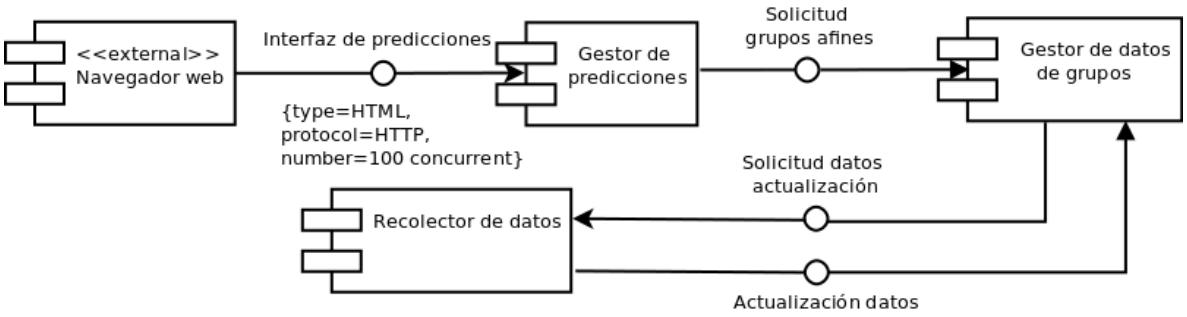
2. Vistas

2.1. Punto de vista funcional

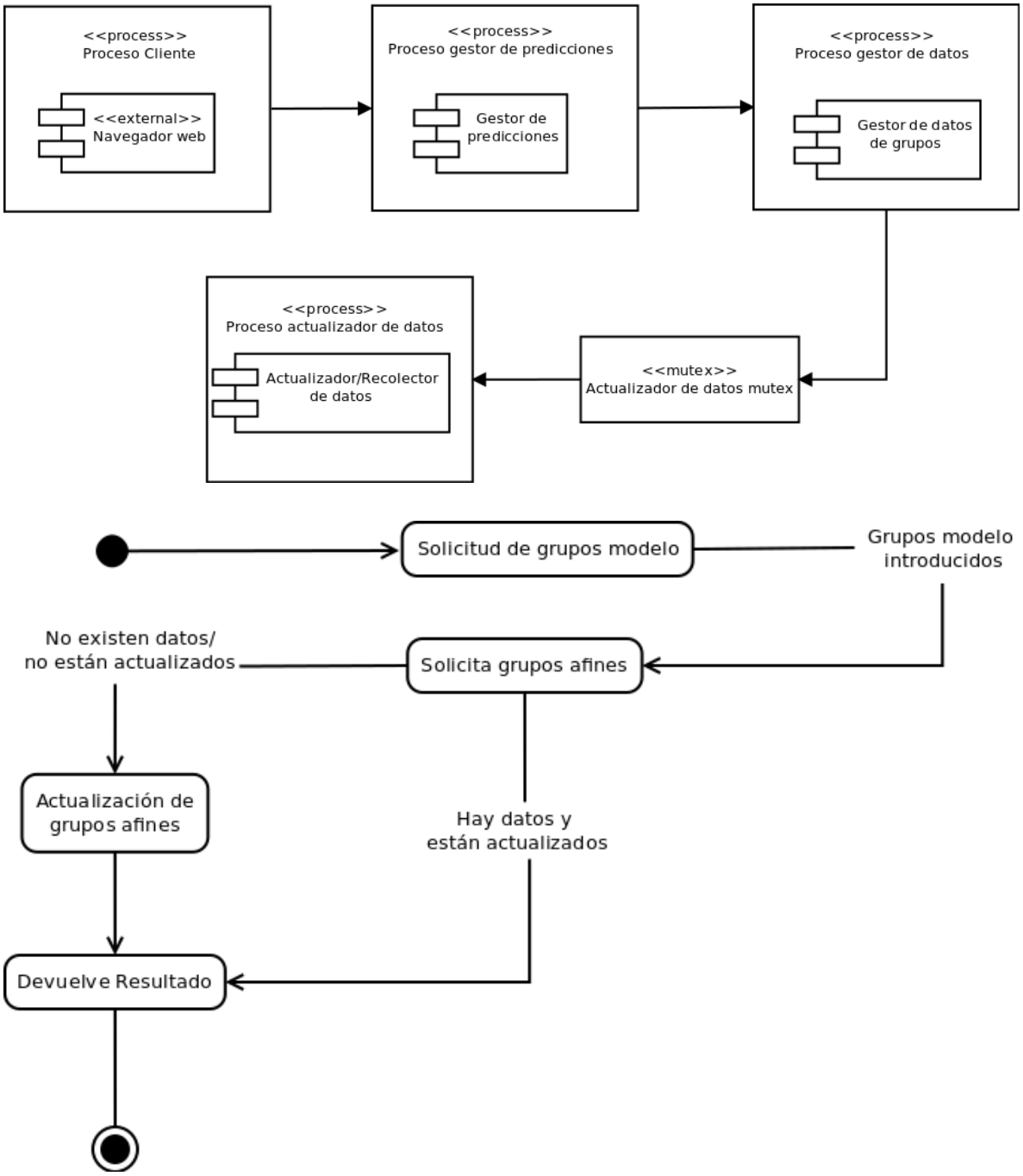
En nuestro punto de vista funcional reseñamos los siguientes elementos:

