PMS (Notas - Explicación practica)

Sintaxis

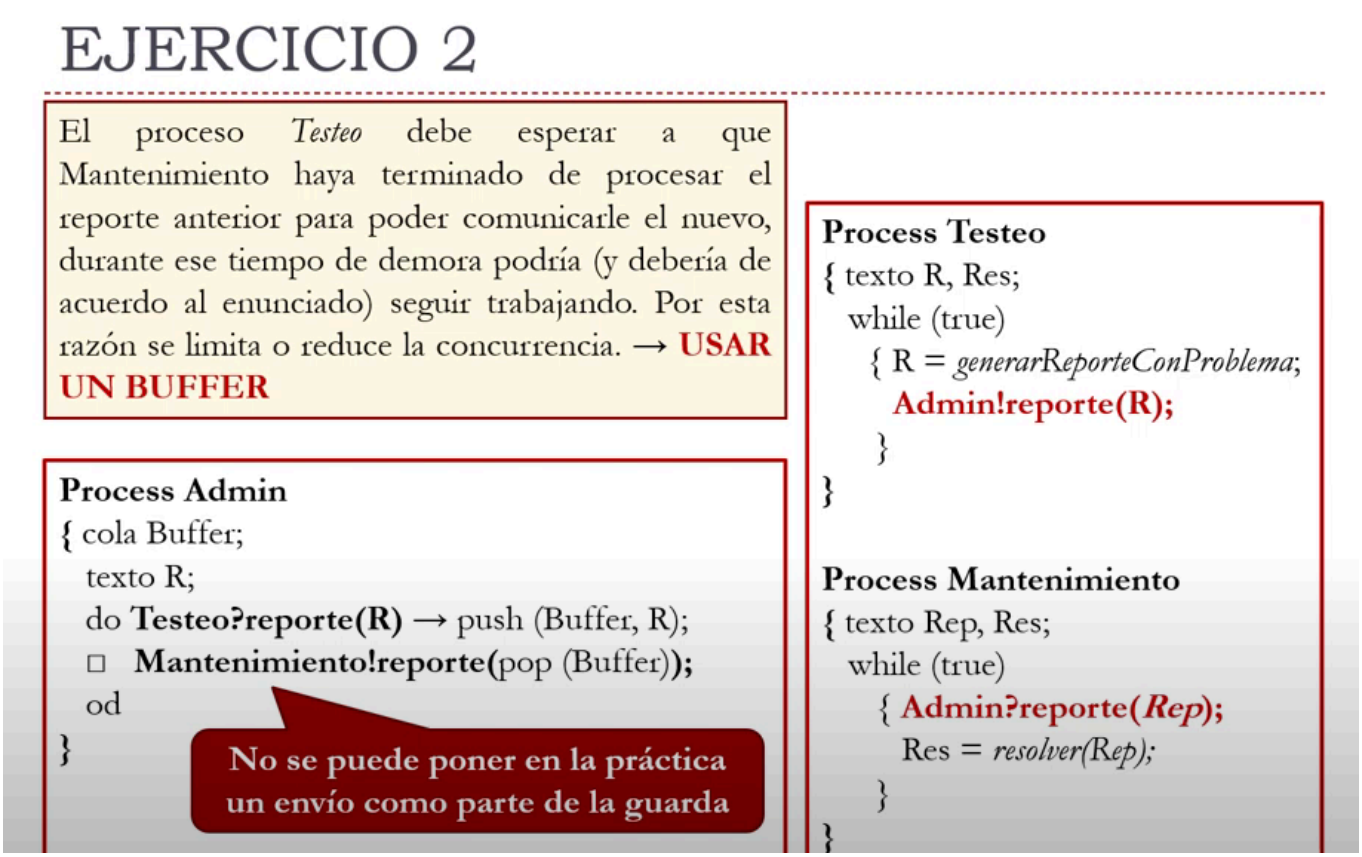
No es necesario la declaración de canales ya que es implícito. Existe un canal por cada par de Process.

```
//Para el envio

ProcessDestino!(parametro)
ProcessDestino!PuertoOpcional(parametro)

ProcessOrigen[num]!PuertoOpcional(id,parametro) //-> Solo puede enviarse un numero no puede usarse el comodin * para enviar a cualquiera...

//Para la recepción

ProcessOrigen?(parametro)
ProcessOrigen?PuertoOpcional(parametro)

ProcessOrigen[num]?PuertoOpcional(id,parametro)
ProcessOrigen[*]?PuertoOpcional(id,parametro) // --> Solo puede usarse el * en la recepción

// Si en el envio y/o en la recepción se acorda un puerto opcional se debe respetar el uso del mismo en las dos acciónes!!
```

Envio

En el envió el proceso origen se queda esperando a que el proceso receptor reciba su mensaje.

