

INFORME: Actores en Scala

Integrantes:

- **Francisco Joray**
- **Matias Valle**
- **Marcos Cravero**

Desarrollo del proyecto

Actores involucrados en el problema

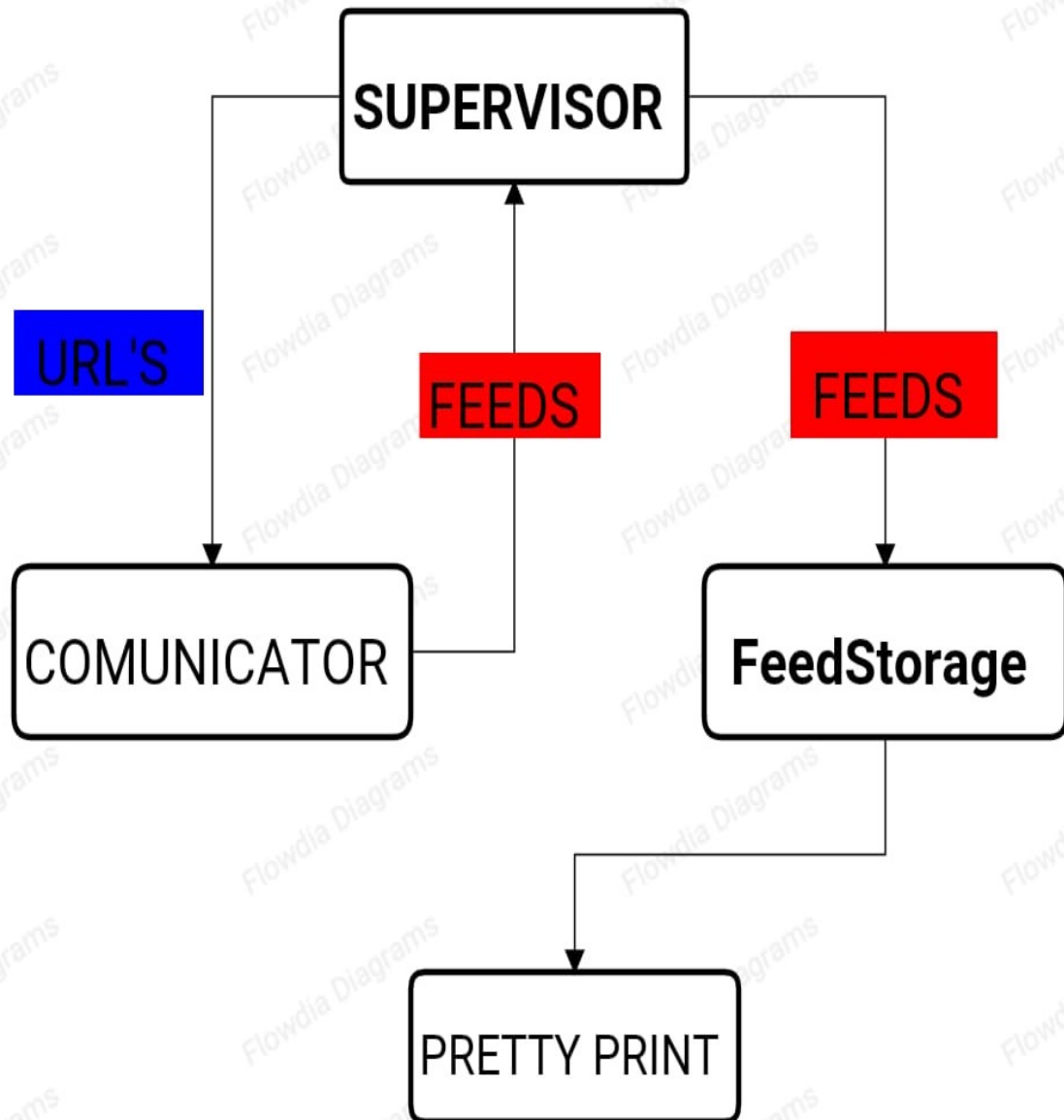
Desarrollamos un sistema el cual esta gobernado por un "Actor Supervisor", que se comunica(enviando y recibiendo mensaje) con dos sub-actores, "Actor FeedStorage" y "Actor Communicator"

