SOFTWARE ARCHITECTURES

Chapter 3

Software Engineering
Computer Science School
DSIC – UPV

Goals

- Introduce the concept of Software Architecture
- Describe the main features of distributed systems, in particular the multi-layered architecture.

Contents

1. INTRODUCTION

Programming in the small/medium/large

Modules

Classes

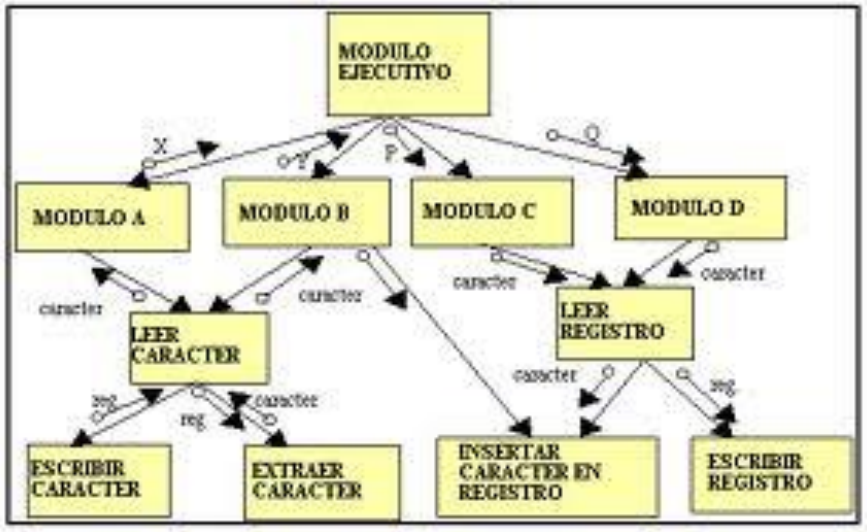
Programming in the small/medium/large

- When systems grow in size it is required an organization in terms of subsystems so that they are manageable

- Throughout history of software development different strategies to manage complexity, usually related with design at different levels of abstraction, have been used

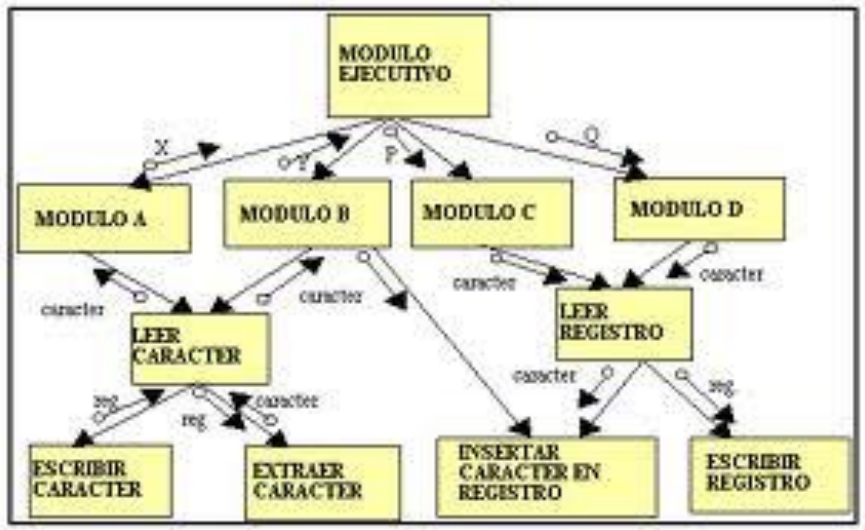
Structured Methods

- Structure diagram
  - Based on the notion of module (Parnas, 1972)
  - A system is partitioned in modules that invoque or provide service to other modules, possibly with data passing in both directions



Module

- Part of a program that implements part of the functionality
- Characteristics:
  - White/Black box
  - Encapsulation



- A module may be decomposed in terms of other modules of a lower level

Modular Architecture

- Preliminary Design: Structuring the system in terms of modules
  - Building structure diagram
  - Module = black box
- Detailed design:
  - Description of process that are implemented by modules
  - Module= white box

Object Oriented Architetures

- Classes as decomposition units
  - Structure + behavior in a module
- The structure of classes is propagated to the code
  - New classes are incorporated when lowering the abstraction level
- Packages as a way of grouping classes in self-contained componets
  - A Java package is a mechanism to organize classes that may be reused if a '.jar' file is created that may be imported in another project. It is called a Java component and it may contain classes and other Java components or libraries

Problems

- Approaches based on modules and objects are low level ones.
- They do not divide the application in terms of functional blocks but they are mere groupings of code

- A more abstract mechanism is necessary to clearly detect the aspects that are present in most software systems

SOFTWARE ARCHITECTURES

Client-Server

Multi-Layered

What do we mean with "Software Architecture"?

