Programación 4

EXAMEN JULIO 2022

Por favor siga las siguientes indicaciones:

- Escriba con lápiz y de un solo lado de las hojas
- Escriba su nombre y número de documento en todas las hojas que entregue
- Numere las hojas e indique el total de hojas en la primera de ellas
- Recuerde entregar su número de parcial junto al parcial
- Está prohibido el uso de computadoras, tabletas o teléfonos durante el parcial

Problema 1 (30 puntos)

Se desea modelar un sistema para gestionar la toma de exámenes. Cada examen corresponde a un único curso, aunque un curso puede tener más de un examen asociado. Cada examen tiene además una fecha y un salón asignado. De los cursos se sabe el nombre y el conjunto de estudiantes inscriptos en el curso. Los estudiantes tienen un número de cédula de identidad, un nombre y pueden ser de grado o de posgrado. De los de grado se sabe la generación a la cual pertenecen y de los de posgrado su área de especialización (por ej. seguridad informática). Luego de que el curso está finalizado, para cada estudiante inscripto en ese curso, se sabe si exoneró, aprobó o reprobó el mismo. Solo quienes hayan aprobado pueden rendir sus exámenes. Cuando un estudiante rinde un examen interesa registrar su número de examen.

Además, se cuenta con la descripción del siguiente Caso de Uso:

Caso de Uso	Registrar asistentes a un examen
Actor	Docente
Descripción	El caso de uso comienza cuando un docente indica que desea registrar los estudiantes que efectivamente asisten a rendir un examen. Para esto pide al sistema la lista de todos los cursos registrados. Luego, el docente indica el nombre del curso para el cual desea registrar estudiantes, la fecha y el salón del examen y el sistema retorna la lista de estudiantes habilitados para rendir el examen. El docente registra como asistente al examen de a un estudiante a la vez en el sistema y el sistema devuelve el número de examen generado para ese estudiante. Cuando el docente termina de registrar estudiantes, debe confirmar o cancelar el registro. En caso de que el docente confirme, el sistema registra la información y devuelve la cédula de identidad y los números de examen de los estudiantes registrados para ese examen, indicando también el nombre del curso, el salón y la fecha. En caso de que el docente cancele, se elimina el registro de lo realizado.

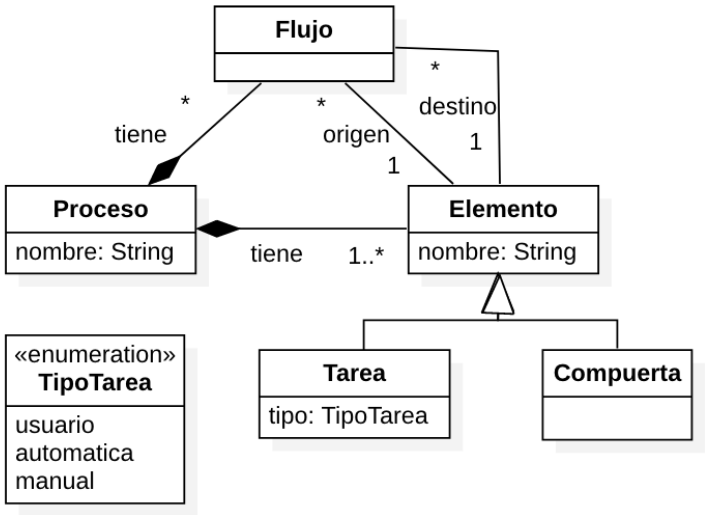
Se pide:

a) ¿Cuál es el propósito de realizar modelos de dominio?
b) Realizar el Modelo de Dominio de la realidad planteada, incluyendo solamente restricciones de integridad circular en lenguaje natural.
c) Realizar un Diagrama de Secuencia del Sistema (DSS) para el Caso de Uso "Registrar asistentes a un examen" indicando el uso de memoria del Sistema y de datatypes, si corresponde.

