Projecte de Programació

Arquitectura 3 capas

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas

La arquitectura del *software* describe los sistemas y componentes computacionales del *software*, y las relaciones entre ellos

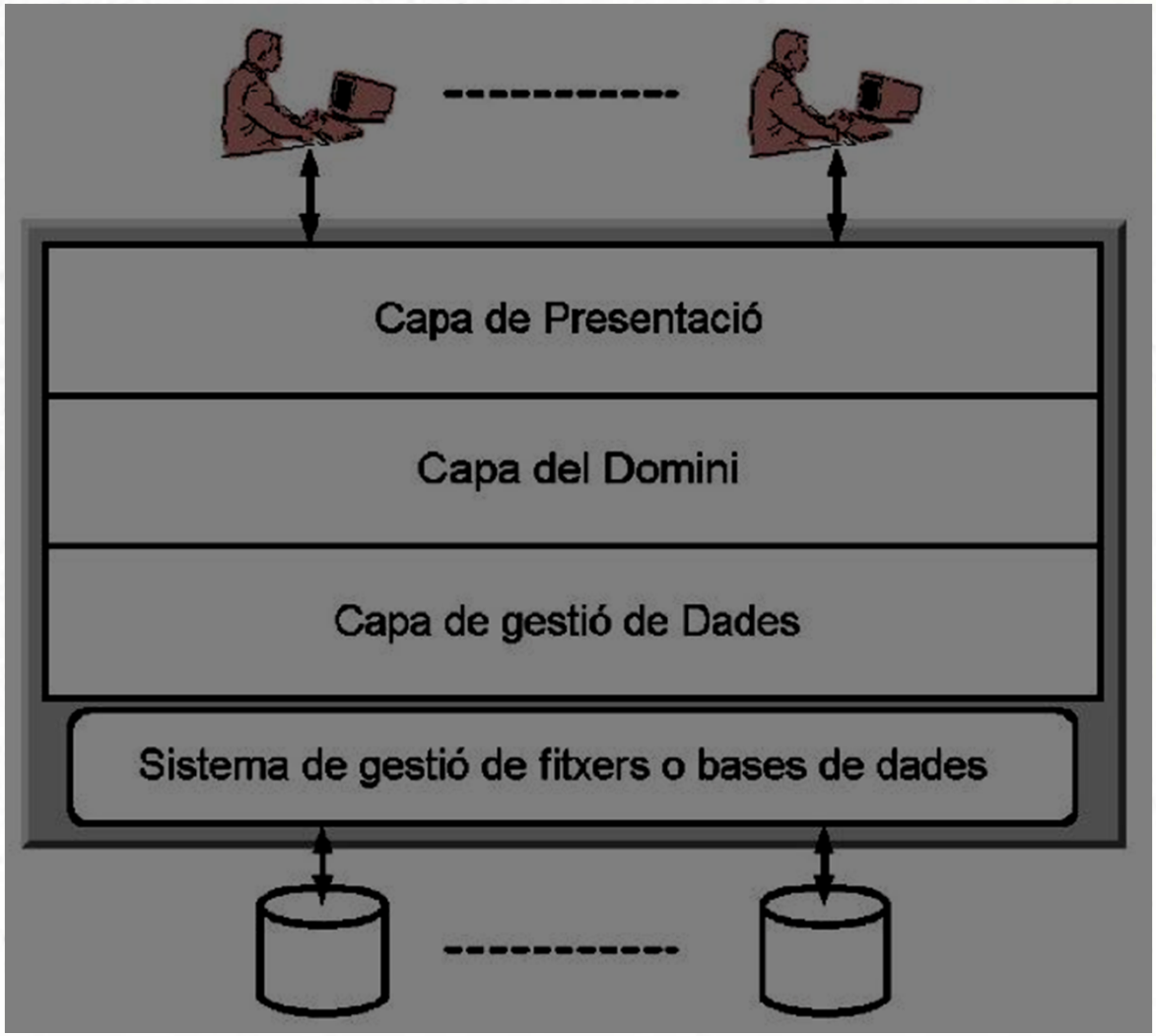
Existen diferentes *patrones arquitectónicos* en los que podemos basarnos para diseñar nuestro sistema. En PROP usaremos la **arquitectura en tres capas**.

