Taller de Programación Concurrente por Paso de Mensajes

Modelos y Paradigmas de Programación

Escuela de Ingeniería de Sistemas y Computación

Universidad del Valle

Profesor Juan Francisco DIAZ FRIAS*

A continuación Usted encontrará una serie de ejercicios que debe resolver **individualmente**. No se admite ninguna consulta con sus compañeros de otros grupos de trabajo, pero sí con el profesor. Esa es la hipótesis básica sobre la cual se fundamenta esta tarea. Cualquier violación a esta regla será considerada un intento de copia y causará la anulación del taller.

La solución a todos y cada uno de los puntos del taller se debe almacenar en un archivo `.oz` tal como se describe en cada punto. Usted debe entregar como solución al examen, el conjunto de esos archivos empaquetado y comprimido en un solo archivo cuyo nombre debe ser análogo a `taller4JFDiaz.zip` o `taller4JFDiaz.tgz` o `taller4JFDiaz.tar.gz` o `taller4JFDiaz.rar`. Al desempaquetar este archivo, debo encontrar solamente los archivos siguientes: `contador.oz`, `portero.oz`, `contador2.oz`, `PuertosConCeldas.oz`, y `CeldasConPuertos.oz` con el contenido descrito abajo. Adicionalmente debe entregar un archivo `practica4.pdf`, con el desarrollo del taller que no corresponde a código Oz.

No envíe ningún archivo adicional, así lo haya utilizado para sus pruebas. Su solución debe ser enviada a través del Campus Virtual en el enlace correspondiente al Taller de Programación Concurrente Declarativa según las fechas indicadas allí. El sistema no permitirá que se envíen soluciones después de dicha fecha.

1. *Implementando puertos con celdas.* En el curso se introdujo el concepto de puerto, el cual es un canal de comunicación sencillo. Los puertos tienen las operaciones `{NuevoPuerto S P}`, la cual devuelve

*juanfco.diaz@correounivalle.edu.co

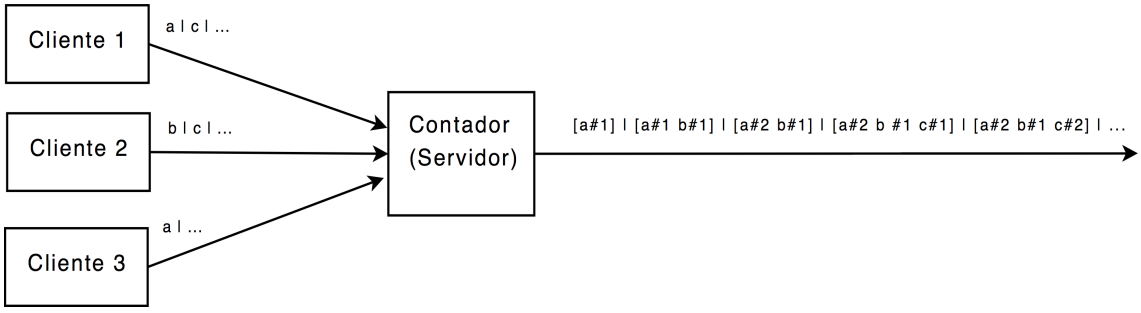
un puerto `P` con flujo `S`, y `{Enviar P X}`, la cual envía un mensaje `X` por el puerto `P`. De acuerdo a estas operaciones, es claro que los puertos son un TAD con estado. En este ejercicio, implemente los puertos en términos de celdas, utilizando las técnicas presentadas en el libro para construir TADs con estado, empaquetados y seguros (con procedimiento de despacho). Su solución debe venir en un archivo denominado **`PuertosConCeldas.oz`** con la implementación de `NuevoPuerto` y `Enviar`.

2. *Implementando celdas con puertos.* Así mismo, en el curso se introdujo el concepto de celda, la cual es un contenedor de valores que pueden cambiar en el tiempo. Las celdas se pueden ver como un TAD con las operaciones siguientes:

Operación	Descripción
`{NuevaCelda X C}`	Crea una celda nueva `C` con contenido inicial `X`.
`{Acceder X C}`	Liga `X` con el contenido actual de la celda `C`.
`{Asignar X C}`	Coloca a `X` como el contenido nuevo de la celda `C`.

   En este ejercicio, implemente las celdas con objetos puerto. Su solución debe venir en un archivo denominado **`CeldasConPuertos.oz`** con la implementación de `NuevaCelda`, `Acceder` y `Asignar`.

3. Ahora que ya conoce el uso de puertos y el modelo de concurrencia por paso de mensajes, seguro resultará más facil implementar el contador con múltiples clientes del taller en concurrencia declarativa (ver figura abajo). Recuerde que el contador debe registrar las ocurrencias de los caracteres enviados por varios clientes y generar un flujo de salida donde cada posición del flujo corresponde al total de ocurrencias hasta ese momento.



   - La figura ilustra una posible salida para las entradas dadas por los clientes. Sin embargo, esta no es la unica salida posible. ¿Por qué? ¿Cuáles son las otras posibles salidas para la misma entrada?

   Su solución debe guardarla en un archivo de nombre `contador.oz`.

4. Un objeto puerto es un concepto muy importante que es utilizado en los proyectos de programación por paso de mensajes. Como usted sabe, las propiedades más importantes de los objetos son el estado y la encapsulación. Estas dos propiedades (entre otras que hacen parte del modelo Orientado

a Objetos) están también presentes en los objetos puerto. Un objeto puerto inicia como cualquier objeto con un estado inicial. Cada vez que un objeto puerto recibe un mensaje, este mensaje es pasado junto a su estado a una función que implementa el comportamiento encapsulado del objeto. Esta función retorna el nuevo estado y el objeto está listo para recibir el siguiente mensaje. El siguiente código corresponde a una implementación de objeto puerto:

```
fun {NuevoObjetoPuerto Comp Inic}
   proc {MsgLoop S1 Estado}
      case S1 of Msg|S2 then
          {MsgLoop S2 {Comp Msg Estado}}
      [] nil then skip
      end
   end
   Sin
in
   thread {MsgLoop Sin Inic} end
   {NewPort Sin}
end
```

La función retorna un puerto y recibe como argumentos una función llamada `Comp` y el estado inicial `Inic`.

a) Implemente una función `Portero` que cuente cuantas personas están en un concierto. `Portero` usa un objeto puerto que recibe 3 clases de mensajes: `getIn(N)`, `getOut(N)` y `getCount(N)`. El mensaje `getIn(N)` incrementa el contador interno con `N` personas. El mensaje `getOut(N)` decrementa el contador como se espera. Finalmente, el portero puede ser consultado en cualquier momento sobre cuantas personas están en el interior con el mensaje `getCount(N)`, el cual liga la variable `N` al número actual de personas.

   Su solución debe guardarla en un archivo de nombre `portero.oz`.

b) Escriba otra versión del servidor `Contador` usando la abstracción `NuevoObjetoPuerto`. Note que aparte del estado interno, el servidor también necesita retornar un flujo de listas con la cantidad de caracteres contados en cada momento. Implemente la función `{Counter Output}` que retorna un objeto puerto. La variable `Output` corresponde al flujo de salida con todos las caracteres contados en el servidor cada vez que recibe un mensaje.

   Pista: Piense en la salida como un flujo de otro puerto. Luego, la función `Comp`, que cambia el estado del servidor, primero enviará el estado al puerto de salida y luego retornará el nuevo estado.

   Su solución debe guardarla en un archivo de nombre `contador2.oz`.