The ***software architecture***, has to do with the design and implementation of high level structures. It is the outcome after assembling a number of different architectural elements in order to adequately satisfy both functional and non functional requirements such as trustability, scalability, portability and availability.

Kruchten, Philippe

Software architecture is important

- In the description phase of the **Software Architecture** the system must be organized in terms of **subsystems.**

- Many times the architecture is based on other similar previously developed systems by means of **architectonic patterns**.

- Some interesting patterns in information systems are: interactive systems, multi-layered systems, **distributed systems**, real time systems, etc.

Types of Systems (non exhaustive list…)

- **Distributed Systems**:

  A software system in which information processing is distributed among different computing nodes.

- **Personal Systems**:

  Non distributed systems that are designed to be run in a personal computer or workstation.

- **Embedded Systems**:

  Information systems (hardware + software), usually real-time ones integrated in a more general engineering system that perform functions of control, processing and/or monitoring.

Distributed Systems Architectures (non exhaustive)

- **Multi-processing architectures:**

  The system consists of multiple processes that may or may not be run in different processors.

- **Client/Server architectures**:

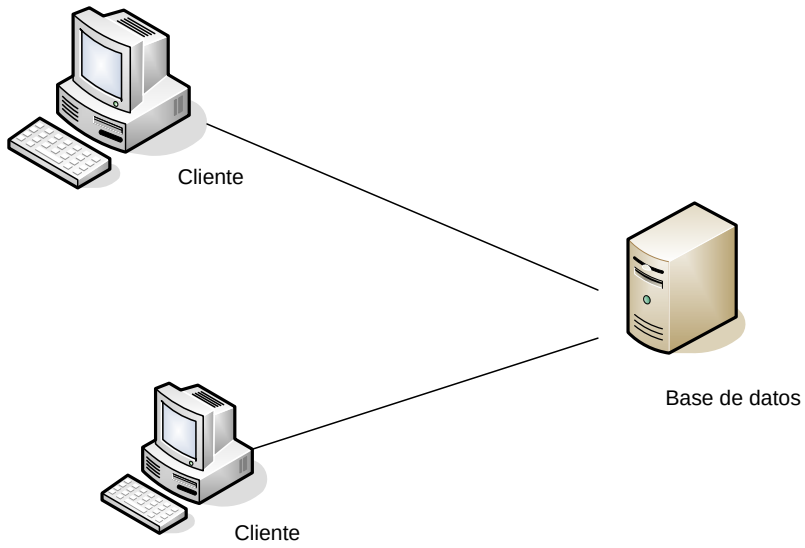
  The system is seen as a set of services that are provided to client applications by server applications. Client and server applications are handled separately.

- **Distributed objects architectures**:

  The system is seen as a set of interacting objects whose location is not relevant. There is no distinction between a provider of a service and a consumer.

Client Server Architecture

- C/S divides an application into 2 components which are run in 1 or more devices:
  - The server (S) is a service provider.
  - The client (C) is a consumer of services.
- C and S interact by means of a message passing mechanism:
  - Service request.
  - Answer.



Multi-Layered Architecture

A **layered system** is a sorted set of subsystems each one defined in terms of the ones located below them and providing the implementation base of the systems above.

- The objects in each layer may be independent (recommended) although there use to be some dependencies between objects of different layers.

- There is a relationship **client/server** between the lower layers (providing services) and the upper layers (using those services).

