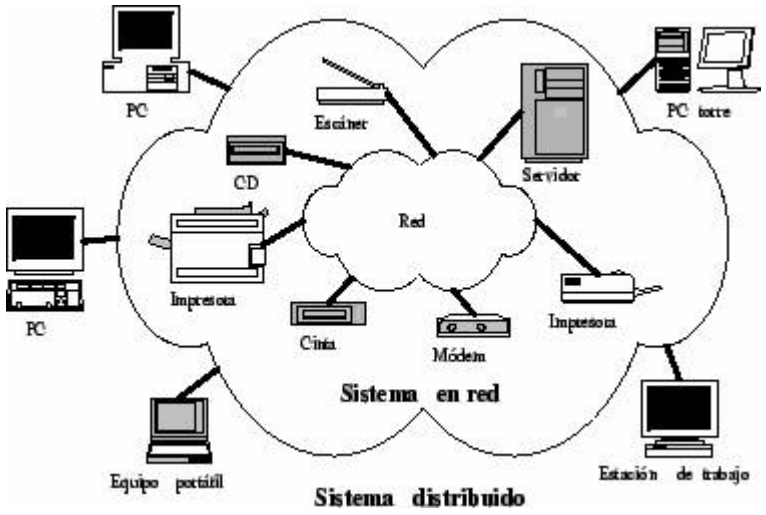
Sistemas distribuidos
Arquitectura de software
Patrones de diseño

