Taller no 1 – Descubrimiento Informacion Utilizando Crawling y Uso de Fuentes RSS

Jose Ramon Alvarez Monto, Fernando Arruza Hedman,
Miguel Alejandro Gonzalez Cardeñoza
Analisis de Informacion sobre Big Data
Universidad de los Andes, Bogotá, Colombia
{jr.alvarez,f.arruza,ma.gonzalez16}@uniandes.edu.co

Fecha de presentación: Septiembre 11 de 2017

Tabla de contenido

1 Introducción 1
2 Descripcion de la solucion 2
3 Metosodos y Tecnologias 2
4 Expresiones en Xquery y RegEx..... 3
5 Analisis y Resultados 3

1 Introducción

El taller no 1, propone la realizacion practica de estrategias de crawling y la utilizacion de fuentes de sindicacion, para el caso del ejercicio los recursos RSS de los portales; eltiempo.com, elmundo.com y Wradio

El desarrollo de las actividades para cumpir de forma satisfactoria con los objetivos, planteados que buscan lograr la recoleccion de informacion por medio de estrategias de crawling del portal Uniandes.edu.co y de los sitios de facultades y/o programas academicos seleccionados. Del mismo modo se propone conocer y gestionar de forma basica los contenidos RSS de .

Como tal el doccumento relaciona en su contenido una descripcion de las actividades realizadas desde los puntos de vista teoricos y practicos, el contenido de lo relacionado con el diseño, creacion e implementacion de la solucion requerida por las instrucciones del taller. De ese modo se busco la manera de lograr cumplir los objetivos propuestos de forma tal que cumpliera con las actividades de recoleccion de informacion sobre los sitios donde se requirio y su organización y almacenamiento en base de datos.

Es importante notar que es un ejercicio practico que fundamenta las actividades de busqueda de informacion que puede ser implementado, realizando ajustes minimos para su correcto funcionamiento, en caso de realizar una selección distinta de fuentes sean portales web o RSS