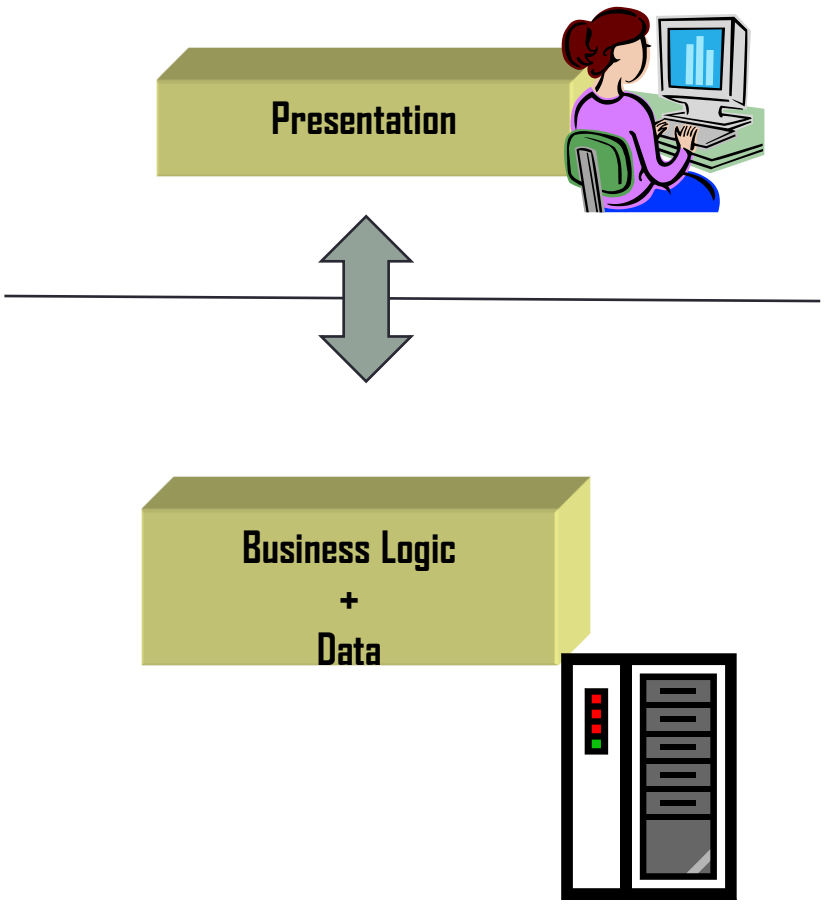
Multi-Layered Architecture

Layered architectures may be **open** o **closed** depending on the dependencies between layers.

- **open**: a layer may use characteristics of any layer.
- **closed**: a layer may only use characteristics of its adjacent lower layer.

It is recommended to use **closed** architectures, because there are fewer dependencies between layers and because it is easier to apply changes because the interface of a layer only affects to its immediate upper layer.

2 – layers Architectures: Thin clients

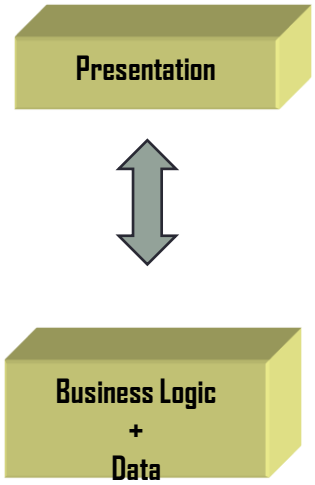


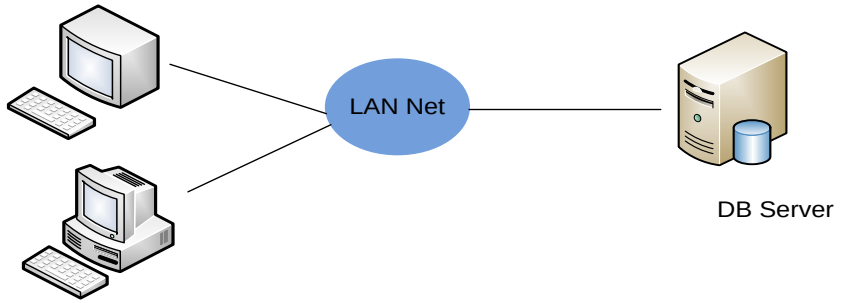
Useful for:

- Legacy systems in which the separation between processes and data management is not feasible
- Data intensive applications (queries and navigation on a DB) with little processing