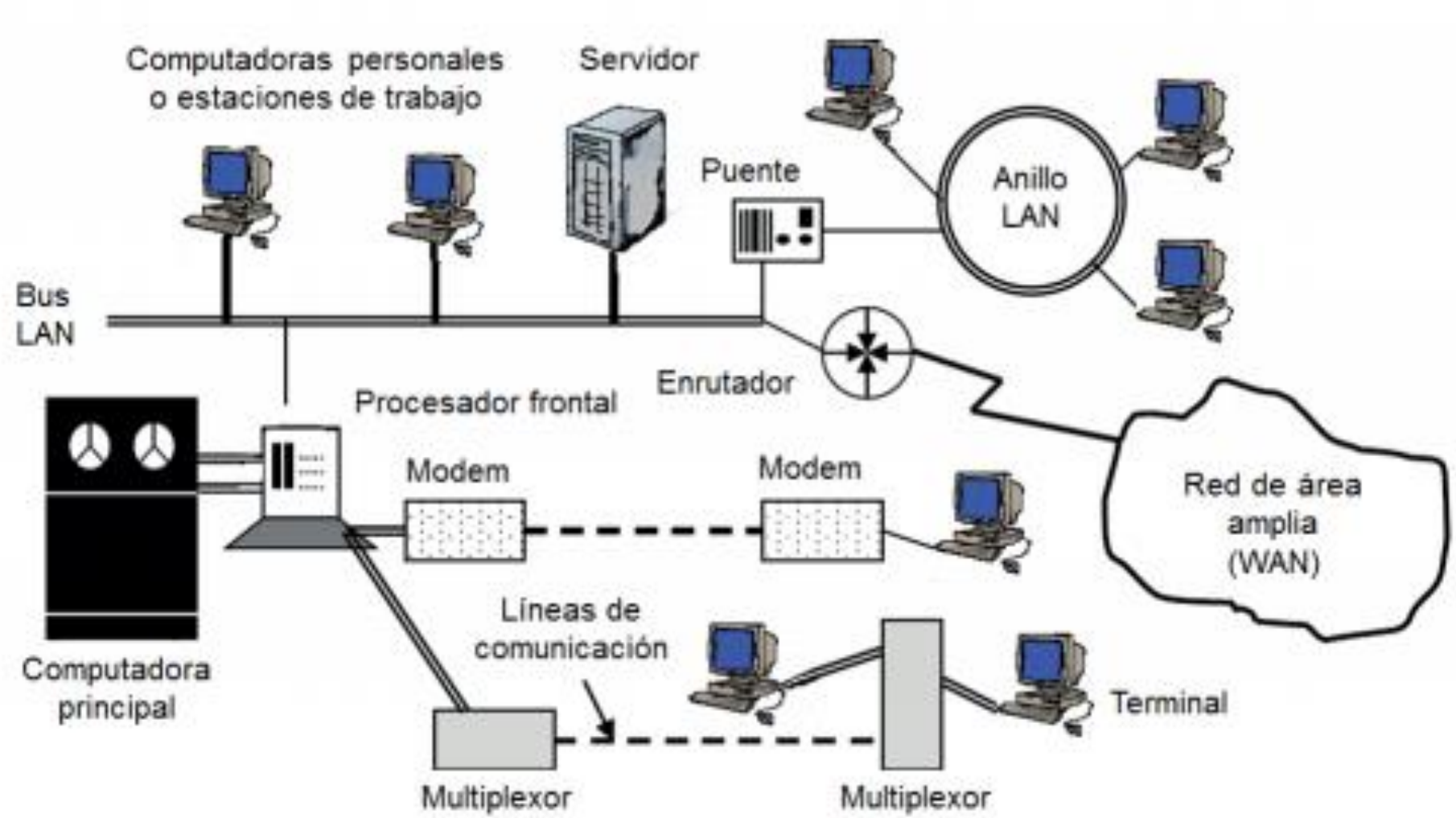
Por: Emmanuel Isaí Chablé Collí

Sistemas distribuidos

Sistemas cuyos componentes **de hardware y software**, que están en **computadoras** conectadas en **red**, se **comunican y coordinan** sus acciones mediante el paso de **mensajes**, para el logro de un objetivo. Se establece la **comunicación** mediante un **protocolo** preestablecido.

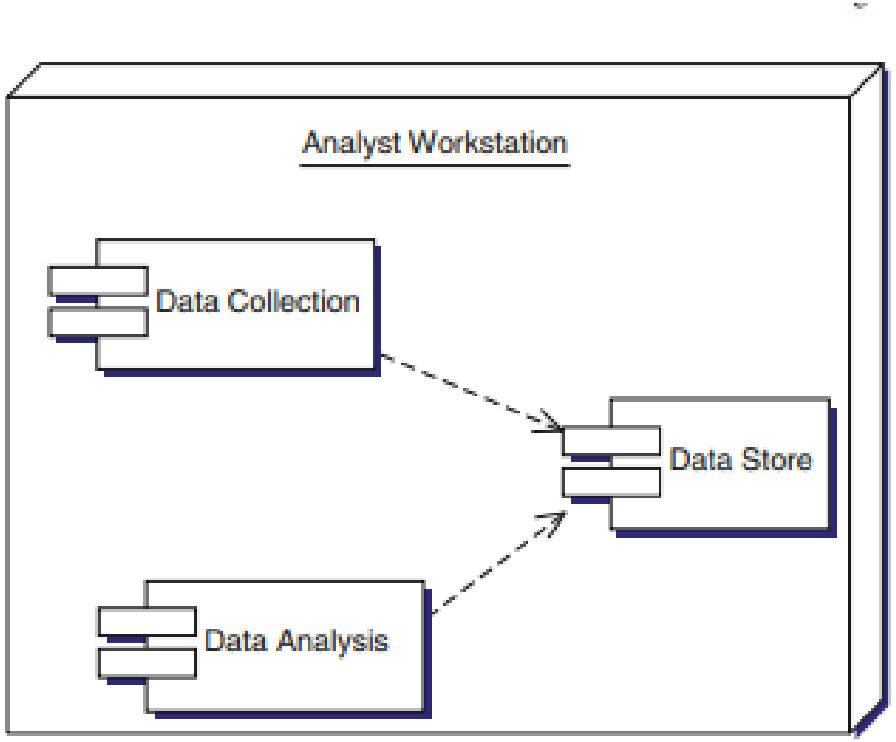


Partes


Computadoras personales
o estaciones de trabajo
Servidor
Puente
Anillo
LAN
Bus
LAN
Enrutador
Procesador frontal
Modem
Modem
Red de área
amplia
(WAN)
Líneas de
comunicación
Computadora
principal
Terminal
Multiplexor
Multiplexor

Arquitectura de software

Estructura o estructuras de un sistema, las cuales están conformadas por elementos (componentes), donde estos muestran sus propiedades externas y las relaciones entre ellos. [L.Bass, P.Clements, R.Kazman, Software Architecture in Practice (2nd edition), Addison-Wesley 2003]



Pipes and filters

Caracteristicas:

- Transformaciones sucesivas de datos

Componentes:

- Filtros: Transforman datos de entrada a datos de salida
- Tuberías: Conector que transforma los datos de salida de un filtro a datos de entrada de otro filtro.