Consta de:
- Capa de *Presentación*
- Capa de *Dominio*
- Capa de *Gestión de Datos* (o de *Persistencia*)

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas

Esencialmente es:



Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas

Capa de Presentación: Responsable de la interacción con el usuario

- Presenta los datos al usuario pero no sabe hacer las transformaciones necesarias para poder obtener los resultados que el usuario quiere

- Se relaciona con los usuarios capturando los acontecimientos y presentando respuestas y resultados

- Se relaciona con la capa de dominio pasándole las peticiones externas y recogiendo los resultados que hay que presentar

- Gestiona la interfície de comunicación con el usuario (incluye comprobación de sintaxis de los datos, NO de sus valores)

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas

Capa de Dominio: Contiene el núcleo del programa. Es la capa que sabe transformar y manipular los datos del usuario en los resultados que espera (la capa no "sabe", sin embargo, ni de dónde vienen estos datos ni dónde están almacenados)

- Se relaciona con la capa de presentación, recibiendo las peticiones externas y retornando los resultados deseados

- Se relaciona con la capa de persistencia, pasando las operaciones de consulta y modificación de los datos, y recibiendo las respuestas y resultados

- En general, el estado del dominio se mantiene y cambia en esta capa

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas

Capa de Persistencia: Esta capa es la responsable de almacenar los datos. Sin embargo, ignora cómo tratarlos

- Se relaciona con la capa de dominio recibiendo las operaciones de consulta y actualización de los datos, y retornando respuestas y resultados de estas peticiones

- Se relaciona con el sistema de gestión de bases de datos o ficheros pasando las operaciones de consulta y/o actualización de los datos en el formato y lenguaje adecuados y recibiendo las respuestas y resultados

- Permite que determinados objetos del dominio sean persistentes y que el dominio ignore dónde se encuentran los datos

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas: Controladores

Cada capa tiene responsabilidades cualitativamente diferentes:

- Presentación
  - Interacción con el usuario
  - Control de las peticiones del usuario
  - *Comunicación con la capa de dominio*

- Dominio
  - Mantenimiento básico de los datos (clases definidas en la especificación)
  - Implementación de una parte de las funcionalidades principales (casos de uso) que corresponde a la capa de dominio: cálculos, modificación del estado del sistema, etc.
  - *Comunicación con las capas de presentación y persistencia*

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas: Controladores

La tarea de comunicación entre capas es parte esencial de las responsabilidades de cada capa, principalmente en las capas de Presentación y Dominio

Por ello las clases que forman parte de cada capa se pueden dividir, a grandes rasgos, en:

- Capa de Presentación: Vistas y ***controladores***

- Capa de Dominio: Clases del modelo y ***controladores***

Así pues, qué son los controladores?

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas: Controladores

Los controladores son clases propias de cada capa que tendrán la *responsabilidad* de:

- **Comunicación entre capas**, manteniendo la sincronización entre capas adyacentes

- Implementar los casos de uso y/o **aglutinar funcionalidades de los casos de uso** relacionados

Además, la presencia de controladores dentro de las capas permite **organizar mejor el código**, pensando en la reutilización

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas: Controladores

Controladores de la capa de Presentación :

- Permiten separar el aspecto externo (vista) de los métodos que gestionan el comportamiento de la interfície (controladores)

- Ligados a los casos de uso, es habitual agrupar los casos de uso de una misma familia en el mismo controlador (ej: ABMC de una clase) -> REUTILIZACIÓN (al menos de la estructura del código)

Controladores de la capa de Persistencia :

- No son estrictamente necesarios, pero podrían encargarse de la conversión entre formatos diferentes de la información, por ejemplo.

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas: Controladores

Controladores de la capa de Dominio:

- Permiten descargar de funcionalidades a las clases del modelo, que pueden encargarse más de la gestión de los datos