- **Navegador web**: elemento funcional externo, que se ejecuta del lado del cliente. Accede a través de la interfaz de predicciones, mediante peticiones HTTP al *gestor de predicciones*.
- **Gestor de predicciones**: se ha pensado en este módulo como un elemento que sintentiza todos los grupos afines sugeridos para unos grupos modelo de entrada, escoge cuáles mostrarle al usuario y devuelve la respuesta al navegador web. Solicita al *gestor de datos* la información disponible para grupos afines a los introducidos por el usuario en el navegador web.
- **Gestor de datos de grupos**: se encarga de acceder a los datos de grupos, y en caso de no existir o estar desactualizados, llama al *recolector de datos*. Caso opuesto devuelve la información al *gestor de predicciones*.
- **Recolector de datos**: se encargar de acceder a los datos a través de las APIs o el *scrapeo* y los devuelve al gestor de datos de grupos para la actualización en la base de datos.

En nuestro caso se ha decidido separar en un único módulo funcional el acceso a la base de datos (escritura y lectura de información de grupos afines).



2.2. Punto de vista de concurrencia



2.3. Punto de vista de despliegue

En esta vista se describe el entorno en el que el sistema será desplegado, incluyendo cada una de las dependencias del mismo. Para este apartado es necesario considerar los siguientes puntos:

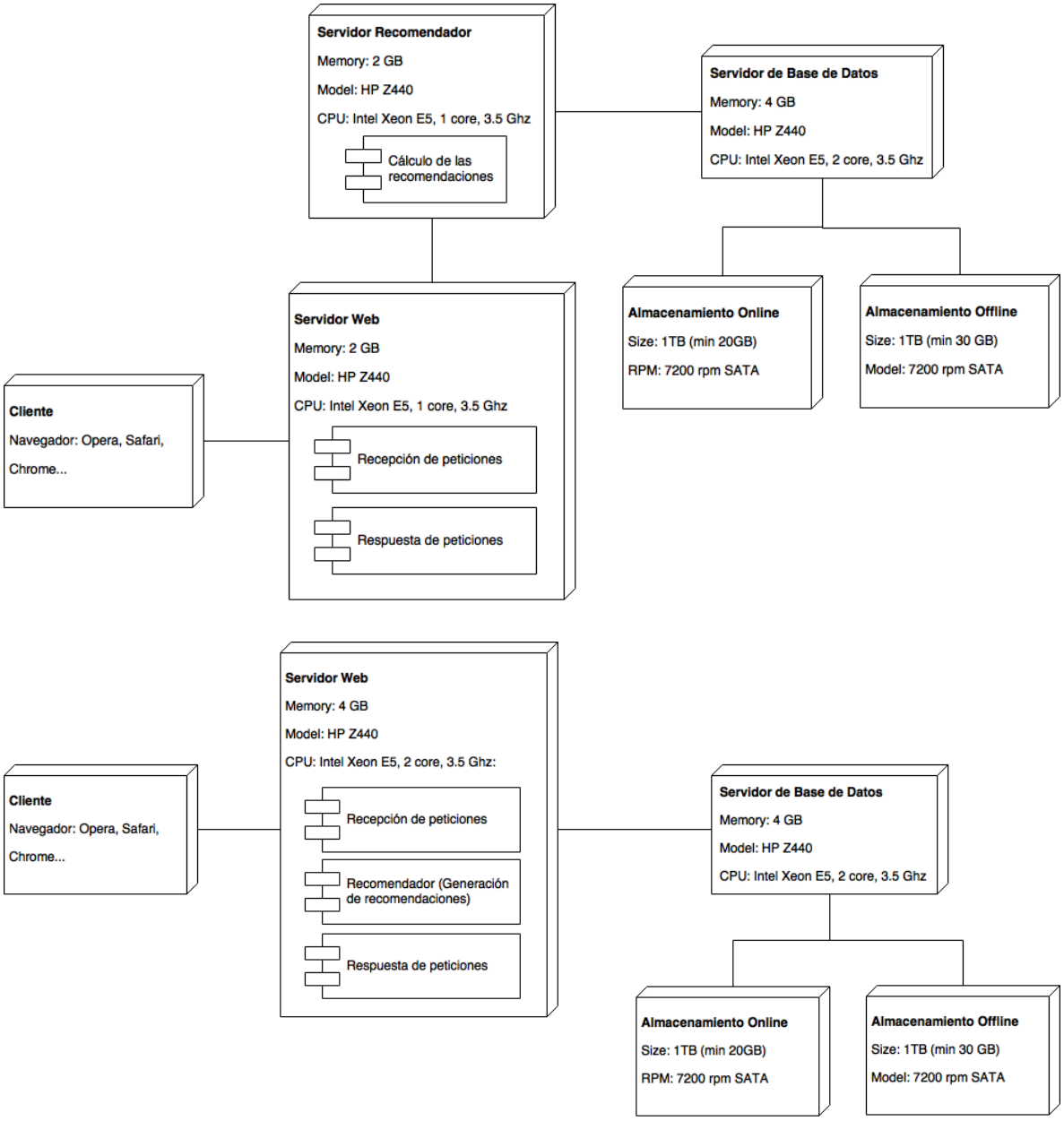
- Tipo de Hardware requerido.
- Especificaciones y precios.
- Software de terceros utilizado en el sistema.
- Compatibilidad de las tecnologías utilizadas en el sistema.
- Requisitos y capacidad de la red.

Se han considerado los siguientes nodos para el diagrama de despliegue:

- Nodos de procesamiento principales:
  - **Servidor recomendador:** Parte del sistema encargado de procesar a partir de la información de la base de datos y la proporcionada por la interacción del usuario final a través del cliente.
  - **Servidor de base de datos:** Parte del sistema encargado de almacenar los datos de la música obtenida a través de software de terceros, en particular +2 APIs y al menos un scrapping. Es necesario la extracción de datos e integración de los datos.
  - **Servidor Web:** Parte del sistema encargado de la recepción y respuesta de las peticiones generadas por el usuario final a través del cliente. El servidor web procesa la petición del usuario y la envía al sistema recomendador con la finalidad de generar una respuesta.
- Nodos de cliente:
  - **Navegador:** El acceso al sistema por parte del usuario final se realiza a través de cualquier navegador web (p.e.: Opera, Edge, Safari, Chrome, Firefox,...)
- Nodos de almacenamiento:
  - **Almacenamiento online:** Parte del sistema que se encarga de almacenar la información principal generada por el sistema.
  - **Almacenamiento offline:** Parte del sistema que se encarga de almacenar las copias de seguridad del sistema, para evitar pérdida de funcionalidad a la hora de una posible mala migración de datos o ataques al servidor.
- Conexión de red.