Responsabilidades de los agentes encontrados

El sistema esta dirigido por el "Supervisor", que se encarga de iniciar el proceso de construccion del feed. Este actor recibe un mensaje del main con los datos de los feeds a subscribirse y comienza su funcion. Una vez recibido estos datos, realiza un "ask"(pregunta a un actor) hacia el actor "Communicator", que es quien se encarga de comunicarse con los enlaces url's pasados para obtener los feeds a los que el usuario desea subscribirse. Una vez los obtiene, le responde al actor "Supervisor" con un mensaje que contiene el feed, completando el ciclo "ask-pattern".

Ahora que tenemos su respuesta, el supervisor se encarga de pasar este feed, mediante un mensaje, al actor "feedStorage" que es el que se va a encargar de almacenar este feed en una variable para luego imprimirlo por pantalla.

Una vez se terminen de pasar todos los feeds a los que el usuario se quiere subscribir, esperaremos 10 segundos para que se haga todo este proceso y mandaremos al actor "Supervisor" un mensaje que detenga la ejecucion. Con esto nos aseguramos que todo actor termine su trabajo, y si no lo termino, al detener el actor supervisor, se detengan el resto de sus sub-actores.



PREGUNTAS:

- Si quisieran extender el sistema para soportar el conteo de entidades nombradas del laboratorio 2, ¿qué parte de la arquitectura deberían modificar?

Agregaríamos otro actor que se dedique al conteo de entidades nombradas. Con este, podemos modificar el supervisor para que se comuniquen con este nuevo actor que se encargaría de contar las entidades. Usamos un “ask-pattern” para mandarle como mensaje a este nuevo actor el feed y este nos respondería con las entidades nombradas en el feed y ya luego podríamos imprimir la lista de ellas.

- Si quisieran exportar los datos (ítems) de las suscripciones a archivos de texto (en lugar de imprimirlas por pantalla):
 - 1 ¿Qué tipo de patrón de interacción creen que les serviría y por qué? (hint: es mejor acumular todo los ítems antes de guardar nada).

Podríamos tener una variable en el actor supervisor que se encargue de ir recopilando todos los feeds y al final escribirlo en un archivo de texto a todos los datos.

Podemos importar “import java.io._” de java que tiene funciones para escribir en archivos, donde se incluye la clase `BufferedWriter` y `FileWriter`, con métodos para escribir en un archivo.

- 2 ¿Dónde deberían utilizar dicho patrón si quisieran acumular todos los datos totales? ¿Y si lo quisieran hacer por sitio?

En caso de hacerlo por sitio, lo único que deberíamos hacer es cambiar el método de imprimir por pantalla por un método que nos imprima en un archivo de texto.

Si en cambio, queremos escribir los datos totales, basta hacer lo que dijimos antes.

- ¿Qué problema trae implementar este sistema de manera síncrona?

Implementar este sistema de manera síncrona nos evitaría poder suscribir varios feeds en diferentes hilos, ya que, le mandaríamos un feed a la vez para que haga todo el proceso. En cambio al poder hacerlo de manera asíncrona, podemos ir suscribiendo diferentes feeds en diferentes hilos en el mismo momento.

- ¿Qué les asegura el sistema de pasaje de mensajes y cómo se diferencia con un semáforo/mutex?

Este sistema de pasaje de mensajes nos asegura que el sistema no se quedara esperando un mensaje que puede nunca llegar, es decir, una vez que enviamos un mensaje, el actor no se queda esperando a recibir una respuesta, puede ir a checkear su “mailbox” y revisar que otros mensajes tiene y seguir realizando su tarea, mientras que otros actores cumplen tambien con su pequeña tarea de manera que cada uno no depende de otro actor.

Recordemos que cada actor se activa al recibir un mensaje, mientras no tengan mensaje, estos no estan activos.

A diferencia de un semaforo, el pasaje de mensaje no espera a que ocurra un evento para avanzar, cada actor actua de manera asincrona.