Proyecto final

Materia: Sistemas Distrbuidos
Término: 2017 I
Profesor: Cristina Abad

Número de integrantes: 1 a 3
Tema: Servicio de compresión
Fecha entrega propuesta (proyectos autodefinidos): Primera martes después de examen parcial
Fecha entrega proyecto: Martes 15 de agosto, 8h00 (en mi oficina y vía SidWeb)
Sistema operativo: Linux
Lenguaje de prog.: Java, C/C++, Clojure, Go, Python

Descripción

En el proyecto, deberán implementar un sistema distribuido que utilice un middleware para comunicación entre procesos distribuidos. El middleware utilizado debe estar entre la lista de los requerimientos (ver abajo) y deben documentar claramente cuál escogieron y por qué. Grupos que deseen obtener puntos extra, deben presentarme una propuesta en la que indiquen qué cosa adicional implementarán para obtener dichos puntos.

Requerimientos

- Utilizar Github. Enviarme el enlace al repositorio vía SidWeb, máximo hasta la semana despúes del examen parcial. **TODOS los miembros del grupo deben tener más de un commit en el repositorio. Necesito poder saber en qué trabajó cada uno.**
- Middlewares permitidos:
    - RPC: gRPC o Apache Thrift ← Solamente para propuestas diferentes a la del servicio de compresión descrito abajo
    - Publish/subscribe: Redis Pub/Sub, Apache ActiveMQ, Amazon SNS
    - Message queues (**RECOMENDADO**): Apache ActiveMQ, RabbitMQ, o Amazon SQS
- Deberán entregar un documento final (2 a 4 páginas, no más) en el que documenten el problema y solución propuesta y decisiones de diseño. Opcional (puntos extra): Pruebas de rendimiento de su sistema.

Alternativas para el proyecto final

- Auto-definido: Presentar propuesta.
- Investigación: Conversar conmigo si quieren colaborar en un proyecto de investigación.
- Servicio de compresión en la nube, con dos colas una de alta prioridad y una de baja prioridad. Debe soportar N clientes y M compression servers.

Rúbricas

Propuesta (no es requerido en caso de optar por proyecto de Servicio de Compresión)

No será calificada, pero debe contener lo siguiente:

- URL a repositorio de Github.
- Describir claramente sistema a implementar.
- Indicar qué middleware utilizarán, y justifica correctamente la selección del mismo.
- Lista de elementos que se comprometen a implementar para el proyecto final.
- Lista clara de qué cosas están fuera del alcance del proyecto. ← Si no colocan esto, puede que yo asuma que harán algo y luego pierdan puntos por no haberlo hecho.

Proyecto final

Será calificado sobre 30. La nota irá a calificación de proyecto final (2do parcial).

- Documento final: 5 puntos
    - Sección de metodología describe y justifica correctamente las decisiones de diseño del proyecto (¿Estructuras de datos usados? ¿Librerías o middlewares utilizados? ¿Manejo de errores? ¿Mecanismos de sincronización usados? ¿Otras cosas importantes?): 5 puntos
- Implementación: 25 puntos
    - Usa estructuras de datos adecuadas para el servidor: 3 puntos
    - Usa correctamente los mecanismos de sincronización necesarios para evitar condiciones de carrera (de ser necesario): 3 puntos
    - Programa cliente está implementado, funciona como fue especificado en la propuesta, muestra mensajes de error y no se cae durante las pruebas: 5 puntos
    - Comunicación cliente/servidor implementada correctamente (usando el middleware indicado en la propuesta): 4 puntos
    - Se ha incluido un Makefile que permite compilar el proyecto en la línea de comando + README que explique lo necesario para compilar/instalar el SW: 3 puntos

    - Servidor funciona correctamente, tal y como fue especificado en la propuesta: 5 puntos
    - Demo (sustentación) correctamente diseñado, que permite observar como funciona el sistema, incluyendo varios clientes simultáneamente usando el sistema e interactuando con el servidor; 1 punto por cada cliente simultáneo, hasta 5 puntos (es decir, para sacar este puntaje completo, deben tener al menos 5 clientes conectados al sistema simultáneamente).
    - EXTRAS: Hasta 5 puntos, dependiendo de lo implementado
- Participación en el proyecto: 10 puntos. Esta participación la evaluaré de dos maneras: (1) en la sustentación del proyecto, y (2) vía el log de commits del estudiante en el repositorio del proyecto. Si un estudiante no asiste a la sustentación o no tiene commits en el repositorio Git del proyecto immediatamente tendrá cero (0) en la nota del proyecto.
    - Estudiante demuestra haber participado activamente en el desarrollo del proyecto y demuestra entender su implementación y resultados: 10 puntos
    - Estudiante demuestra haber participado parcialmente el desarrollo del proyecto y demuestra entender parcialmente su implementación y resultados: 7.5 puntos
    - Estudiante demuestra participación limitada en el proyecto y/o no logra comprender los resultados obtenidos: 5 puntos
    - Estudiante demuestra una colaboración mínima en el desarrollo del proyecto: 2.5 puntos
    - Estudiante no colaboró en el desarrollo del proyecto: 0 puntos
- Nota Final: (Nota documento + nota implementación) * (Nota participación en proyecto)/10

Detalles adicionales sobre el proyecto de Servicio de Compresión en la Nube

- Deben implementar un servicio de compresión en la nube (similar al AWS Elastic Transcoder, pero para comprimir varios archivos, en lugar de convertir formatos).
- Para la compresión de los archivos pueden usar programas de línea de comando o librerías.
- Deben usar un middleware de cola de mensajes (ej: AWS SQS, ActiveMQ, RabbitMQ). Puede estar instalado en máquinas virtuales.
- Deben hacer pruebas con al menos tres clientes (en diferentes máquinas virtuales). **Deben tener al menos 2 versiones del sw cliente, en dos lenguajes de programación diferentes.** No es necesario un GUI. Los clientes son los productores de trabajos de compresión.
- Deben hacer pruebas con al menos tres compression servers (en diferentes máquinas virtuales); estos son los consumidores de trabajos de compresión (obtienen un trabajo de la cola y luego lo ejecutan).
- Deben usar dos colas de mensajes: una de alta prioridad y una de baja prioridad.
- Los pedidos deben estar serializados con **Apache Thrift o Protobuf** u algún otro formato binario; **NO con JSON ni XML ni texto plano**.
- Deben definir las operaciones soportadas por su servicio. Por ejemplo, las operaciones soportadas por Elastic Transcoder están listadas en: http://docs.aws.amazon.com/elastictranscoder/latest/developerguide/operations-jobs.html ; el soportar pipelines es opcional, así que solamente necesitan implementar (como mínimo) createJob, readJob y cancelJob.