Layers versus Tiers

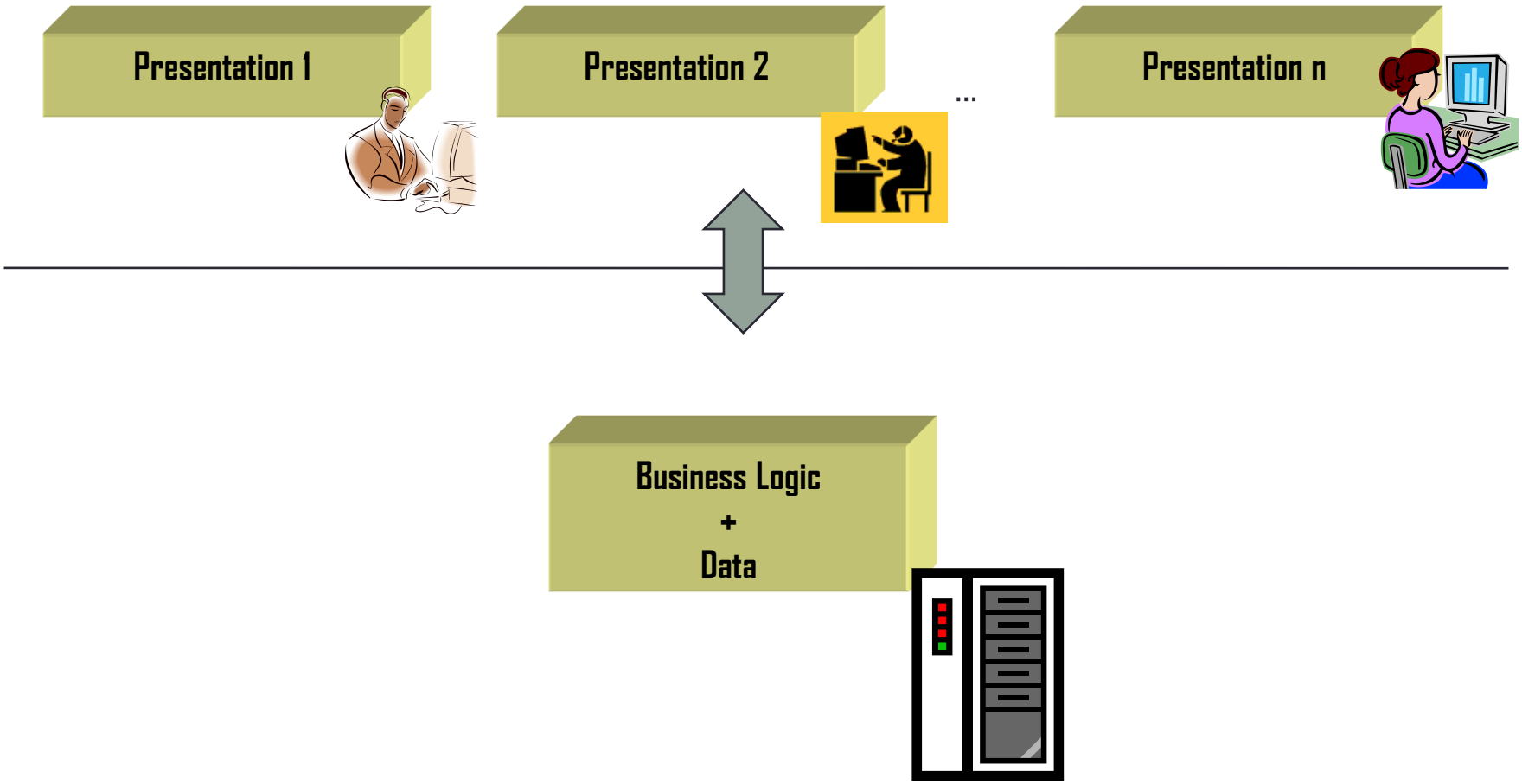
- Layer refers to a logical segmentation of the solution whereas tier refers to a physical segmentation or location.