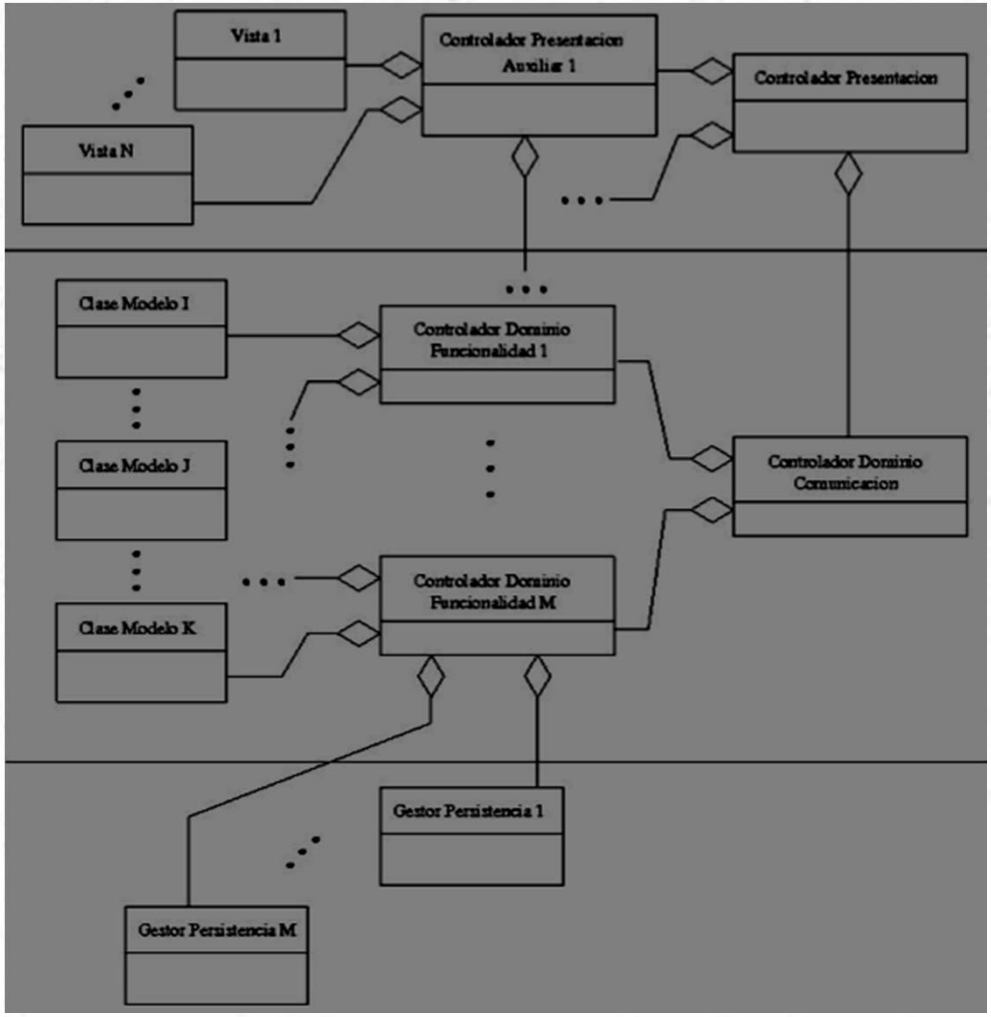
- Permiten que las funcionalidades concretas de la aplicación (los algoritmos ligados a los casos de uso) se implementen dentro de los controladores -> REUTILIZACIÓN (de algoritmos y clases del modelo)

Puede haber uno o más controladores por capa, dependiendo de la **granularidad** que se quiera tener

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas: Controladores

Esquema general:



Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas: Controladores

Este esquema general es una versión muy GENERAL: No hace falta que tengamos tantos controladores

En particular, Controlador Dominio Comunicación si hay mucha carga de ello en el proyecto. Si no, Controlador de Presentación puede usar directamente los controladores de dominio

La manera de distribuir los controladores en cada capa **NO es única**, pero suele ir guiada por los casos de uso: un controlador de la capa de dominio integrará tantas clases del modelo como necesite, junto con los gestores de disco correspondientes, para implementar los casos de uso asociados

Suele haber un gestor de persistencia por cada clase persistente del modelo (aunque se pueden agrupar varias clases en uno)

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas: Controladores

Así pues, un programa principal (aquel que pone en marcha la aplicación, *usualmente el controlador de presentación*) podría seguir un esquema similar a:

- El controlador de presentación, al crearse/inicializarse, crea una instancia de controlador de dominio y del resto de controladores de presentación (si los hay)

- El controlador de dominio, al crearse/inicializarse, crea una instancia del resto de controladores de dominio (si los hay)

- El resto de controladores de presentación crean una instancia de sus vistas asociadas

- El resto de controladores de dominio crean las clases del modelo que necesiten

No es necesario hacerlo todo al principio, puede hacerse bajo demanda

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas: versión PROP

- La comunicación (relaciones entre clases y mensajes entre objetos) sólo se produce **entre elementos de la misma capa o entre capas adyacentes**.
- Si es entre capas adyacentes, la comunicación se ha de hacer vía controladores. Única excepción: podemos pasar un controlador de dominio como parámetro a una vista para que ésta trabaje directamente con él

