Actividades Prácticas - Guía Nro. 2

Patrones de Diseño

Modalidad: Grupal

Entrega: En PDF, con carátula que incluya el número de grupo y nombre de los integrantes. Subir al repositorio del grupo y avisar al docente por Slack (fecha límite 11/06).

Observ.: Cada diagrama deberá incluir el listado de autores y revisores (individual).

Bibliografía:

- Design Patterns: Elements of Reusable Object-Oriented Software, Erich Gamma, Richard Helm, Ralph Johnson y John M. Vlissides, Addison-Wesley Professional, 1994.
- Design Patterns Java Workbook, Steve Metsker, Addison Wesley, 2002.
- Apuntes del curso en aula virtual.

Recomendaciones:

- Considerar de qué forma los patrones resuelven problemas de diseño.
- Leer la sección que describe el propósito de cada patrón.
- Estudiar las interrelaciones entre los patrones.
- Estudiar la Estructura, Participantes y Colaboraciones.
- Hacer referencia al ejemplo de código fuente.
- Implementar las clases y métodos relacionados con el patrón.
- No es correcta la idea de tratar de emplear patrones tanto como sea posible.
- La experiencia es la mejor herramienta para saber cuándo un patrón de diseño es apropiado y cuando no.

Ejercicio 1:

En una empresa existen dos tipos de empleados:

- Los empleados de planta permanente que cobran la cantidad de horas trabajadas por $30, más un bono por antigüedad y un monto fijo por salario familiar. Asimismo se les aplica un descuento por obra social.
- Los empleados de planta temporaria que cobran la cantidad de horas trabajadas por $20, más salario familiar (no cobran antigüedad ni se les descuenta obra social).

Con un diagrama de clases y **al menos 1 patrón de diseño**, defina una jerarquía de empleados que:

  - Sea polimórfica respecto al cálculo del sueldo.
  - Defina una única manera de calcular el sueldo.
  - Sea extensible para agregar futuros tipos de empleados.

Ejercicio 2:

Se debe diseñar un sistema que simula un sistema de archivos (filesystem) para un sistema operativo. Se trabajará con archivos y carpetas. Las carpetas pueden contener archivos u otras carpetas.

Se desea contar con el método tamanio() donde:

- Para un archivo, es el tamaño del mismo.
- Para una carpeta, es el tamaño de los archivos que contiene.

¿Cómo lo diseñaría? ¿Es apropiado utilizar algún patrón de diseño?

Ejercicio 3:

El objetivo es modelar el sistema operativo de un teléfono móvil. El equipo debe responder a los siguientes eventos:

- Llamada entrante.
- Corte llamada.
- Llamada rechazada.
- Corte llamada participante.

Mientras una llamada está establecida, puede recibir otras llamadas. En ese caso, las opciones son:

- Pasar a conferencia tripartita.
- Rechazar la llamada.
- Si se pasa a conferencia, las llamadas siguientes se rechazan.

Modo reunión:

- Se apaga el timbre.
- No se reciben llamadas; son registradas para que el usuario las vea más tarde.
- Si el usuario rechaza 3 llamadas consecutivas, el teléfono automáticamente se pone en modo reunión.
- Si el usuario realiza una llamada durante el modo reunión, el teléfono vuelve al modo normal.

¿Con qué patrón de diseño modelaría este comportamiento? Explicar con un diagrama de clases. También el caso “en-llamada a modo-reunión” con un diagrama de secuencia.

Ejercicio 4:

El objetivo es diseñar un cliente de email (similar al Mozilla Thunderbird o MS Outlook). Este cliente manejará una única cuenta de mail. Debe permitir enviar, recibir, copiar y mover emails entre carpetas. Una cuenta tiene una carpeta de entrada predeterminada (Inbox) y una carpeta en la cual se almacenan los mensajes enviados (Sent Mails). Algunas de las características con las que se deberá contar son:

- El usuario puede exportar los mensajes y carpetas en diferentes formatos. Por ejemplo, exportar las carpetas a directorios y subdirectorios y los emails a archivos de texto dentro de los directorios, o exportar a un único archivo XML.

- Para descargar los mensajes, el cliente debe conectarse con el servidor de correo. Existen diferentes servidores de email o Mail Transfer Agents (MTA), por ejemplo, Sendmail y Postfix. Existen clases que se conectan directamente con estos servidores, la clase SendmailConnection y la clase PostfixConnection, pero ambas definen un método diferente para recibir email, por ejemplo: en SendmailConnection el método es getMail() y en PostfixConnection el método es lookupEmail(). Es deseable que el cliente de mail pueda usar cualquiera de estas clases de manera indistinta.