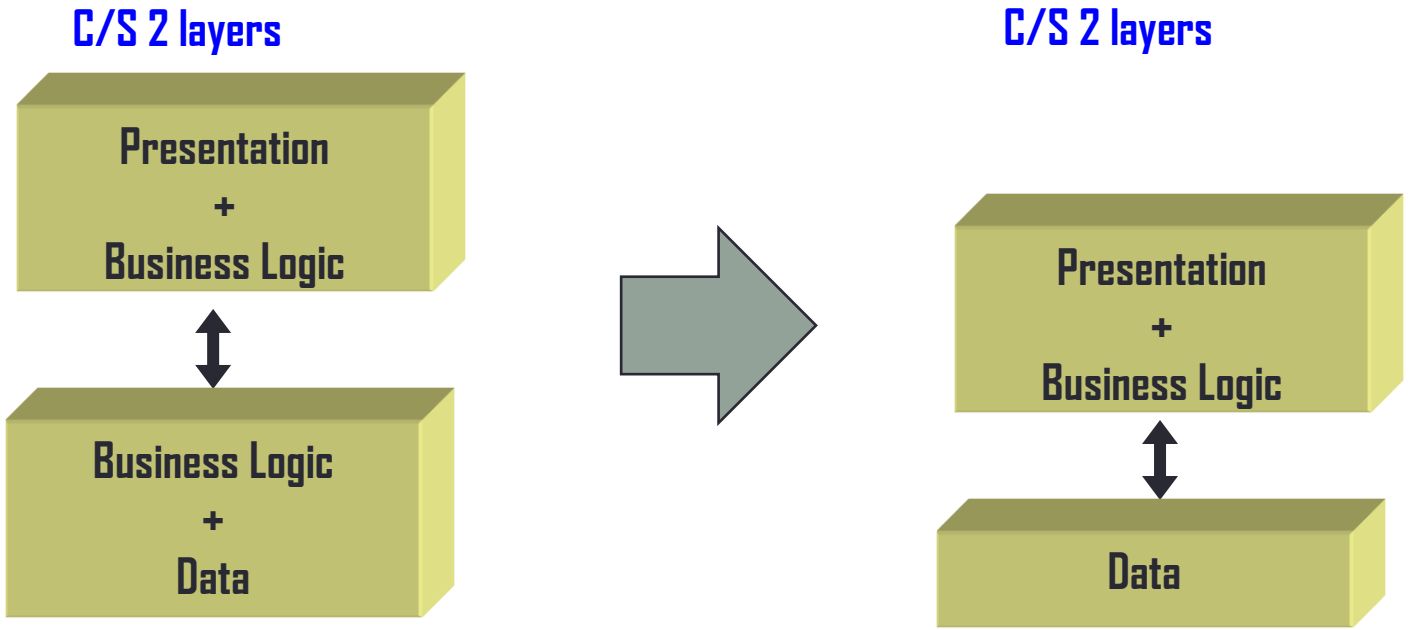
Architecture with 2 layers: Thin clients