- La comunicación entre capas, al realizarse entre controladores, se hace vía tipos de datos generales (por ejemplo, vectores de *Strings*), NUNCA vía instancias de datos del modelo -> puede ser una buena idea añadir en las clases del modelo 2 operaciones:
  - *vector<String> toString()*
  - *constructor (vector<String>)*

- El esquema conceptual de los datos se implementa en la capa de dominio: **patrón DOMAIN MODEL** (versus patrón Transaction Script)

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas: Ventajas

- *Intercambiabilidad*: Un cambio en la interfície del programa no afectará al resto, sólo la capa de presentación. Un cambio en algoritmos o estructuras de datos sólo afectará a la capa de dominio. Un cambio en la representación de los datos (cambio en el SGBD, o entre BD y ficheros) sólo afectará a la capa de persistencia

- *Reusabilidad*. Cualquier capa se puede reusar "fácilmente"

- *Portabilidad*. La capa de dominio, que encapsula la lógica del programa, es bastante independiente de cambios de plataforma, sistema operativo, etc.

Fase de Diseño - Arquitectura 3 capas

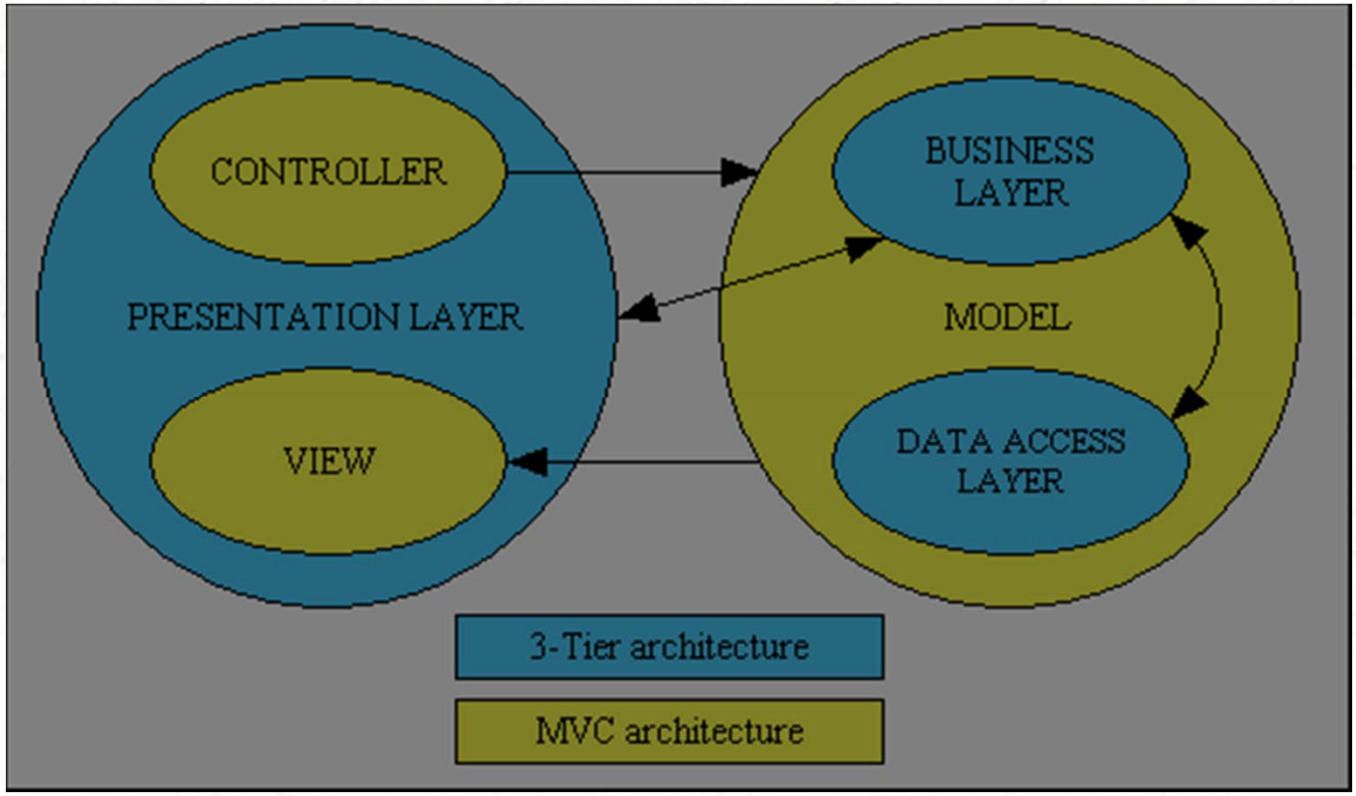
Arquitectura en tres capas: Inconvenientes