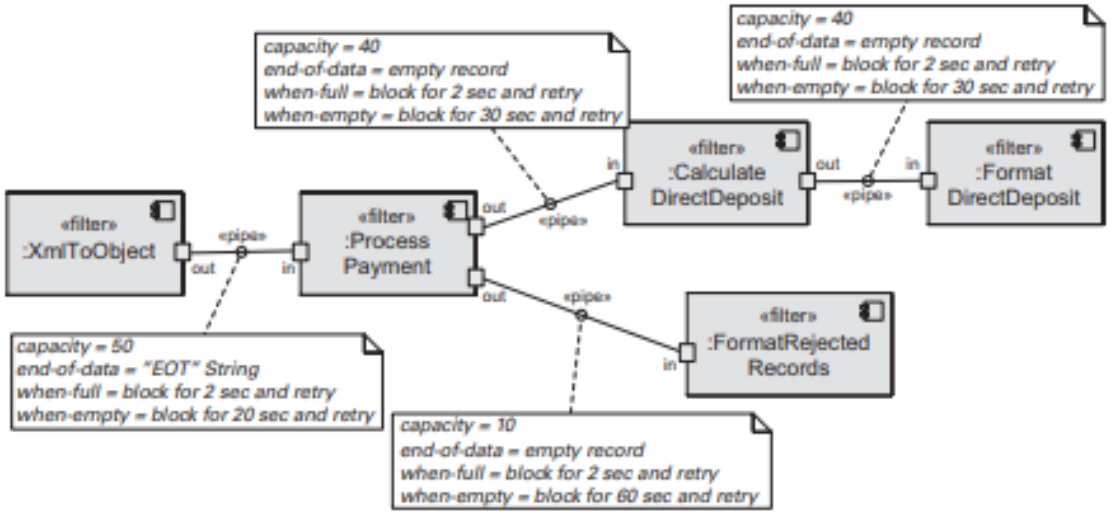


FIGURE 13.8 A UML diagram of a pipe-and-filter-based system

Ejemplos:
Proceso de pago
Analizador de texto

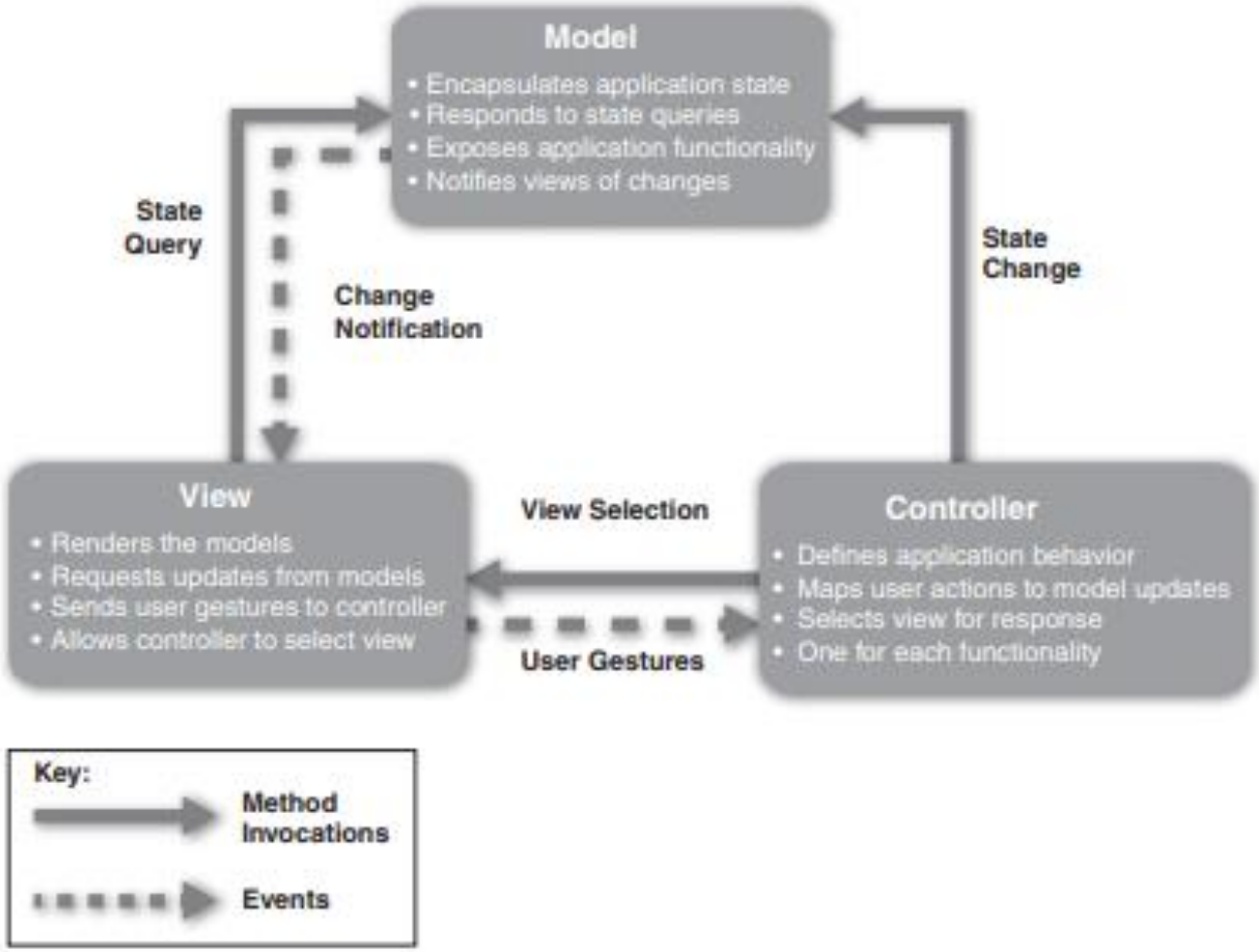
MVC

Caracteristicas:

- Separa de lógica de negocios de la interfaz grafica

Componentes:

- Modelo
- Vista
- Controlador



Cliente Servidor

Caracteristicas:

- Múltiples recursos y servicios compartidos distribuidos entre clientes