1 Application – N platforms:



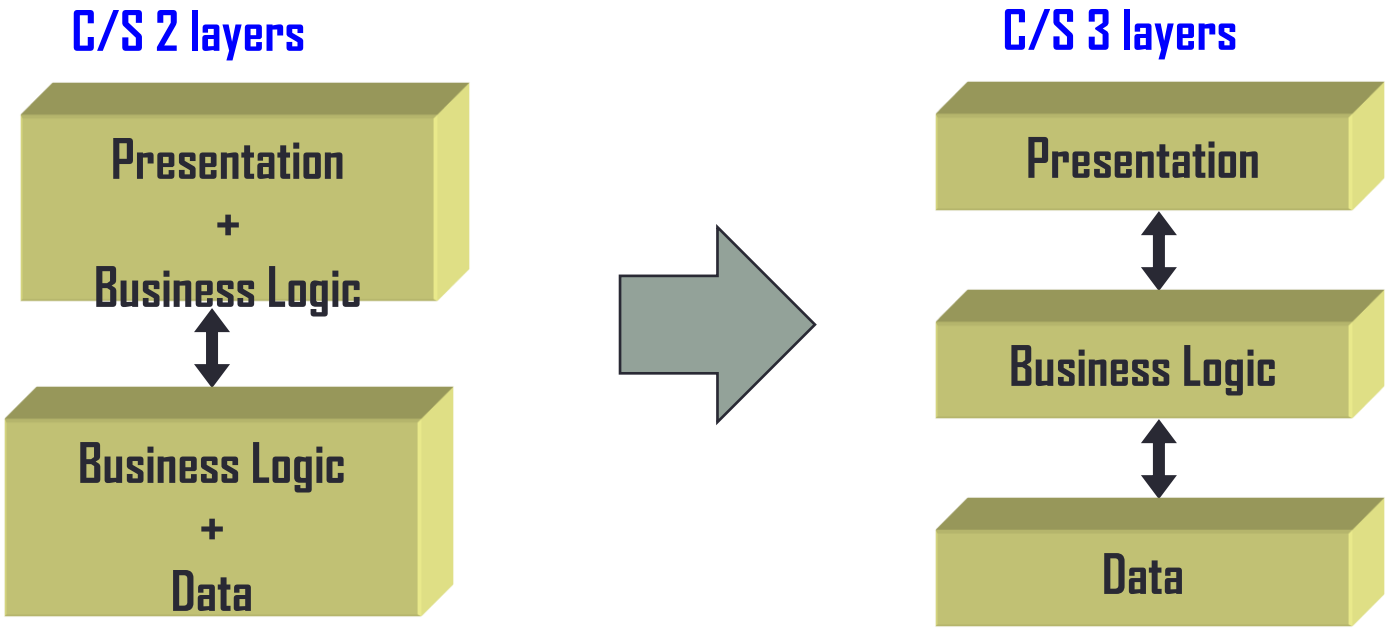
Architecture with 2 layers: Fat clients

Part of the logic (e.g. validations, business rules) is moved to the client



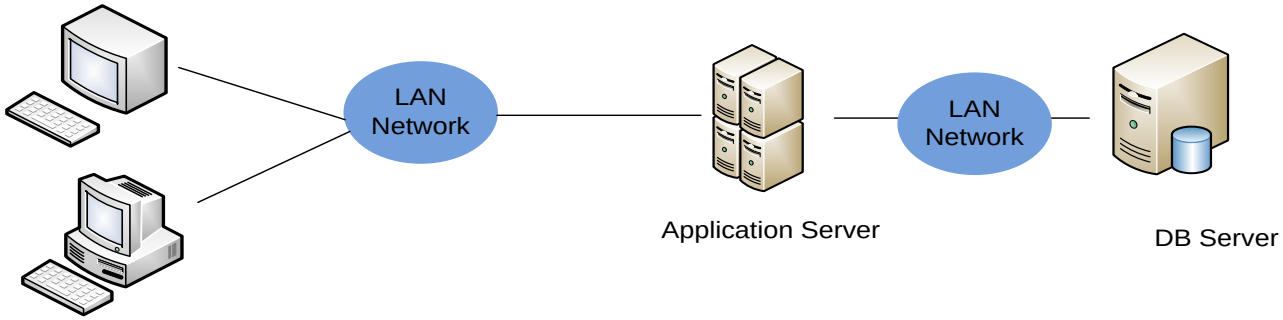
(Excel + Access)

Solution: 3 layers

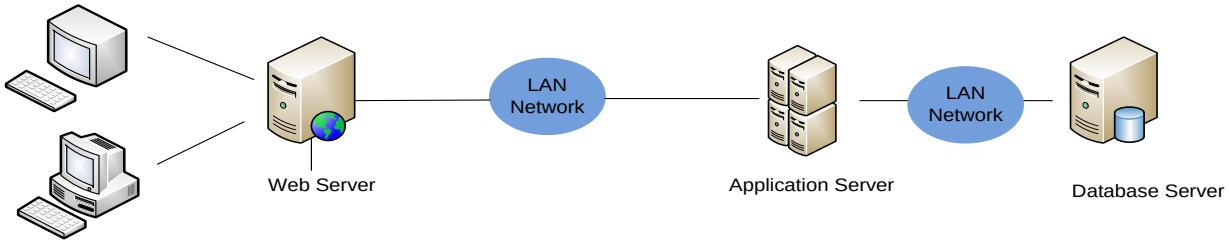


Client Server 3/N-tiers

- Architectures 3-tiers

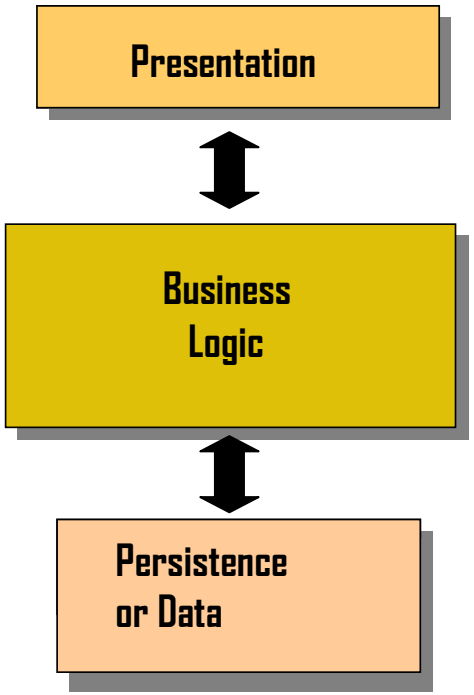


- Architectures 4-tiers



3-layered Architecture

(Generic)

