Sistemas distribuidos
Laboratorio #1

Ayudante: Maximiliano Pérez Rodríguez
Profesor: Daniel Wladdimiro Cottet

Primer semestre 2017

1 Objetivo

Aplicar los conceptos de arquitectura distribuida, realizar una implementación con el patrón de diseño *Publish-Subscribe*

2 Enunciado

Dentro de la Internet no es fácil realizar búsquedas de ciertos tópicos, menos aún que lleguen alertas que estén determinadas por lo publicado por distintos usuarios. Supongamos que deseamos saber las novedades de los últimos libros de SciFi, de esta manera, cuando estamos suscritos a este tópico, todos los autores que realicen publicaciones de este tipo, va a llegarnos una notificación de esto. El diseño de planteado anteriormente se llama *Publish-Subscribe*.

Esta arquitectura basada en coordinación de mensajes de manera acoplada se basa en que existen cierta cantidad de *Publisher*, *Topics* y *Subscriber*. En la Figura 1 se observan los tres componentes mencionados anteriores, donde el *Publisher A* va a enviar mensajes a los tópicos X e Y. Suponiendo que no existe una falla en el envío del mensaje, al llegar este mensaje al tópico X, se deberá enviar un mensaje al *Subscriber O* y al *Subscriber Q*.

Para la inicialización de este sistema, es necesario que se registren los *Publisher* a un tópico en específico, en el caso de no existir, crearlo. Posteriormente, a cada tópico puede registrarse un *Subscriber*, de esta manera, al momento de realizar una publicación por parte del *Publisher*, a cada *Subscriber* registrado le llegará una notificación de lo ocurrido. Cabe destacar que al igual que se pueden publicar en un tópico, el *Publisher* puede solicitar eliminar una publicación o desligarse de un tópico. Por otra parte, el *Subscriber* puede registrarse o desligarse de un tópico, además de un *request update*, el cual es un método que está disponible para procesar las notificaciones que van llegando por parte del tópico. Y por último, el tópico, el cual publica a cada subscriptor en caso que haya habido una nueva publicación.

Cabe destacar que al realizar este proceso, existe un acoplamiento por parte de los entes, debido que si se realiza una acción por el componente *Publisher*, esto va a tener una repercusión directa en el componente *Subscriber*.

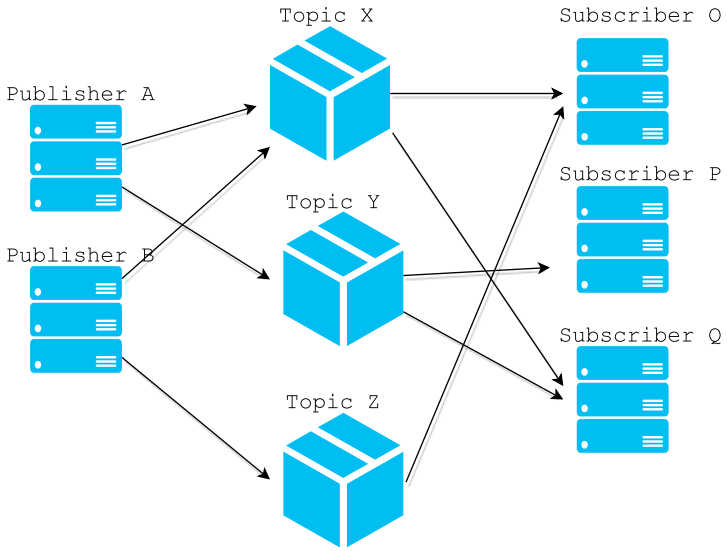

Figure 1: Ejemplo de arquitectura de un sistema distribuido basado en el diseño de Publish-Subscribe

Tomando en consideración los aspectos anteriores, se hace necesario realizar un diseño más pulcro de cada uno de los componentes. A continuación se presentan los distintos métodos que va a poseer cada uno de estos objetos, los cuales van a tener una repercusión en el sistema:

Publish
+ Register publisher
+ Publish
+ Delete publication
+ Deregister publisher

Figure 2: Clase Publisher

Subscriber
+ Register subscriber
+ Request Update
+ Deregister subscriber

Figure 3: Clase Subscriber

Topics
+ Publish

Figure 4: Clase Topic

3 Evaluación

La entrega del laboratorio deberá ser subida al Moodle del curso en Usachvirtual. Este debe ser subido en un archivo comprimido `tar.gz`, con su respectivo archivo de configuración y código fuente para la ejecución del simulador.

En el caso que no se cumpla con ninguna de las solicitudes descritas anteriormente, se evaluará con la nota mínima.

La *fecha de término* del laboratorio es el **20 de Abril**.

4 Importante

- Detección de copias de trabajos (actuales o anteriores) se calificará con la nota mínima.
- Copy/Paste tendrá un descuento de 1 punto si no cuenta con el debido reconocimiento de la fuente desde donde se obtuvieron los extractos del código.
- El trabajo debe ser coherente con lo especificado en el enunciado.
- Por cada día de atraso desde la fecha de entrega del código tendrá un descuento de 1 punto.

5 Referencias

- PeerSim. `http://peersim.sourceforge.net/`
- Tanenbaum, A. & Van Steen, M. - Distributed Systems: Principles and Paradigms. *Cap. 13*