# Programación 4

EXAMEN FEBRERO 2017

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 examen junto al examen.
- Está prohibido el uso de computadoras, tabletas o teléfonos durante el parcial.

## Problema 1 (35 puntos)

*Parte A:*

a) Explique brevemente la diferencia entre agregación compuesta y agregación compartida.

b) Explique qué es un tipo asociativo.

*Parte B:*

Se desea modelar un sistema de gestión de tareas para proyectos de desarrollo de software. En este se utilizan proyectos para agrupar tareas y usuarios asignados a trabajar en ellas. Los usuarios, identificados por un email, pueden crean y asignar tareas dentro de los proyectos a los que pertenezcan. Estas tareas se identifican por un código, y pueden ser opcionalmente asignadas a un único usuario. De cada tarea se conoce su título y descripción, fecha de creación y estado (Pendiente, En Curso, Completada). Existen distintos tipos más específicos de tarea: Bug y Mejora, para los cuales se registra la prioridad y fecha de entrega respectivamente. Los usuarios pueden agregar comentarios a las tareas de sus proyectos, donde se registra su fecha y contenido. Además se pueden adjuntar recursos al proyecto, estos se identifican por un nombre, y pueden ser asociados opcionalmente a una tarea dentro del proyecto por los usuarios. De cada asociación interesa registrar el recurso, la tarea, el usuario y la fecha. Los usuarios además pueden comentar sobre la cada una de estas asociaciones.

| | |
|---|---|
| Caso de Uso: | ComentarRecursoAsociado |
| Actor: | Usuario |
| Descripción: | Este caso de uso comienza cuando el usuario indica que quiere comentar sobre un recurso asociado a una tarea, para ello indica la tarea sobre la que quiere comentar. El sistema despliega una lista de recursos asociados a esta y el usuario elige sobre cual quiere comentar e indica el contenido del comentario, el cual es dado de alta. |

**Se pide:**

a) Modelo de Dominio de la realidad anterior (considerando el Caso de Uso) con restricciones en lenguaje natural.

b) Diagrama de Secuencia del Sistema (DSS) del Caso de Uso, incluyendo el uso de Datatypes y de manejo de memoria del Sistema, en caso de ser necesario.

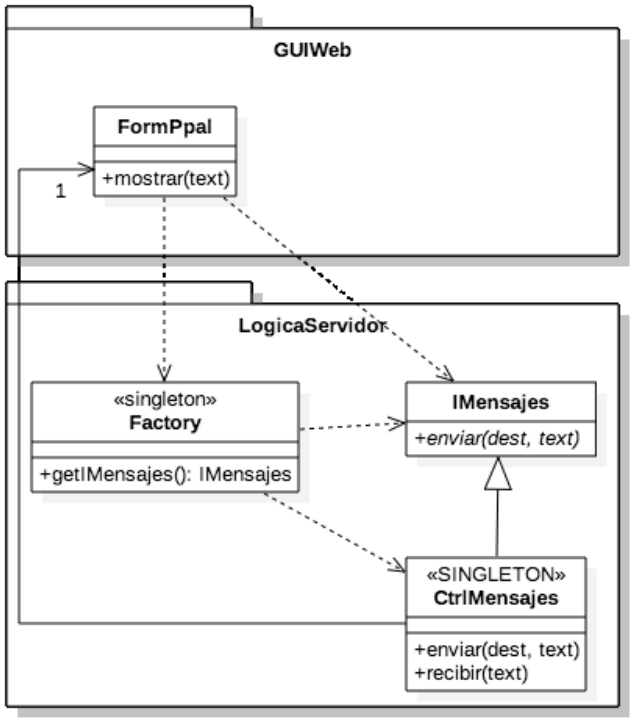
## Problema 2 (30 puntos)

*Parte A:*

¿Cuáles son las dos actividades (o sub-etapas) en la Etapa de Diseño?

*Parte B:*

Imagine que ud. recientemente se ha cambiado de trabajo a una posición de Arquitecto de Software en una reconocida empresa de productos de software del Uruguay, y en su primer día de trabajo le muestran el siguiente Diagrama de Clases:




Este diagrama representa la arquitectura y diseño actual del producto estrella de la empresa denominado "Kezmo", el cual es un producto para colaboración empresarial que contiene, entre otras prestaciones, un chat, el cual es el foco del diagrama anterior.

En términos generales, este diagrama muestra la Capa Lógica de Kezmo (LogicaServidor), la cual permite a la Capa Presentación (GUIWeb) enviar y recibir mensajes. La clase FormPpal representa el formulario principal de la solución Web, el cual permite mostrar un mensaje al usuario en la forma de pop-up o ventana emergente.

El envío de mensajes se realiza mediante la interfaz IMensajes (notar la operación enviar(dest,text)) a la cual la GUI accede mediante la fábrica, mientras que la recepción de mensajes comienza cuando alguien invoca a la operación recibir(text) de la clase controladora, la cual muestra el mensaje invocando a la operación mostrar(text) del formulario principal.

**Se pide:**

a) Realice un Diagrama de Comunicación mostrando las interacciones necesarias, según lo descrito anteriormente, para recibir un mensaje.