- Promover reusabilidad y modificabilidad

Componentes:

- Cliente: Invoca servicios
- Servidor: Provee servicios
- Peticiones/Respuestas: Conector de datos que funcionan mediante protocolos de Peticiones/Respuestas.
- Acoplamiento: Relación entre clientes y servidores.

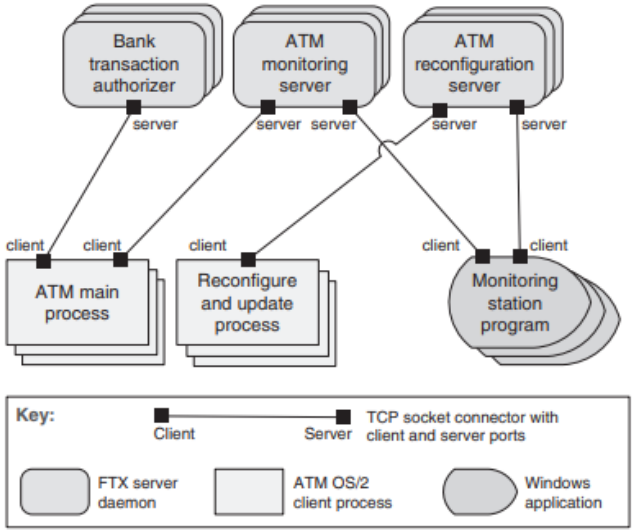
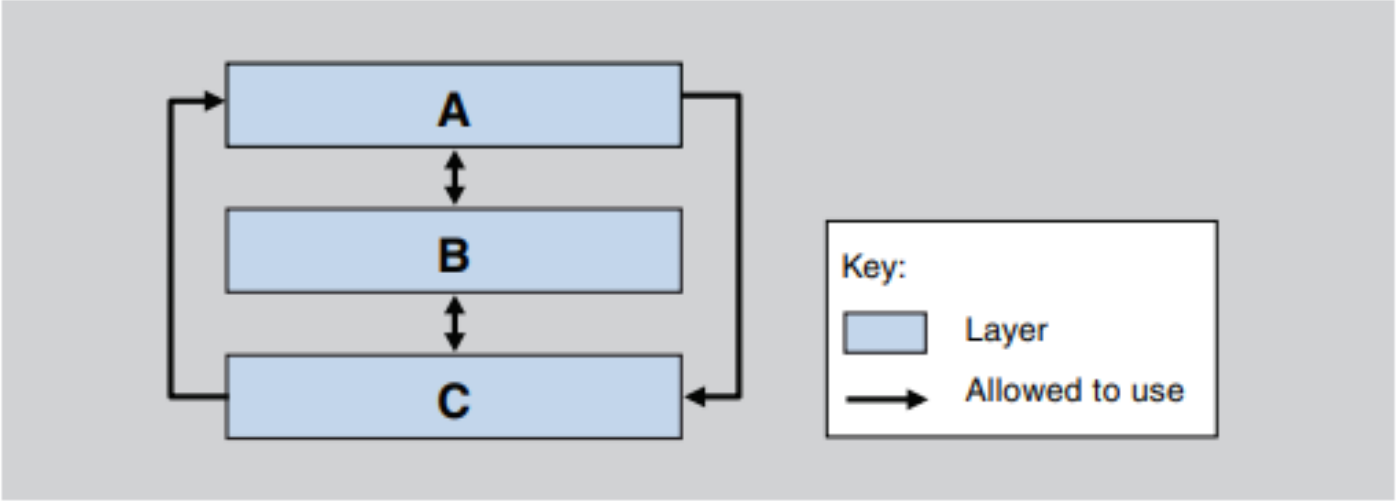