Si nos ponemos a pensar en los principales cuellos de botella de una aplicación web, estos son claramente los servidores y su actividad, por lo que es muy posible que el servidor que se encarga de la recomendación y el servidor web se integren en uno solo, ya que la interacción entre ellos debe ser rápida.

En el primer diagrama se considera el servidor recomendador de manera independiente del servidor web y en el segundo caso el servidor web tiene un módulo que realiza las tareas del recomendación, sin embargo estas agrupaciones pueden llegar a ser lógicas y los tres servidores pueden actuar en una única máquina en la que a cada servidor se le asigna una cantidad determinada de recursos físicos.



Esta interpretación del sistema desde el punto de vista del despliegue se realiza sin tener en cuenta el servicio Google App Engine que se usará en la práctica para trabajar, puesto que el servicio sustituirá esta parte del diseño.

2.4. Punto de vista operacional

En esta vista se describe como el sistema funciona, será administrado y mantenido cuando el sistema está funcionando en su entorno de producción. Es necesario considerar los siguientes puntos:

- Instalación y actualización.
- Migración de funcionalidades y datos.
- Backup y restauración del sistema.

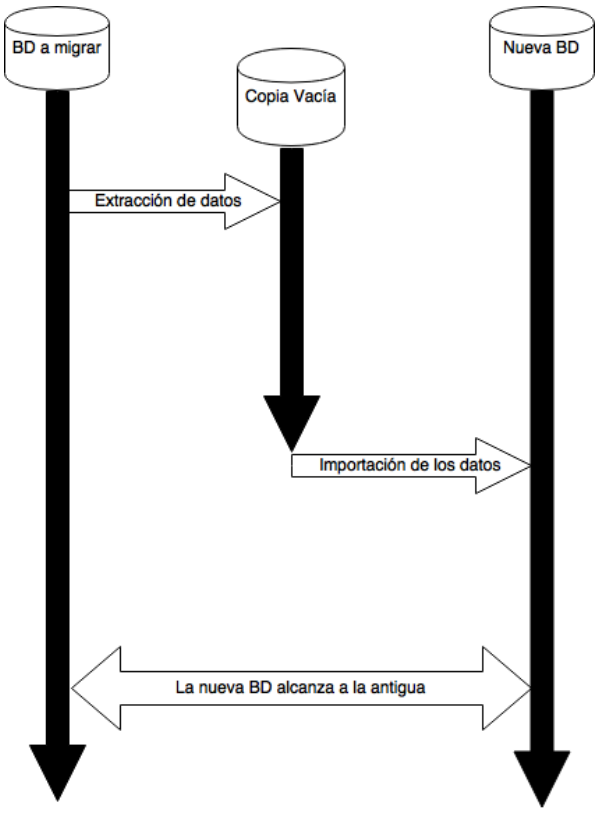
2.4.1. Instalación y actualización

El modelo de instalación para el sistema de recomendación de música debe constar de los siguientes elementos instalados en el siguiente orden: * Grupos de instalación: * Python 2.7: Contiene todo el software necesario para desarrollar el sistema de recomendación, la instalación depende del sistema: * Windows: Es necesario descargar e instalar el software a partir del archivo binario obtenido de la web oficial de python: https://www.python.org/downloads/windows/ * Mac OS X: Es necesario descargar e instalar el software a partir del archivo binario obtenido de la web oficial de python: https://www.python.org/downloads/mac-osx/ * Linux/UNIX: Es necesario descargar e instalar a través de la terminal el software a partir de los archivos obtenido de la web oficial de python: https://www.python.org/downloads/source/ * Librería flask para python: Una vez instalado python la librería se instala ejecutando el comando "pip install flask". * Base de datos: Google App engine trabaja directamente con una base de datos noSQL, tan solo sería necesario instalar el esquema de base de datos en el sistema. * Librería Recomendadora: Se trata de la librería principal del sistema, se darán más detalles de instalación más adelante.