Proponer un diseño que pueda incluir uno o más patrones de diseño. Explicar la solución con un diagrama de clases y otro de componentes.

Ejercicio 5:

Se debe diseñar una aplicación móvil (app) para la venta de artículos de computación. La app deberá mostrar en un listado los artículos junto con una imagen, una descripción resumida y el precio. También debe permitir seleccionar uno o más artículos, especificar la forma de pago y envío una vez confirmada la operación de compra. Algunas de las funcionalidades que deberá ofrecer son:

- Listado de artículos: Los datos de los artículos a mostrar en un listado (por categoría o como resultado de una búsqueda) pueden provenir de diferentes fuentes de datos (servicios web de proveedores). El objetivo es poder recorrer de manera uniforme la colección de artículos recuperada.

- La app deberá permitir a los usuarios suscribirse en anuncios de ofertas. Para evitar el excesivo consumo de datos, se debe diseñar una solución donde sea el servidor quien notifique a los clientes (app) las promociones especiales y no los clientes quienes lo consulten si hay nuevas ofertas.

Modelar el sistema mediante el paradigma OO y expresar la solución con un diagrama de clases. También incluir información de despliegue.

Ejercicio 6:

Telefoni es una startup que se dedica a la compra y venta de celulares usados. Su negocio consiste en adquirir dispositivos usados, y luego de un proceso de diagnóstico y reacondicionamiento, ofrecerlos nuevamente para su venta. La empresa adquiere celulares a través de distintos acuerdos con empresas de telefonía. Una vez que los equipos son reacondicionados, estos son ofrecidos al público masivo a través de una plataforma de comercio electrónico (e-commerce) propia de la empresa.
Telefoni actualmente cuenta con un sistema que le permite gestionar las ventas, pero con un limitado control de stock y además no soporta la gestión de compras. Es por eso que consideró necesario el desarrollo de un nuevo sistema que permita soportar todos los procesos operativos de la empresa.
El equipo funcional de Telefoni realizó un estudio del sistema a diseñar, del cual surgió el siguiente listado de requerimientos funcionales:

1. El sistema deberá soportar múltiples tipos de operaciones, que en principio serán: Compra y Venta.
2. Cada Operación deberá contener la fecha-hora en la que fue realizada, también la marca, modelo y número de serie del celular que fue comprado o vendido, el monto total de la misma, y el usuario que la registró. En las operaciones de compra, también deberá quedar registrado el origen de la compra (la telefónica de donde proviene ese equipo).
3. Cada operación impactará en el stock por unidad, además el sistema deberá registrar ubicación (isla/estantería) donde está depositada la unidad y si fue inspeccionado, reacondicionado o descartado. Cada una de las transiciones deberá tener asociada la marca temporal, de modo tal de poder conocer el tiempo transcurrido en cada una, la antecesora y la siguiente.
4. Para las operaciones de venta se deberá soportar múltiples medios de pago, existiendo hasta el momento: Efectivo, Transferencia bancaria, Tarjeta de crédito y Medios electrónicos (Ej.PayPal). Tanto para el pago con Tarjeta, como para el Pago Electrónico, el sistema debe comunicarse con los distintos Gateways de pago. Para realizar esta tarea, los proveedores de los medios de pago exponen diversas APIs REST (una para el pago con tarjeta, y otra para el electrónico) que facilitan la operación e integración entre sistemas.
5. Una vez que la compra se confirmó, el pedido ingresa al subsistema de Logística (Aplicación Web) del depósito de Telefoni, donde un técnico se encargará de obtener el equipo solicitado, empaquetarlo y trasladarlo hasta el área de despacho donde luego será retirada por el correo para ser entregado al cliente.
6. El comprador puede monitorear el estado de su pedido a través de una aplicación mobile que Telefoni provee. El mismo será notificado de las actualizaciones de estados por los que vaya pasando su pedido (por ejemplo, cuando el pedido tiene asignado un equipo, deberá notificar, así como también cuando el pedido sea empaquetado y depositado en el área de despacho).

Modelar el sistema con el paradigma OO y expresar la solución con un diagrama de clases. Si lo considera apropiado, puede utilizar patrones de diseño. Explicar el caso “nueva compra” con un diagrama de secuencia. También describir componentes y despliegue con una vista combinada.