PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERIA
DEPARTAMENTO INGENIERIA DE SISTEMAS

Introducción a Sistemas Distribuidos – Proyecto 1
Período Académico 2019-1

Entrega 19-20 de Marzo de 2019

Consulta Proyectos Comunitarios

Objetivo

El objetivo general del primer proyecto del curso consiste en apropiar los paradigmas de comunicación distribuida, a través de la implementación de un sistema que preste un servicio social utilizando un mecanismo distribuido de comunicación de procesos.

Objetivos Específicos

- Poner en práctica conceptos de sistemas distribuidos en un problema práctico del mundo real.
- Implementar, mediante el uso de sockets e hilos, un mecanismo de comunicación distribuida.
- Implantar mecanismos de tolerancia a fallos para hacer más robusto al sistema.

El proyecto se focalizará en la implementación y uso de mecanismos de comunicación distribuida cliente-servidor, en este caso con uso de *Proxies*, para ofrecer un servicio de consulta que le permita a una comunidad expresar su opinión (nivel de aprobación) sobre proyectos sociales que proponen entidades gubernamentales.

Descripción del Sistema a Desarrollar

El proyecto consiste en desarrollar e implementar una aplicación distribuida para dar acceso a los ciudadanos de una zona territorial para expresar su nivel de aprobación a diferentes iniciativas gubernamentales. Por ejemplo, la alcaldía puede desde su servidor solicitar que se haga una consulta sobre la construcción de una escuela. Las consultas, provenientes de diversas entidades gubernamentales, podrían ser sobre la construcción de un puente, las restricciones de uso del suelo, la realización de un evento social, entre otros. La aplicación debe permitir a un usuario registrar su nivel de aprobación (alto/medio/bajo) sobre un proyecto puesto a consideración de la comunidad.
En el caso de aplicación real se requiere incluir mecanismos de autenticación fuertes. Para este proyecto, dado que se centra en el diseño e implementación distribuida, la información que va desde el cliente será solo el id del usuario, la consulta/proyecto, y el voto (alto/medio/bajo); al conectarse un usuario le debe suministrar al cliente su id. El sistema debe validar que es un id válido y prevenir que vote más de una vez sobre una misma consulta/proyecto.
El sistema debe permitir la conexión de múltiples fuentes de consultas/proyectos a través de múltiples nodos *Proxy*. Los usuarios (asociados a un cliente) hacen el proceso de *binding* a un solo *Proxy*. Cada vez que un cliente hace un *request* a través del *Proxy*, recibe en el *reply* las nuevas consultas/proyectos que hayan sido propuestos por las fuentes. Luego, el cliente hace un nuevo *request* con el voto del usuario para cada uno de las consultas/proyectos propuestos, y espera el *reply* que confirma que el voto ha sido recibido correctamente.

Cada fuente de consultas/proyectos se asimila a una entidad gubernamental que propone proyectos a los ciudadanos. Para facilitar la prueba del sistema estas fuentes deben simular la generación en forma automática de consultas a los ciudadanos. Lo que se requiere es "simular" fuentes de consultas mediante el uso de archivos que incluyan una secuencia de consultas/proyectos, cada una con una etiqueta de tiempo que indique el momento de su ocurrencia. De esta forma, un hilo que forma parte de la fuente de información se puede encargar de ir leyendo este archivo e "inyectando" la información acorde con la etiqueta de tiempo. Una vez el hilo que maneja el archivo genera la información, los procesos encargados de la distribución actúan, aplican los filtros y reenvían la información a quienes corresponda.

Comunicación Distribuida

La lógica del Proxy debe ser acorde con el modelo de caché, tanto en la dirección C/S como en la S/C. Cada entidad gubernamental debe poder consolidar el total de votos de todas sus consultas/proyectos. Finalmente, se debe incluir un mecanismo de tolerancia a fallas para el caso en que un *Proxy* deje de funcionar. Para el manejo de fallos y evitar cuellos de botella, se debe establecer un mecanismo de "*Directorio de Proxies Distribuidos"* que les permita a los procesos encontrar un *Proxy* con bajo nivel de carga (mecanismo de balanceo de carga).

Dados los objetivos de aprendizaje del proyecto, no se pueden utilizar middlewares existentes que ya implementen esquemas de comunicación distribuida. Todo el desarrollo se debe realizar utilizando directamente los mecanismos básicos de comunicación tipo socket y de manejo de hilos de Java.

Equipos de Trabajo

El proyecto se realizará en grupos de trabajo de 3 personas.

Entrega y Condiciones

La entrega se realizará, el martes 19 de marzo (grupo profesor González) y miércoles 20 de marzo (grupo profesor Páez) a las 9am. La sustentación se realizará ese mismo día.

La entrega se debe hacer a través de UVirtual; además del código, se debe adicionar la documentación correspondiente con el esquema de procesos y sus interacciones, el diagrama UML para identificar las clases utilizadas y la descripción general de los componentes principales. No se considera documentación al código fuente, pero si se espera que esté bien estructurado y sea de fácil comprensión. No se requiere hacer interfaces gráficas sofisticadas, pero sí de fácil uso y que permitan verificar el correcto funcionamiento del sistema distribuido corriendo en al menos 3 máquinas de forma ágil y flexible. Se debe hacer la sustentación con el código que se entrega en las máquinas de las salas de los laboratorios de Ing. Sistemas y deben estar presentes todos los integrantes del grupo.

No puede existir replicación de código entre grupos, lo cual se consideraría plagio.

Grupo	G1	G2	G3	G4
Registro de Cliente (Asignación de ID) (5%)				
Validación de usuario con hash (5%)				
Voto de usuario (ID, Consulta/proyecto, voto) (10%)				
Validación (ID válido, voto único) (10%)				
Múltiples fuentes, Consultas/proyectos (10%)				
Binding a un solo proxy (5%)				
Protocolo (request consulta proyecto, request confirmación de voto) (10%)				
Uso de hilos para lectura y envío de información (5%)				
Consolidación de votos (10%)				
Tolerancia a fallos (*proxies*) (10%)				
Identificación de bajo nivel de carga (Directorio de *proxies* distribuidos) (10%)				
Documentación (10%)				