Patrones de Comunicación

Request-Reply

- **Sincrónico** (bloqueante) por defecto.
- ACK trivial (el mismo reply).
- Operación **asincrónica**: dos REQ-REP para mandar solicitud y esperar rta.
- **Estructura:** `messageID | requestID | operationID | argumentos`.
- **Tolerancia a fallos:** timeout con retries.

Publisher-Subscriber

- Comunicación por **eventos**.
- Productores y consumidores.
- Arquitectura basada en:
  - **Tópicos**: indicando tipo de evento (tópico) -> **BUS**.
  - **Canales**: orientadas a canales específicos -> **colas**.

Pipeline

- `Source - Filter(s) - Sink`.
- **Flujo de datos** procesados **secuencialmente** por filtros.

Modelo de Procesamiento

- **Worker por Filter.** Una unidad de procesamiento a cada etapa del pipeline.
- **Worker por Item.** Una unidad de procesamiento a cada item.
  - Un worker acompaña a un dato paso a paso h/ el final del pipeline.

Tipos de etapas (filtros)

- **Paralela:** cada item es **independiente** de los anteriores y posteriores, **admite paralelismo**.
- **Secuencial:** no puede procesar más de uno a la vez.
  - Los puede retornar **ordenados o desordenados**.

Ventajas

- **Algoritmos online.** Iniciar procesamiento antes de que estén todos los datos.
- **Información infinita.**

Direct Acyclic Graphs (DAG)

- **Instrucciones** modeladas mediante un **grafo de flujo de datos**.
  - Nodos = tareas.
  - Aristas = flujo de información.
- **Acíclicos.**
- Calcular **camino crítico**.
- Admite **lazy loading** (nodos requeridos por dependencias).
- Modelar dependencias entre procesos:
  - Dependencia implica posibilidad de bloqueo.
  - Cíclico implica posibilidad de deadlock.