FIGURE 13.9 The client-server architecture of an ATM banking system

Capas

Caracteristicas:

- Módulos que contienen diferentes componentes los cuales están relaciones entre si
- Las capas se pueden relacionar y comunicarse con otras.



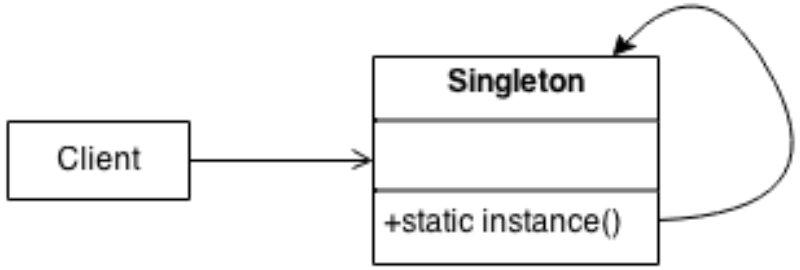
Patrones de diseño

En la ingeniería de software, un patrón de diseño es una **solución general repetible** a un problema común en el diseño de software. Un patrón de diseño **no es un diseño terminado** que se puede transformar directamente en código. Es una **descripción o plantilla** sobre cómo resolver un problema que se puede utilizar en muchas situaciones diferentes.

Singleton

Caracteristicas:

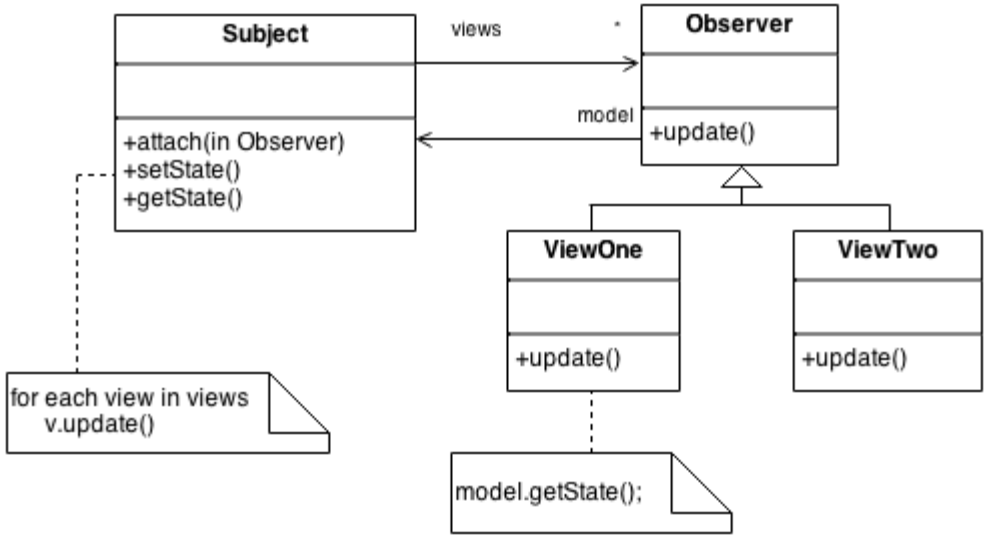
- Asegura que la clase solo tenga una instancia y provee un punto global a esta.
- Encapsula la inicialización o el proceso de construcción.



