Grupo Miércoles 12:00-14:00 semanas A
- Informe previo / Práctica 4 -
Autores:
Félix Ozcoz Eraso 801108
Héctor Toral Pallás 798095

Variables compartidas:

int N_W : tamaño del tren
int w : personas dentro del vagón
bool puertasSalidaAbiertas
bool puertasEntradaAbiertas
condition_variable trenVacio : para esperar a que el tren esté vacío

Variables añadidas MonitorTren:

Compartidas:

bool limpiando: indicador si se está limpiando el tren
condition_variableconductor : sirve para esperar a que el tren esté lleno; para esperar que todos los usuarios hayan bajado y para esperar que termine la limpieza del tren.
condition_variable puedesBajar: para esperar a poder descender del tren

No compartidas:

condition_variable limpiar: (sólo usada por el proceso limpieza); para esperar a limpieza del tren

Sincronización de procesos:

Las sincronización se producirá a la entrada al tren, durante la limpieza y durante la bajada del tren.

Durante la entrada, podrán entrar w pasajeros como máximo, el resto de usuarios quedan a la espera de otro viaje.

Durante la bajada de pasajeros, el conductor espera a que quede el tren vació y no permite subir a ningún usuario hasta que haya terminado la limpieza, y entonces comienza la limpieza.

La limpieza comienza si el conductor se lo notifica, en cualquier otro caso debe esperar; comienza limpieza nadie puede subir al tren, finaliza limpieza, avisa conductor, y deja subir pasajeros.

Especificación MonitorTren:

```
monitor MonitorTren

        //-------------------------------  variables
        integer N_W
        integer w
        boolean puertasSalidaAbiertas
        boolean puertasEntradaAbiertas
        condition trenVacio

        //-------------------------------- variables nuevas añadidas
        boolean limpiando
        condition conductor
        condition limpiar
        condition puedesBajar

        //-------------------------------- operaciones

        // procesos usuario
        operation monta()
                <await   w > N_W  and  !puertasEntradaAbiertas
                        waitC(trenVacio)

                >
                w++
                <await w == N_W
                        signalC_all(conductor)

                >
        end


        operation desmonta()
                <await !puertasSalidaAbiertas
                        waitC(puedesBajar)

                >
                w--
                <await w == 0
                        signalC_all(conductor)
                >
        end

        // procesos conductor
        operation iniciaViaje()
                puertasEntradaAbiertas := true
                puertasSalidaAbiertas := false
                <await w < N_W
                        waitC(conductor)
                >
                puertaEntradaAbiertas := false
        end


        operation avisaFinViaje()
                <puertaSalidaAbiertas := true>
        end
```

```
operation esperaHayanBajado()
        signalC_all(puedesBajar)
        <await w > 0
                waitC(conductor)
        >
        puertasSalidaAbiertas := false;
end


operation avisaLimpieza()
        limpiando := true
        signalC_all(limpiar)
end


operation esperaFinLimpieza()
        <await limpiando
                waitC(conductor)
        >
        puertasEntradaAbiertas := true;
        signalC_all(trenVacio)
end


// proceso limpieza
operation esperaAvisoInicio()
        <await !limpiando
                waitC(limpiar)
        >
end

operation avisaFinLimpieza()
        limpiando := false
        signalC_all(conductor)
end
```

Esbozo alto nivel:

```
proceso usuario()
        MonitorTren.monta(i)
        MonitorTren.desmonta(i)
end

proceso conductor()
        MonitorTren.iniciaViaje(i)
        // viaje
        MonitorTren.avisaFinViaje(i)
        MonitorTren.esperaHayanBajado(i)
        MonitorTren.avisaLimpieza(i)
        MonitorTren.esperaFinLimpieza(i)
end

proceso limpieza()
        MonitorTren.esperaAvisoInicio(i)
        // limpiando...
        MonitorTren.avisarFinLimpieza(i)
end
```