# **Lectura y actividad sobre arquitectura de actores**

**Arquitectura y diseño de software**

|  |  |  |
| --- | --- | --- |
| **Táctica** | **Descripción de la táctica** | **Uso/implementación hecha en Play** |
| Transactions | Una transacción es la agrupación de varias etapas sucesivas de tal modo que todo el paquete se puede deshacer de una vez. Las transacciones se utilizan para evitar que los datos se vea afectado si un paso en un proceso falla y también para evitar colisiones entre varios procesos simultáneos que acceden a los mismos datos. | Es aplicada mediante Executor Framework cuando se define una ThreadPoolExecutor, dado que facilita la gestión atómica de operaciones mediante la cola de mensajes principal. |
| Introducir concurrencia | La concurrencia consiste en paralelizar los procesos para optimizar el uso de los recursos computacionales. | Los threads en Play se pueden suspender hasta que se hayan cumplido los requerimientos en los threads paralelos para seguir su ejecución. Suspender y reanudar la ejecución de threads libera threads que estaban anteriormente bloqueados y disminuye la latencia. |
| Cola FIFO distribuida | Asignar multiples colas que procesen las peticiones en orden de llegada por actores ayuda a agilizar el procesamiento de peticiones. | Los clientes mandan peticiones a la capa de servicios. Aquí se les asigna un actor y su petición entra a una cola FIFO (mailbox) para ser procesada por un thread cuando el despachador se lo asigne. Se distribuyen los recursos equitativamente pues todas las peticiones tienen la misma importancia |
| Dynamic priority scheduling: round robin | Cada vez que se genera una posibilidad de asignación de recursos, se asigna el recurso disponible a la siguiente petición. | El despachador se encarga de asignar los threads a algun item de los mailboxes de los actores tan pronto algun thread se libere. |