Observer

Caracteristicas:

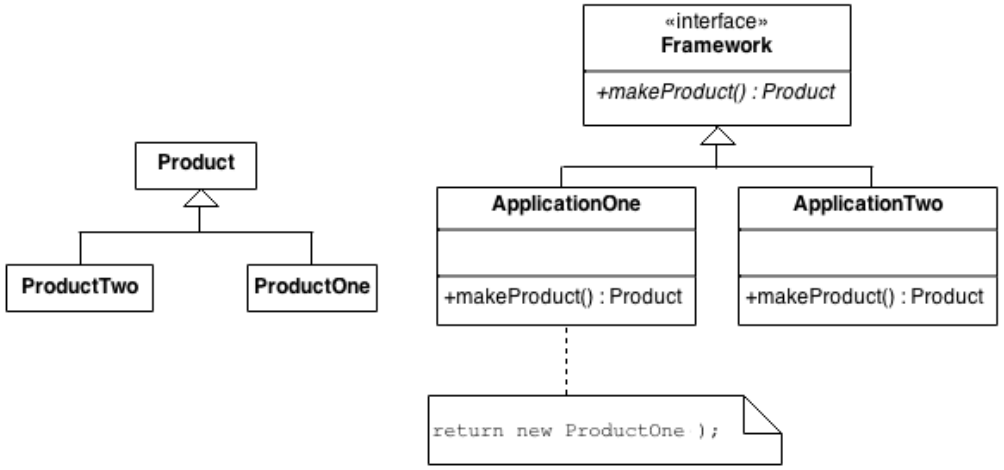
- Define dependencias entre uno y mas objetos donde al percibir un cambio de esta notifica a cada de los objetos dependientes.
- Encapsula los componentes centrales y proporciona una jerarquía de observadores.
- Puede usarse para ser la “Vista” del MVC



Factory

Caracteristicas:

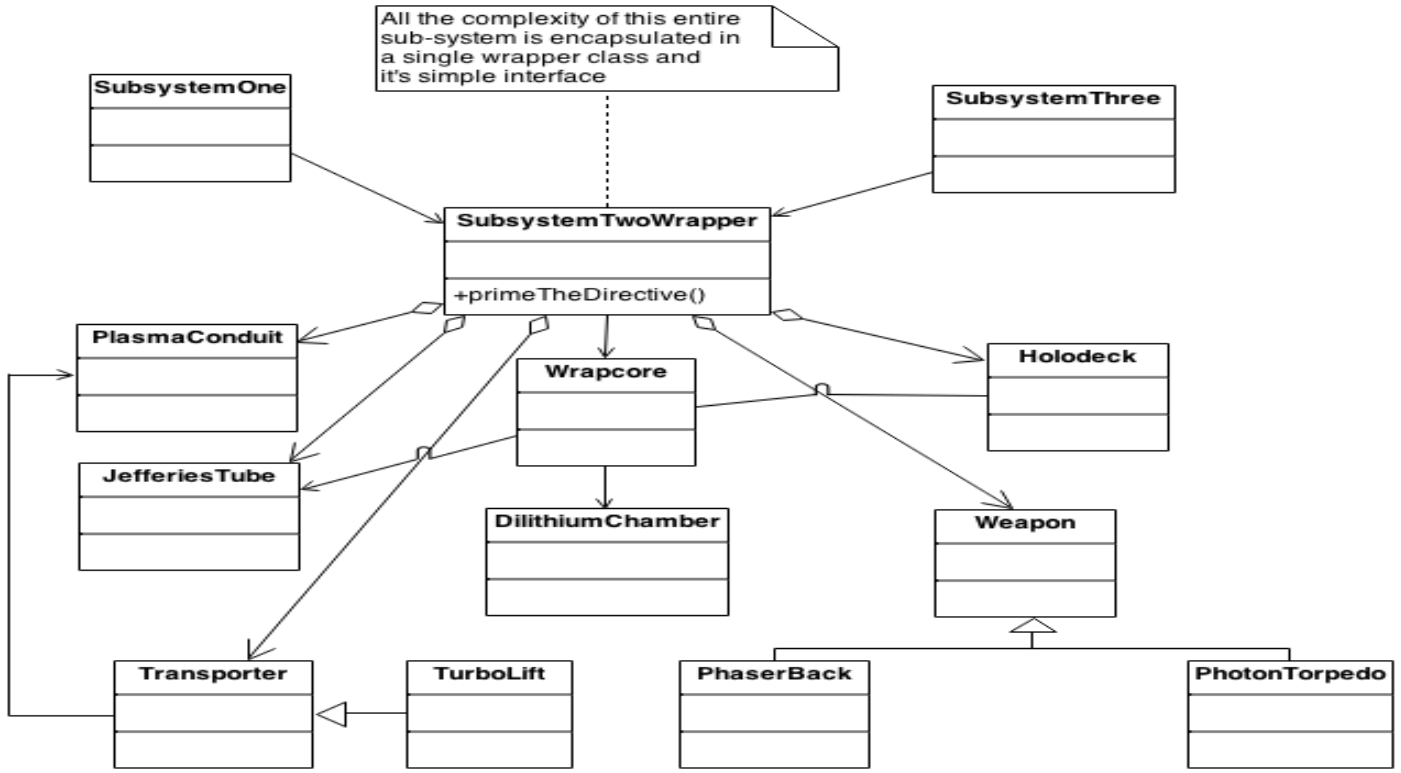
- Se encarga de crear múltiples instancias de una clase sin necesidad de llamar al constructor de esta.
- Define una interfaz para la creación de un objeto donde cada subclase se encarga de generar una instancia especifica de ese objeto.