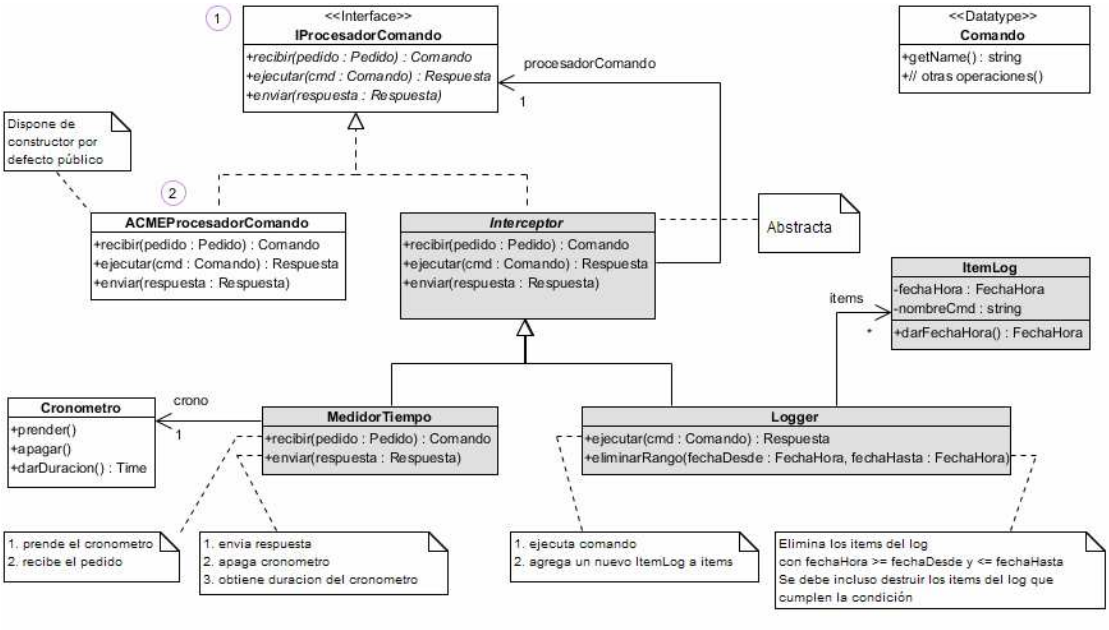
b) ¿Qué crítica le haría a esta solución? Justifique su respuesta.

c) ¿Qué patrón(es) de diseño propondría considerando además que la empresa se encuentra a punto de desarrollar las distintas versiones móviles (iOS, Android), cada una con su propia Capa Presentación pero utilizando la misma Capa Lógica y que se pretende que sea la Lógica la que actualice automáticamente la Presentación?

d) Realice el nuevo Diagrama de Clases, contemplando su solución.

## Problema 3 (35 puntos)

Suponga que ud. trabaja para una empresa que cuenta con una interfaz que brinda el servicio de procesar comandos (**`IProcesadorComando`**) y con una implementación de la misma (**`ACMEProcesadorComando`**) adquirida por su empresa de la que no dispone su código fuente. Dicha interfaz ofrece 3 operaciones: **`recibir`**, **`ejecutar`** y **`enviar`**. Ver 1 y 2 del diagrama de clases de diseño abajo. Básicamente, un procesador de comando recibe un pedido y crea un comando, ejecuta el comando y crea la respuesta, y finalmente envía la respuesta.




Su empresa necesita agregar funcionalidades a la implementación adquirida. A corto plazo, requiere poder medir la duración de procesar un comando y mantener un log básico de los comandos ejecutados. Estas funcionalidades las quiere poder asociar indistintamente a las instancias de la clase **`ACMEProcesadorComando`**. Por ejemplo, a una instancia poderle medir la duración, o mantener un log o ambas cosas a la vez. A otra instancia de la misma clase podría interesar no asociarle ninguna funcionalidad adicional.

Para llevar esto adelante se realizó el diseño con los elementos que aparecen en gris en el diagrama de clases. La idea seguida consistió en la definición de una clase abstracta **`Interceptor`** que encapsula a una instancia que realiza a la interfaz. **`Interceptor`** simplemente delega sus operaciones a la instancia que encapsula. Es cada especialización de **`Interceptor`** que redefine las operaciones que corresponda de acuerdo a la funcionalidad que se quiere brindar. Observar que tanto **`MedidorTiempo`** como **`Logger`** también están realizando a la misma y única interfaz. Las redefiniciones que tienen lugar se indican en las notas. **`Logger`** a su vez provee una operación adicional **`eliminarRango`** que permite depurar el volumen de información que mantiene.

**Considerar:**

- Suponga la existencia de la interface *ICollectible* e implementaciones de *ICollection* (clase *List*), *IKey* (clase *OrderedKey*), *IDictionary* (clase *OrderedDictionary*) e *IIterator*, según sea necesario.
- Es posible utilizar las clases *set<T>*, *vector<T>* o *map<K,T>* de la STL.
- Las implementaciones deben incluir constructores y destructores.
- No implementar setters y getters, salvo la operación **`darFechaHora`**.
- No incluir directivas al precompilador.

**Se pide:**

1) Crear en c++ los objetos siguientes dando una línea de código c++ para cada parte comenzando con la forma **`IProcesadorComando* obj = …;`**

   a) Un objeto que permita trabajar con una instancia del procesador de comandos adquirido sin medición de duración ni log.

   b) Un objeto que permita trabajar con una instancia del procesador de comandos adquirido y medición de duración.

   c) Un objeto que permita trabajar con una instancia del procesador de comandos adquirido y con ambas funcionalidades: medición de duración y log.

2) Teniendo en cuenta los considerandos, implementar .h y .cpp en c++ de **`IProcesadorComando`**, **`Interceptor`**, **`MedidorTiempo`**, **`Logger`** e **`ItemLog`**. Asumir que:

   – el resto de lo que aparece en el diagrama de clases está ya implementado de acuerdo a como está especificado en él, y

   – el data type **`FechaHora`** tiene sobrecargado todos los operadores de comparación.