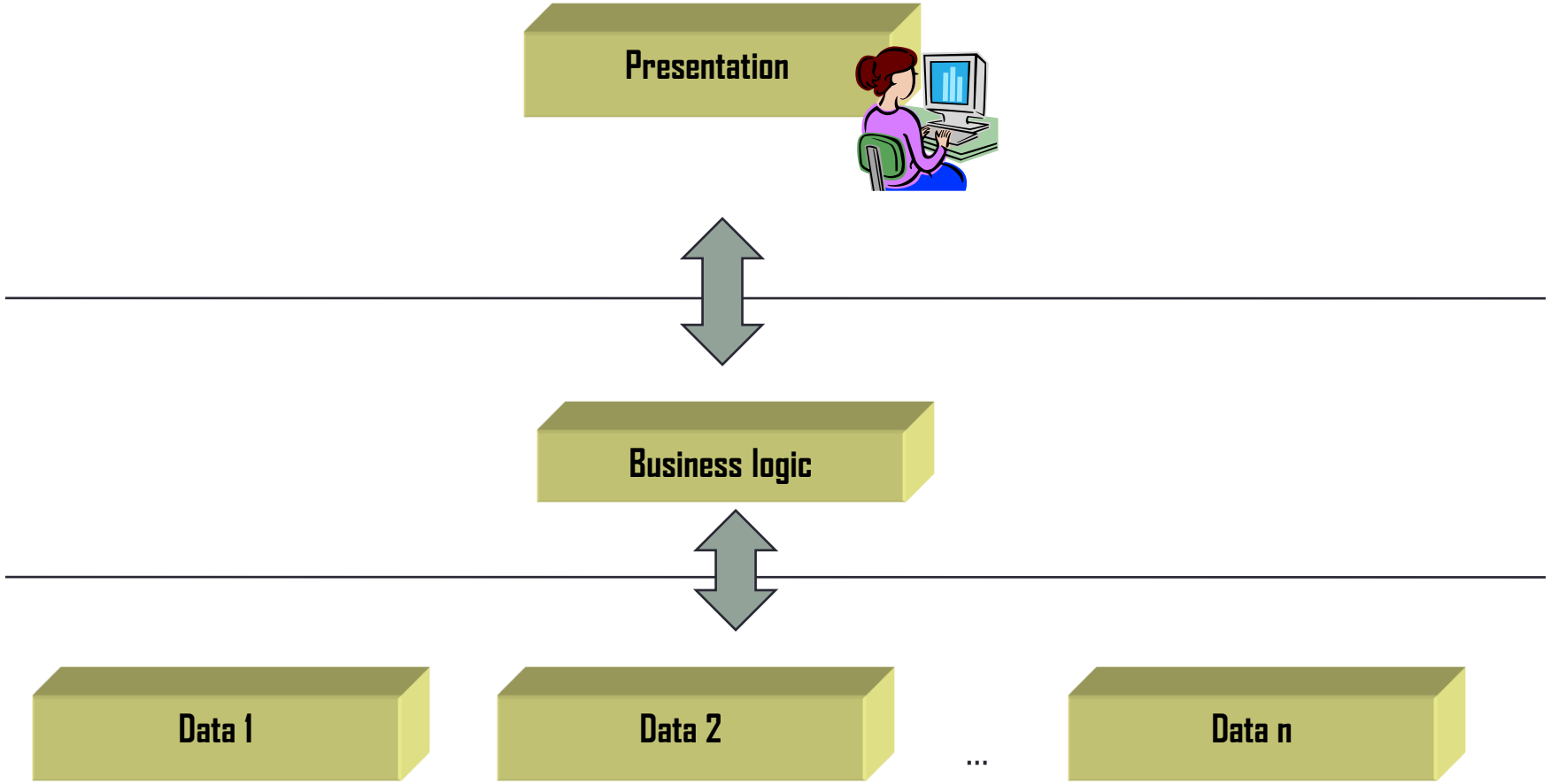


- **Presentation**
  - Presentation of computation results to the user and user input detection.
- **Business Logic**
  - Provide the functionality of the application
- **Data**
  - Provide persistence to data by means of databases or files…