Facade

Caracteristicas:

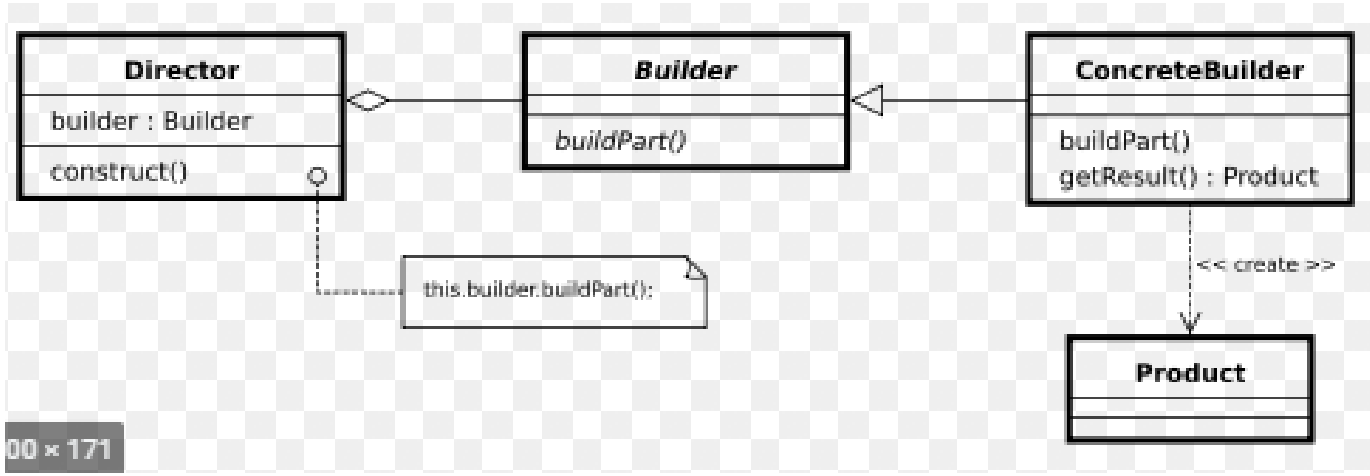
- Provee una interfaz la cual es un punto de acceso hacia un subsistema, define a un alto nivel interfaces que puedan usarse fácilmente.
- Encapsula un subsistema complejo con una simple interfaz.



Builder

Caracteristicas:

- Separa la construcción de un objeto completo de su representación, pero mantiene el mismo proceso de construcción, y permite crear diferentes representaciones de los objetos resultantes.
- Separa diferentes partes de su representación, para crear diferentes objetos complejos.



Diferencias entre patrón de diseño y arquitectónico

- Los patrones arquitectónico definen un **comportamiento general** de todo un sistema entero, el cual es de muy **alto nivel** lo que significa que no puedes codificarlo simplemente observando un diagrama de arquitectura.
- Los patrones de diseño son **plantillas** para resolver problemas específicos de **bajo nivel** lo que quiere decir que están más cerca de ser codificados.

Referencias

- Bass, L., Clements, P., & Kazman, R. (2012). Software Architecture in Practice. PEARSON Education.
- Lopez Fuentes, F. (2015). Sistemas distribuidos. México.
- sourcemaking.com. (2007). Obtenido de sourcemaking.com: https://sourcemaking.com/design_patterns