- *Ineficiente*: Demasiados mensajes, estructura rígida

- *Redundante*: A veces se hace lo mismo (o similar) en diferentes capas. Por ejemplo, validar que un campo está dentro del rango correcto

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas y MVC

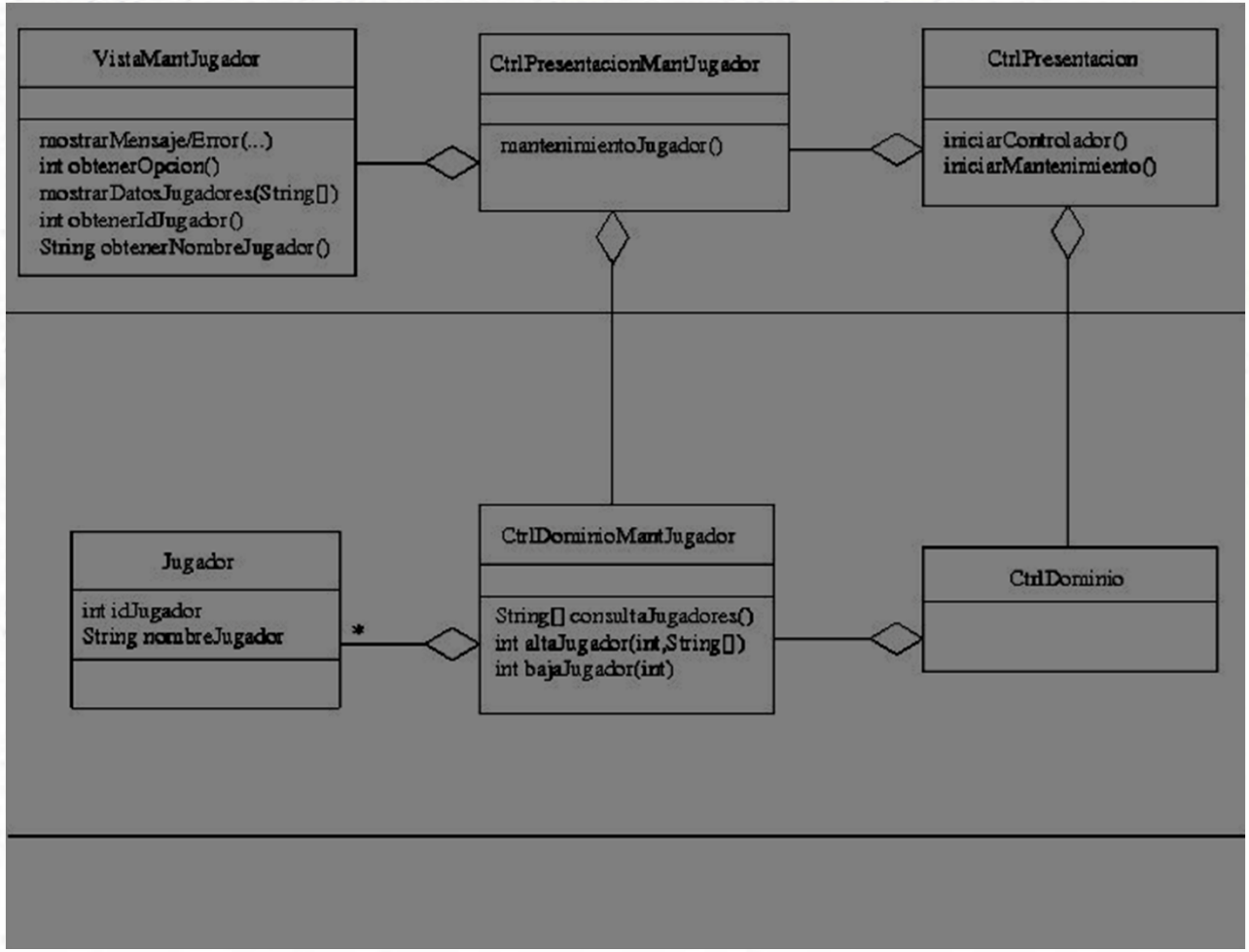


(figura 8 de http://www.tonymarston.net/php-mysql/3-tier-architecture.html)

Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas: Ejemplos

Mantenimiento Jugador



Fase de Diseño - Arquitectura 3 capas

Arquitectura en tres capas: Ejemplos

Mantenimiento Genérico