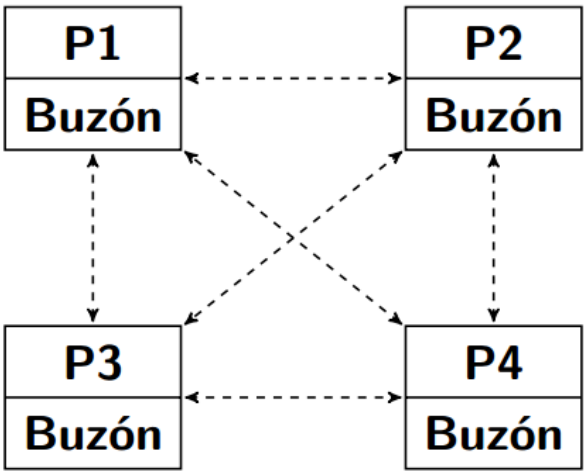
Practica 3
Marta Frías Zapater (535621)
Juan Vela García (643821)

Resumen

En todo sistema distribuido se necesita intercambiar información entre distintos componentes del mismo. Para ello existen diversos paradigmas, entre los que se encuentra el de paso de mensajes. Por lo general, este tipo de modelo no garantiza un estado global consistente y los resultados son impredecibles, por lo que entra en juego la sincronización entre los distintos participantes, los cuales deberán enviarse mensajes entre ellos para ponerse de acuerdo sobre quien puede acceder a dicho recurso y quien no (espera de turno) en cada momento del tiempo.

Teniendo en cuenta que existe un gran número de situaciones posibles que hacen que un algoritmo de sincronización sea eficiente para unos casos y no lo sea para otros, se debe buscar la forma de simplificar la tarea abstrayendo cualquier funcionalidad externa y de menor nivel que el propio algoritmo. Se propone por tanto un sistema de intercambio de mensajes, entre un número indefinido de procesos (locales o remotos), que dé soporte a casi cualquier tipo de mensaje, y con la adicción de un buzón de mensajes recibidos pero no procesados (recepción asíncrona por parte del proceso) en previsión de que no siempre se pueden atender las peticiones instantáneamente (eliminando el acoplamiento temporal).



Esquema del paso de mensajes entre cuatro procesos con buzones de recepción asíncrona.

Se ha tenido especial cuidado en la documentación y la estructuración de todo el sistema con previsión de su futura reutilización

1. Sistema de paso de mensajes

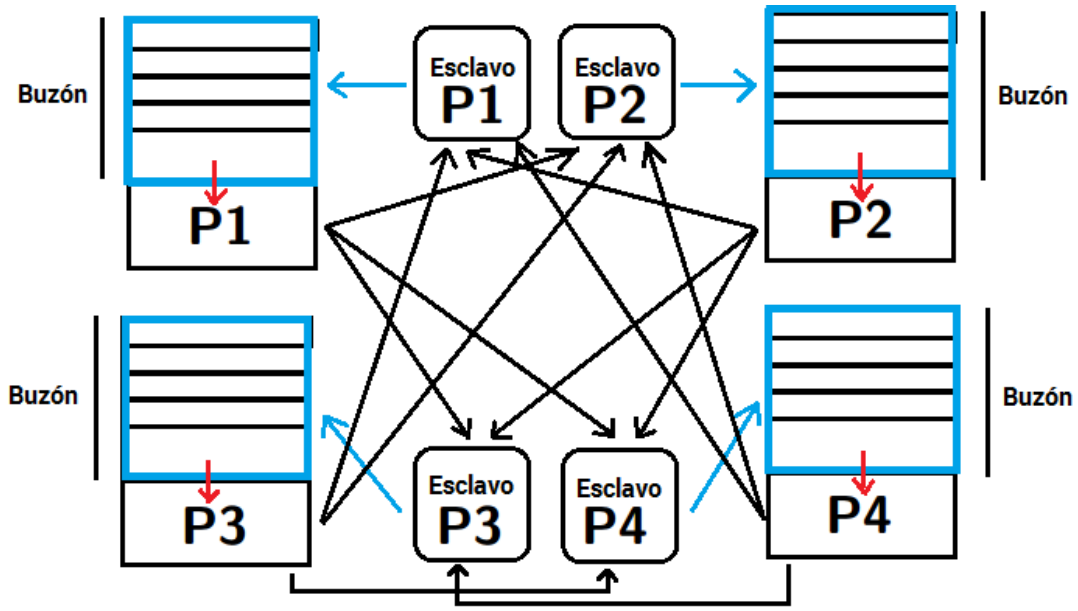
El sistema está compuesto por **dos elementos principales** *y dos secundarios.* Los dos principales, el sistema de mensajes y su esclavo, están relacionados entré si por una dependencia productor-consumidor (siendo el buzón el elemento intermedio), por lo que se ha adoptado el patrón de diseño con el mismo nombre para solucionar la problemática derivada.

1. **Sistema de mensajes :** Es el encargado de proporcionar las funcionalidades de envío y recepción de mensajes a otros procesos. Solo puede haber uno por cada proceso.

2. **Esclavo del sistema de mensajes :** Es un "hilo" encargado de atender los mensajes recibidos (concurrentemente con el sistema de mensajes), y de almacenarlos en el buzón. Solo puede haber uno por cada sistema de mensajes.

3. *Buzón de mensajes* **:** Es el encargado de proporcionar servicios de almacenamiento y recuperación de mensajes no procesados. Garantiza la integridad del buzón y de la recuperación de mensajes. Solo puede haber uno por cada sistema de mensajes.