Problema 2 (35 puntos)

Un motor de procesos permite automatizar un conjunto de pasos que se realizan en un orden determinado para lograr un objetivo. Un proceso se compone de elementos que pueden ser tareas o compuertas para tomar decisiones. Las tareas pueden ser realizadas por un usuario, ser automáticas o manuales. Además, el proceso está compuesto de flujos que permiten conectar un elemento de origen con uno de destino y definen un orden de ejecución entre ellos. En tanto las tareas tienen como máximo un flujo de entrada y uno de salida, las compuertas pueden tener varios flujos de entrada o salida.

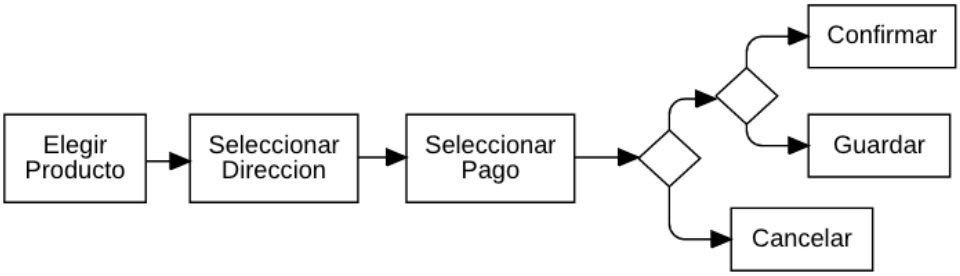
Se realizó el siguiente Modelo de Dominio de la realidad.



Restricciones:

- El nombre del Proceso es único.
- El nombre del Elemento es único dentro de un Proceso.
- Si un Flujo pertenece a un Proceso, los Elementos que conecta deben pertenecer al mismo Proceso.
- Una Tarea está relacionada con un único Flujo como máximo a través de la relación "origen" y con un único Flujo como máximo a través de la relación "destino"

Por ejemplo, en la siguiente figura se muestra un proceso de compra. El proceso está compuesto de seis tareas (rectángulos), dos compuertas (rombo) y siete flujos (flechas). Las tres primeras tareas: elegir producto, seleccionar dirección y seleccionar pago, se realizan en secuencia conectadas por flujos. Luego de seleccionar pago, hay un flujo a una compuerta que permite decidir si se hace una de dos cosas: cancelar la compra, o una segunda compuerta que permite confirmar o guardar la compra para después.



Se definió la siguiente operación del sistema.

`proximasTareas(String proc, String elem): Set(String)`	
Descripciòn	Retorna los nombres de las siguientes tareas de usuario a ser ejecutadas a partir de un elemento del proceso.
Parámetros	String `proc`: nombre del proceso String `elem`: nombre del elemento
Precondiciones	• Existe un Proceso con nombre `proc` • Existe un Elemento con nombre `elem` que pertenece al Proceso de nombre `proc`
Postcondiciones	• Devuelve los nombres de todas las tareas de usuario que le siguen al elemento, a través de sus flujos de salida. En caso de que el/los siguientes elementos sean una compuerta, se retornan también los nombres de las tareas de usuario que le siguen a través de sus flujos de salida, recursivamente.

Se pide:

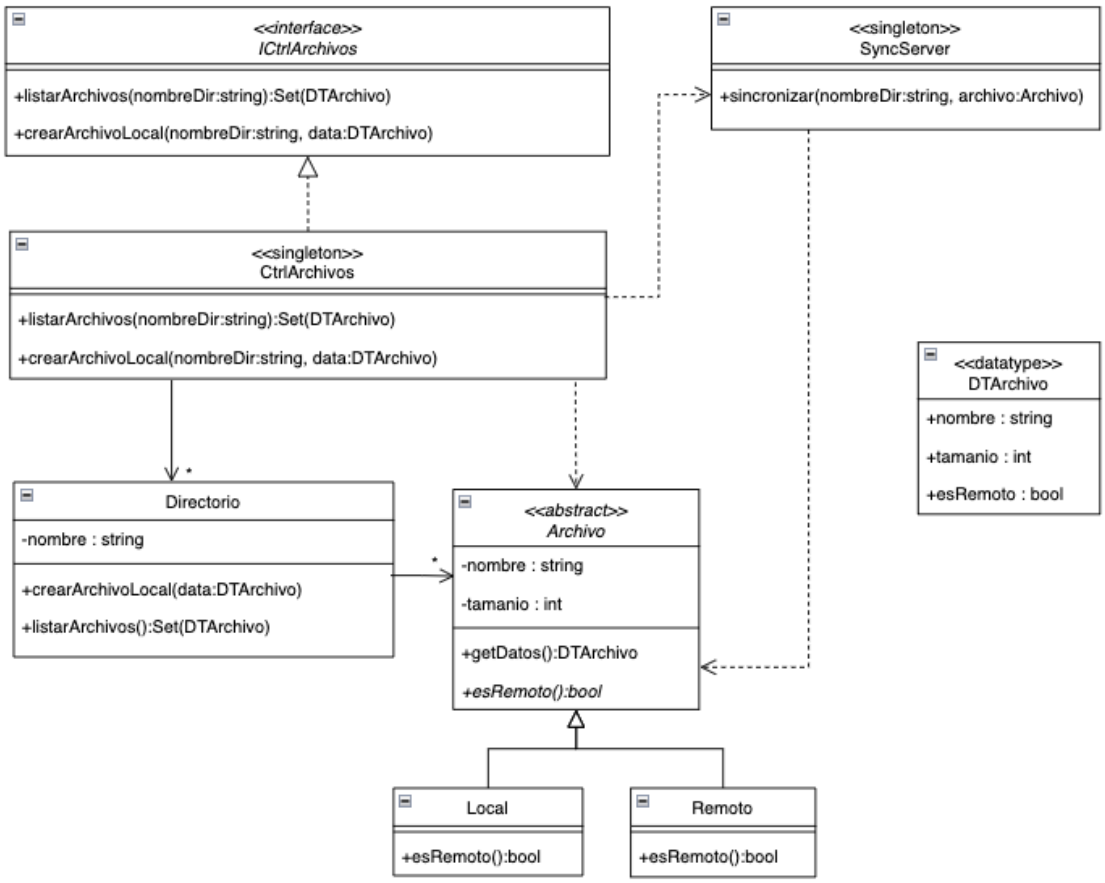
a) Realizar el Diagrama de Comunicación correspondiente a la operación especificada.
b) Realizar el Diagrama de Clases de Diseño (DCD) resultante del diagrama anterior.

Problema 3 (35 puntos)

Se desea implementar un sistema que gestione archivos distribuidos dentro de una red. El sistema permite a los usuarios crear y eliminar directorios y archivos compartidos dentro de la red, que luego serán sincronizados en todas las terminales de la red que cuenten con el sistema instalado.

El equipo de desarrollo generó el siguiente Diagrama de Clases parcial que se deberá seguir en la implementación. Tenga en cuenta que:

- La operación `listarArchivos` permite obtener los datos de todos los archivos ubicados en un directorio.
- La operación `crearArchivoLocal` crea un archivo dentro del directorio indicado y luego sincroniza los cambios realizados en el sistema mediante una llamada a la operación `SyncServer::sincronizar`.



Se pide:

a. Implementar en C++ los .h de las clases `ICtrlArchivos` y `CtrlArchivos`.
b. Implementar en C++ los .h de las clases `Archivo` y `Local`.
c. Implementar en C++ el .cpp de la clase `CtrlArchivos`. No incluya destructor. Incluya código de manejo de excepciones en la operación `crearArchivoLocal()` de la clase `CtrlArchivos` para el caso en que no exista el directorio en donde se crea el archivo.

Observaciones:

- Puede utilizar colecciones genéricas (realizaciones de `IDictionary` e `ICollection`) o paramétricas (contenedores STL).
- No incluir directivas al precompilador (#include, etc).
- No es necesario implementar setters y getters adicionales de las clases, salvo que se requieran en algún método.
- No implementar el datatype.