2.4.2. Migración del sistema y los datos

La migración de las funcionalidades podrían realizarse de manera que la antigua versión del sistema se continúe ejecutando de forma paralela a la del sistema con las nuevas o mejoradas funcionalidades, esto ocurre hasta que este último este totalmente operativo entonces se para la ejecución del sistema anterior. Esto es debido a que en nuestro sistema no se realizan peticiones críticas que necesiten de constante rigor. Aunque también sería posible adoptar el modelo big bang, es decir reinicios programados cada cierto tiempo que reinicien las funcionalidades del sistema para aplicar las actualizaciones o añadir nuevas funcionalidades.

La migración de datos es una operación más crítica en el caso de que deba realizarse durante el funcionamiento del sistema, en el caso del modelo big bang es tan sencillo como realizar una copia de la base de datos con el nombre deseado mientras que el sistema está apagado, estaría programado para que se realizara antes del inicio del sistema. En el caso de que la migración se produzca durante la ejecución del sistema se realiza creando una nueva base de datos, se extraen los datos actuales de la anterior base de datos y posteriormente se cargan en la nueva base de datos en el tiempo, se irán realizando las mismas operaciones en ambas bases de datos hasta que las dos llegan al punto en que son iguales y se continua con la nueva base de datos tal y como se refleja en el siguiente gráfico:



2.4.3. Backup y restauración del sistema

El sistema realizará de forma programada copias de la base de datos y se almacenará en un disco duro no conectada a la red, cuya única finalidad es almacenar los datos fechados de cada copia de la base de datos. En caso de que sea necesario una restauración de la base de datos se podría realizar la misma técnica empleada en la migración pero en sentido opuesto.

3. Presupuesto

El presupuesto se basa principalmente en el diagrama de despliegue. Asumiendo que el proyecto recibe el visto bueno y suponiendo que el sistema será capaz de atender al menos a 50-100 usuarios concurrentes en su inicio podemos definir los siguientes presupuestos:

3.1. Sistema “in House”

- Hardware necesario:

    - Servidor (hosting):
    - CPU: Quad-Core
    - RAM: 8-16 GB
    - HDD Online: 20 GB
    - HDD Offline: 30 GB
    - Refrigeración
- Software necesario:
    - Sistema servidor Linux (p.e. CentOS)
    - Python y librerías.
    - Dominio.
- Servicios:
    - Luz.
- Personal/Mantenimiento.

| Elementos | Desglose | Coste inicial | Coste inicial por usuario | coste mensual fijo | Coste mensual por usuario | |————–|————|————|————|————-|————-| | Servidor | - CPU: Quad-Core - RAM: 8-16 GB - HDD Online: 20 GB - HDD Offline: 30 GB - Refrigeración | 1431.67 € | 14.3167 - 28.6324 € | 119,3058 € | 1,193058 - 2,3861 | | Dominio | dominio .com (porkbun) | 8.84 € | 0.0884 - 0.1768 € | 0.7366 € | 0.01473 - 0.00736 €| | Luz | | 180 - 360 € | 2.7 - 5.4 € | 15 - 30 € | 0.225 - 0.45 € | | Mantenimiento | | Personal | | Total |

3.2. Sistema basado en la nube

Estimación con AppEngine:

- 1 instancia por mes
- 219 horas de cálculo CPU al mes (aprox 20 minutos cada hora).
- Memoria usada: 91.25 GB al mes.
- Disco usado: 30 GB al mes.
- 300 MB de tráfico saliente al mes.
- Almacenamiento cloud: 20 GB

Total: 44.20$ al mes $\approx$ 41€ al mes.

4. Diseño arquitectonico