4. *Sobre* **:** Se encarga de encapsular el mensaje a enviar y los datos de identificación del emisor y el receptor. Puede haber tantos como se requieran.



Las flechas negras representan los mensajes (sobres) enviados desde un proceso hasta el esclavo receptor. Las flechas azules representan una inserción (en exclusión mutua y siempre que el buzón no esté lleno) de un mensaje en el buzón. Las flechas rojas representan la obtención (en exclusión mutua y bloqueante si el buzón está vacío) de un mensaje del buzón por parte de un proceso.

1.1. Sistema de mensajes

En el momento en que se inicia, obtiene del fichero de red las direcciones de los demás procesos y el puerto en el que deberá estar a la espera el esclavo tras ser iniciado. Cuando se quiere enviar un mensaje a otro proceso se localiza su dirección a partir de su identificador, y tras iniciar una conexión entre los dos, el mensaje se codifica en una secuencia de bytes y se envía. Puede ocurrir que el buzón del destinatario esté lleno, en cuyo caso el mensaje se descarta. Por otra parte, cuando se quiere recibir un mensaje, se extrae del buzón y se recupera. Puede ocurrir que el buzón esté vacío, en cuyo caso se detendrá la ejecución hasta que se reciba algo (más detalles en Buzón). Por último, cuando se quiere detener el sistema se da la orden de parada al esclavo. Dado que éste está bloqueado, se inicia una conexión local y se envía un mensaje irrelevante, de modo que se vuelva a comprobar la condición de parada y el esclavo termine íntegramente.

1.2. Esclavo del sistema de mensajes

En el momento en que se inicia entra en un bucle que se ejecutará hasta que se dé la orden de parada. Al comienzo de cada iteración se bloquea a la espera de conexiones en la dirección especificada. Cuando otro proceso inicia una conexión se recibe y decodifica el mensaje, para posteriormente almacenarlo en el buzón.

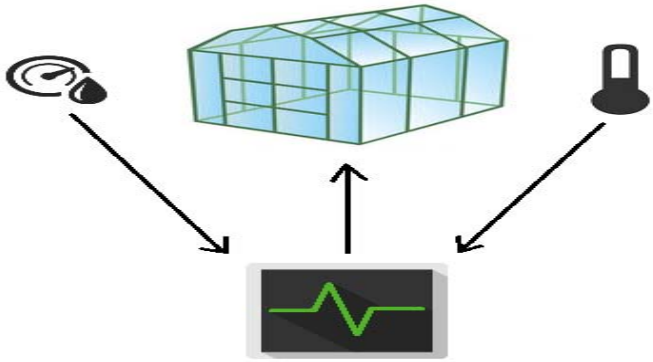
1.3. Buzón de mensajes

En el momento en que se crea se inicializa la estructura interna de almacenamiento (de ahora en adelante *eia*) y se preparan los mecanismos de control de acceso a la sección crítica (la *eia*). Cada vez que se extrae un mensaje del buzón se bloquea el acceso a las secciones que modifican la *eia*, de modo que todo acceso se haga en exclusión mutua, manteniendo la consistencia del estado. Además, si el buzón está vacío, se detiene la recepción hasta que haya al menos un elemento disponible. Cuando termina se desbloquea el acceso. Por otra parte, cada vez que se añade un mensaje al buzón se bloquea el acceso a las secciones que modifican la *eia*, de modo que todo cambio efectuado mantenga la consistencia del estado. Cuando termina se desbloquea el acceso.

1.4. Sobre

Es la estructura que permite el intercambio de mensajes atómicos entre procesos al encapsular los datos necesarios para la comunicación en una sola entidad. Se puede transformar en una secuencia de bytes.

2. Aplicación de gestión de un invernadero



Se propone un sistema gestor de un invernadero automatizado, y dos sensores enviando datos periódica y simultáneamente. El proceso identificado con el número 1 se encargará de hacer las funciones de sistema gestor, y el resto de hacer las funciones de sensores que se comunican con él. El gestor atenderá los mensajes de los sensores secuencialmente, actualizando su estado y tomando decisiones en consecuencia. Sin embargo, el hecho de que sea secuencial no garantiza la consistencia, ya que distintas ordenaciones de las operaciones pueden dar lugar a distintos entrelazados y resultados.

2.1. Traza de error

A continuación se presentan las trazas de dos ejecuciones del sistema distribuido. Se puede observar que el simple hecho de recibir ciertos datos en un orden distinto puede desencadenar distintas acciones.

Traza 1:

Sensor-> LA HUMEDAD ES ALTA
Sensor-> LA TEMPERATURA ES ALTA
Gestor -> SE HA ACTIVADO LA VENTILACION

Traza 2 :

Sensor-> LA TEMPERATURA ES ALTA
Gestor -> SE HA ACTIVADO LA PULVERIZACION
Sensor-> LA HUMEDAD ES ALTA
Gestor -> SE HA ACTIVADO LA VENTILACION
Gestor -> SE HA DETENIDO LA PULVERIZACION
Gestor -> AVISO: La cosecha esta en peligro!!!

3. Trazabilidad

Se han realizado pruebas con un distinto número de procesos, pruebas básicas de número y formato de argumentos; y de existencia, formato y corrección del fichero de red. Además, se ha probado el sistema sobre dos dispositivos unidos por una red. Se ha utilizado la aplicación de ejemplo como prueba para el intercambio de mensajes ya que se realizan las dos operaciones básicas (enviar y recibir). Con ayuda del depurador se pueden observar diferentes entrelazados y sus efectos sobre el estado global de la base de datos.