2 Descripcion de la solucion

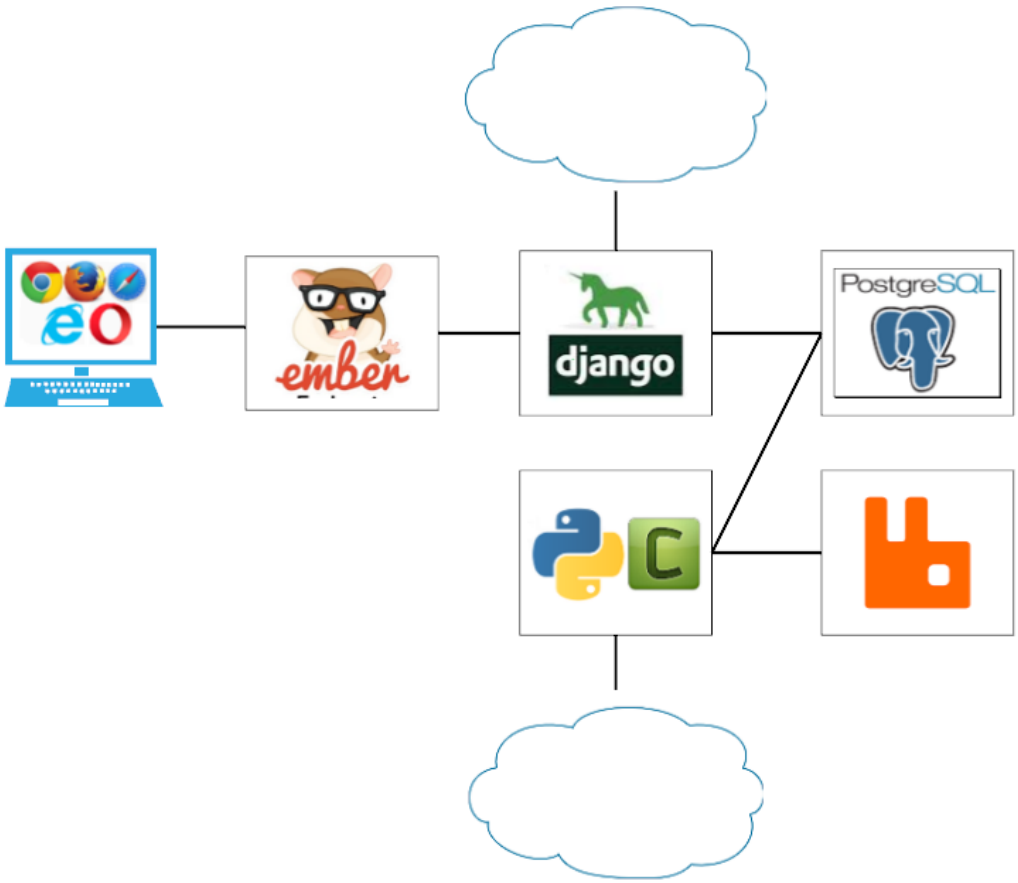

Figura 1. Esquema de despliegue de servicios

La aplicación de la capa cliente fue construida utilizando emberJS y está destinada exclusivamente a la consulta y visualización de información que solicita el cliente final y que provee la capa de servicios implementado en python basado een el framework Django. Las peticiones realizadas desde el navegador son recibidas por la aplicación emberJS y direccionadas a los servicio expuestos por Django. La aplicación emberJS está soportada sobre el servidor y balanceador Ngnix, en cambio la aplicación Django está soportada sobre Gunicorn, ambas se comunican a través de protocolo http mediante servicios REST. La solución incluye en su alcance funcional dos servicios macros, el primero orientado a la consulta de la información de profesores de UNIANDES información que se obtiene a través de la realización de scraping sobre los sitios Web de la intranet de la universidad y el segundo orientado a la consulta de fuentes RSS de noticias de tres medios de comunicación conocidos y prestigiosos. El primero se implementó como una tarea programada ejecutada mediante Celery y cuyo resultado se almacena en una base de datos relacional, soportada sobre PostgreSQL. El segundo se implementó haciendo consulta directa a las fuentes por demanda del usuario final. En ambos casos los resultados se exponen a través de servicios REST y esta infomración es presentada al usuario en la aplicación cliente.

3. Métodos y tecnologías

- Ngnix: servidor Web que publica recursos (aplicación cliente). En caso de colocar varios nodos en la capa de aplicaciones se puede utilizar como balanceador de carga.
- Gunicorn: contenedor de aplicaciones python, utilizado para soportar la aplicación desarrollada con Django.

- Django: framework ágil basado en python para desarrollo de aplicación Web.
- Django Rest Framework: componente se software destinado a la implementación de servicios REST e integrado con Django.
- Celery: manejador de tareas asíncronas y tareas programadas.
- RabbitMQ: servicio de mensajería, basado en el patrón publicador/suscriptor.
- PostgreSQL: manejador de bases de datos relacionales.
- FeedParser: componente de software (librería) desarrollado en python destinado al procesamiento de fuentes RSS.
- Scrapy: componente de software (librería) desarrollado en python destinado a la realización de tareas de búsqueda de contenido en la Web.
- Django-scrapy: conector entre django y scrapy desarrollado en python destinado a gestionar la comunicación entre los elementos antes mencionados, ejemplo la persistencia de información resultante del scrapy en base de datos, a través de los modelos de django.
- Docket: gestor de contenedores. Se utilizó en este ejercicio para desplegar los componentes de la arquitectura.

4. Expresiones en Xquery y RegEx

En la implementación realizada no fue necesario el uso de expresiones regulares debido a que se utilizó FeedParser como componente de procesamiento de fuentes RSS y este abstrae la tarea de parsear los archivos XML, generando la información en listas con formato JSON. Este componente también provee mecanismos para realizar filtrado de información por tanto la navegación en el contenido de la fuente no resulta necesaria.

En la implementación del scraping se utilizaron sentencias XPATH para navegar por el código HTML de las páginas Web y así conseguir obtener las información necesaria.

5. Análisis de los Resultados

- La consulta de fuentes RSS se implementó de manera directa contra el recurso del proveedor y por demanda del usuario final, lo cual es un requisito del enunciado del ejercicio. Esto implica que los tiempos de respuesta a las solicitudes no sean los más adecuados ya que dependen de factores externos que no podemos controlar como: capa de red (intranet e intenet), disponibilidad de la fuente y su desempeño.
- Una alternativa para solucionar lo mencionado en el punto anterior, sería colocar una tareas programada que consulte la fuente RSS y persista los datos. De esta manera las solucitudes de los usuarios de la aplicación serían menos latentes. La frecuencia de actualización de datos dependerá de la variabilidad de los datos de la fuente externa.
- La realización de tareas de scraping implica que los componentes desarrollados se ajusten a la estructura y/o diseño de las páginas a revisar, sobretodo a los niveles de estandarización que se sigan en su construcción. En el caso de la páginas de UNIANDES se pudo comprobar que no siguen lineamientos comunes, incluso esto se presenta dentro de un mismo portal. Esto implica que se tengan que construir procesos de scraping pràcticamente por cada página dentro del sitio.
- La calidad de los resultados alcanzados no es la mejor debido a que los datos extraidos son confiables pero resulta muy dificil garantizar que estén completos.
- Como alternativa de solución, se propone estandarizar el diseño de los portales creando pautas o lineamientos de codificación que sean aplicados por los diferentes equipos dedicados a la tarea.