ARQUITECTURA SOFTWARE

Práctica 4:

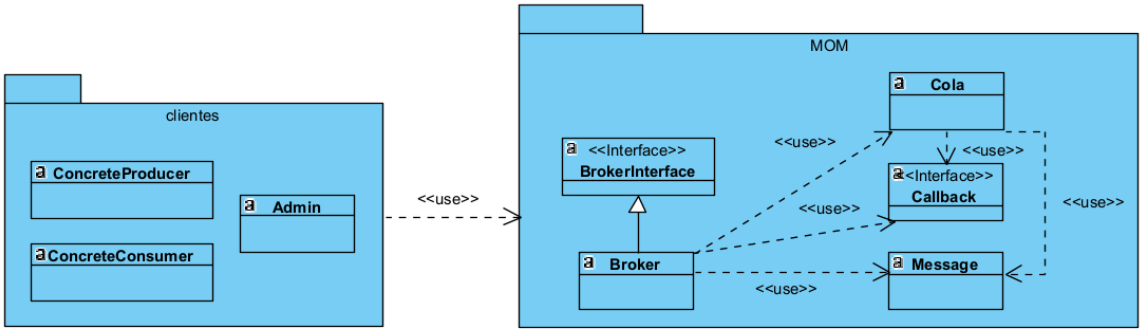
BROKER DE MENSAJES

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

Jorge Hernández 872838
Laura Hernández 872203
20/05/2025

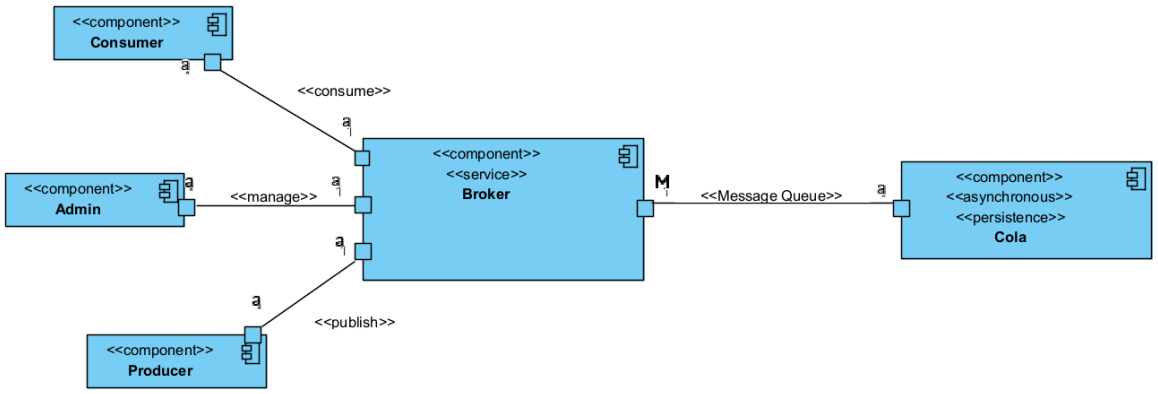
Vista de módulos

La vista del sistema se organiza en dos módulos principales: Clientes y MOM. En el módulo Clientes, destacan las clases *ConcreteProducer* y *ConcreteConsumer*, que representan entidades capaces de producir y consumir mensajes respectivamente. Además, la clase Admin realiza las funcionalidades de listado y eliminación de colas. Por su parte, el módulo MOM agrupa los elementos encargados de la infraestructura de mensajería. La clase Broker implementa la interfaz *BrokerInterface*, lo que permite desacoplar a los clientes de la implementación concreta del intermediario, facilitando así la evolución independiente de los módulos. Otros componentes relevantes de MOM son *Cola*, que representa la estructura de almacenamiento temporal de mensajes, *Message*, que encapsula el contenido transmitido, y *Callback*, que define una interfaz remota que permite al Broker notificar al consumidor cuando llega un mensaje, simulando un procesamiento asíncrono mediante una espera y confirmación (ACK). La relación de uso entre Clientes y MOM refleja claramente el patrón de diseño basado en intermediación, donde el broker centraliza la lógica de comunicación, mejorando la escalabilidad y flexibilidad del sistema.



Vista de componentes y conectores

En esta vista de componentes se destaca el rol central del componente *Broker*, que actúa como intermediario entre productores y consumidores. El *Producer* publica mensajes a través del *Broker*, mientras que el *Consumer* se suscribe y consume los mensajes también mediante el mismo intermediario. Por su parte, el componente *Admin* gestiona la eliminación de colas. La comunicación entre estos componentes está basada en una API común expuesta por el *Broker*, lo cual centraliza la lógica del sistema y favorece el desacoplamiento. El componente *Cola*, etiquetado como asíncrono y persistente, representa una cola de mensajes donde el *Broker* almacena temporalmente los mensajes antes de entregarlos a los consumidores. Esta estructura permite una comunicación desacoplada en el tiempo, donde productores y consumidores no requieren estar activos simultáneamente. Gracias a esta arquitectura, el sistema logra flexibilidad, escalabilidad y tolerancia a fallos, características clave en entornos distribuidos basados en mensajería.



Vista de distribución

Esta vista de despliegue muestra cómo se distribuyen los distintos artefactos del sistema en tres nodos de red independientes, todos ubicados en el laboratorio L1.02 del edificio Ada Byron. La comunicación entre nodos se realiza mediante RMI, lo que permite invocaciones remotas entre las diferentes partes del sistema. En el nodo lab102-204 se despliegan los componentes *Consumer.java* y *Admin.java*, que representan respectivamente al consumidor de mensajes y al administrador del sistema. El nodo lab102-206 aloja *Broker.java*, el componente central que actúa como intermediario entre productores y consumidores, coordinando la entrega de mensajes y gestionando las colas de comunicación. Finalmente, en el nodo lab102-207 se ejecuta *Producer.java*, encargado de generar y enviar mensajes al broker. Esta distribución refleja una separación clara de responsabilidades y permite escalar el sistema horizontalmente, añadiendo más productores o consumidores en distintos nodos sin afectar al núcleo de la arquitectura.