No se puede usar una sentencia de envio en la PRACTICA como parte de la guarda :



El DO

El do corta cuando todas las guardas son falsas.

El bucle tiene **guardas** (condiciones de habilitación) separadas por `[]`, o sea, alternativas.
El formato general es:

```
do
    G1 -> S1
[]  G2 -> S2
[]  G3 -> S3
od
```

- En cada iteración, el proceso **elige una de las ramas cuyo guarda `Gi` sea verdadera**.
- Si **ninguna guarda está habilitada**, **el `do` termina** (ese es el “corte”).
- Si al menos una sigue habilitada, se ejecuta una de ellas, y después vuelve a evaluar todas.

> Las guardas en la practica solo pueden ser de recepción.....

Ejercicio 2

EJERCICIO 2

La segunda guarda debe tener una condición booleana, ya que no se puede procesar el pedido de *Mantenimiento* si no hay reportes pendientes. Por lo tanto debo demorar la recepción de ese mensaje hasta estar en condiciones de entregarle un reporte.

```
Process Admin
{ cola Buffer;
  texto R;
  do Testeo?reporte(R) → push (Buffer, R);
  □ not empty(Buffer); Mantenimiento?pedido() →
                Mantenimiento!reporte (pop (Buffer));
  od
}
```

```
Process Testeo
{ texto R, Res;
  while (true)
    { R = generarReporteConProblema;
      Admin!reporte(R);
    }
}

Process Mantenimiento
{ texto Rep, Res;
  while (true)
    { Admin!pedido();
      Admin?reporte(Rep);
      Res = resolver(Rep);
    }
}
```

> Cosa a tener en cuenta :
>
> 1. Como no se puede tener una recepción como parte de la guarda lo que hace el proceso Mantenimiento es enviar una mensaje como un SIGNAL y este le responderá con lo pedido.
> 2. Debemos tener en cuenta que no se debe enviar nada de la cola si la misma esta vacía y como no podemos hacer uso de if para preguntar sobre la condición de la cola. Se pone una condición a cumplirse en la guarda.
> 3. En este caso usamos un proceso Admin ya que queremos que el proceso Testeo no se quede esperando a que el proceso Mantenimiento reciba el reporte. Como no hay datos compartidos esta involucrado este nuevo proceso.

Ejercicio 3

EJERCICIO 3

```
Process Admin
{ cola Fila;   texto R;   int idT;

  do Testeo[*]?reporte(R, idT) → push (Fila, (R,idT));
  □  not empty(Fila); Mantenimiento?pedido() → pop (Fila, (R, idT))
                                               Mantenimiento!reporte (R, idT);
  od
}
```

```
Process Testeo[id: 0 ..N-1]
{ texto R, Res;
  while (true)
    { R = generarReporteConProblema;
      Admin!reporte(R, id);
      Mantenimiento?respuesta(Res);
    }
}
```

```
Process Mantenimiento
{ texto Rep, Res;      int idT;
  while (true)
    { Admin!pedido();
      Admin?reporte(Rep, idT);
      Res = resolver(Rep);
      Testeo[idT]!respuesta(Res);
    }
}
```

Este proceso Admin nos demuestra que puede usarse tanto como para mantener un orden y/o para mantener los datos acá y que el los procesos puedan seguir trabajando de forma normal y no deban esperarse mutuamente.
Cosas a tener en cuenta :

1. Para que se respete el orden lo que debe hacerse es mandarse el id ya que el admin recibirá el pedido de cualquier id por eso el uso de * comodín y pusheara el reporte con el id. Para que luego se le envié a ese mismo id el resultado.

Ejercicio 4

EJERCICIO 4

```
Process Admin
{ cola Fila;    texto R;    int idT, idM;

  do Testeo[*]?reporte(R, idT) → push (Fila, (R,idT));
  □ not empty(Fila); Mantenimiento[*]?pedido(idM)→
                                        pop (Fila, (R, idT))
                                        Mantenimiento[idM]!reporte (R, idT);
  od
}
```

```
Process Testeo[id: 0 ..N-1]
{ texto R, Res;
  while (true)
    { R = generarReporteConProblema;
      Admin!reporte(R, id);
      Mantenimiento?respuesta(Res);
    }
}
```

```
Process Mantenimiento[id: 0..2]
{ texto Rep, Res;       int idT;
  while (true)
    { Admin!pedido(id);
      Admin?reporte(Rep, idT);
      Res = resolver(Rep);
      Testeo[idT]!respuesta(Res);
    }
}
```

> Como en el envió no se puede usar el comodín, solo es en la recepción. Entonces cuando hay más de una persona de mantenimiento lo que se cambia nomas es de donde se recibe desde el admin y a quien se lo envía.