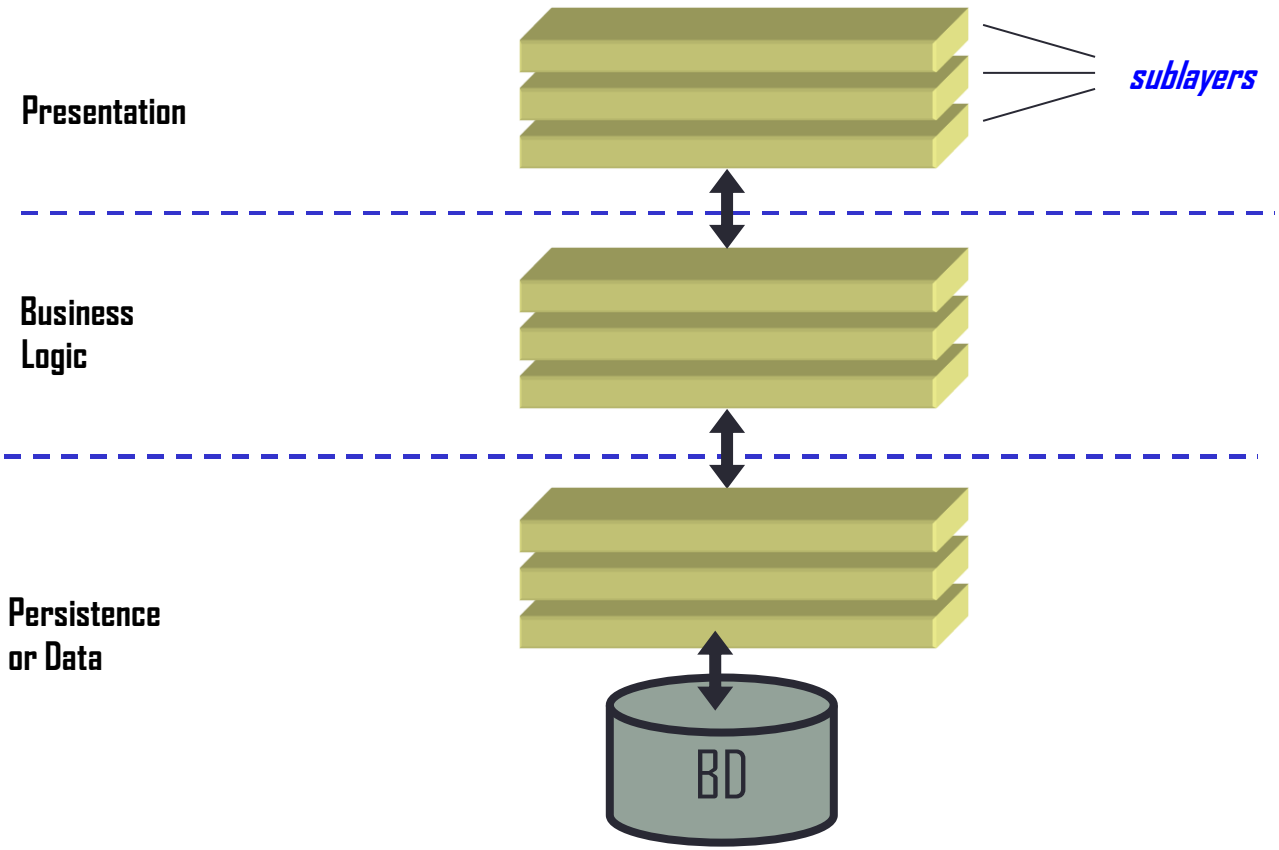
Advantages

- Isolate business logic in a separate component.
- Distribution of layers in different machines or processes.
- Possible parallel development.
- Assigning resources to each layer.
- SOFTWARE REUSE …

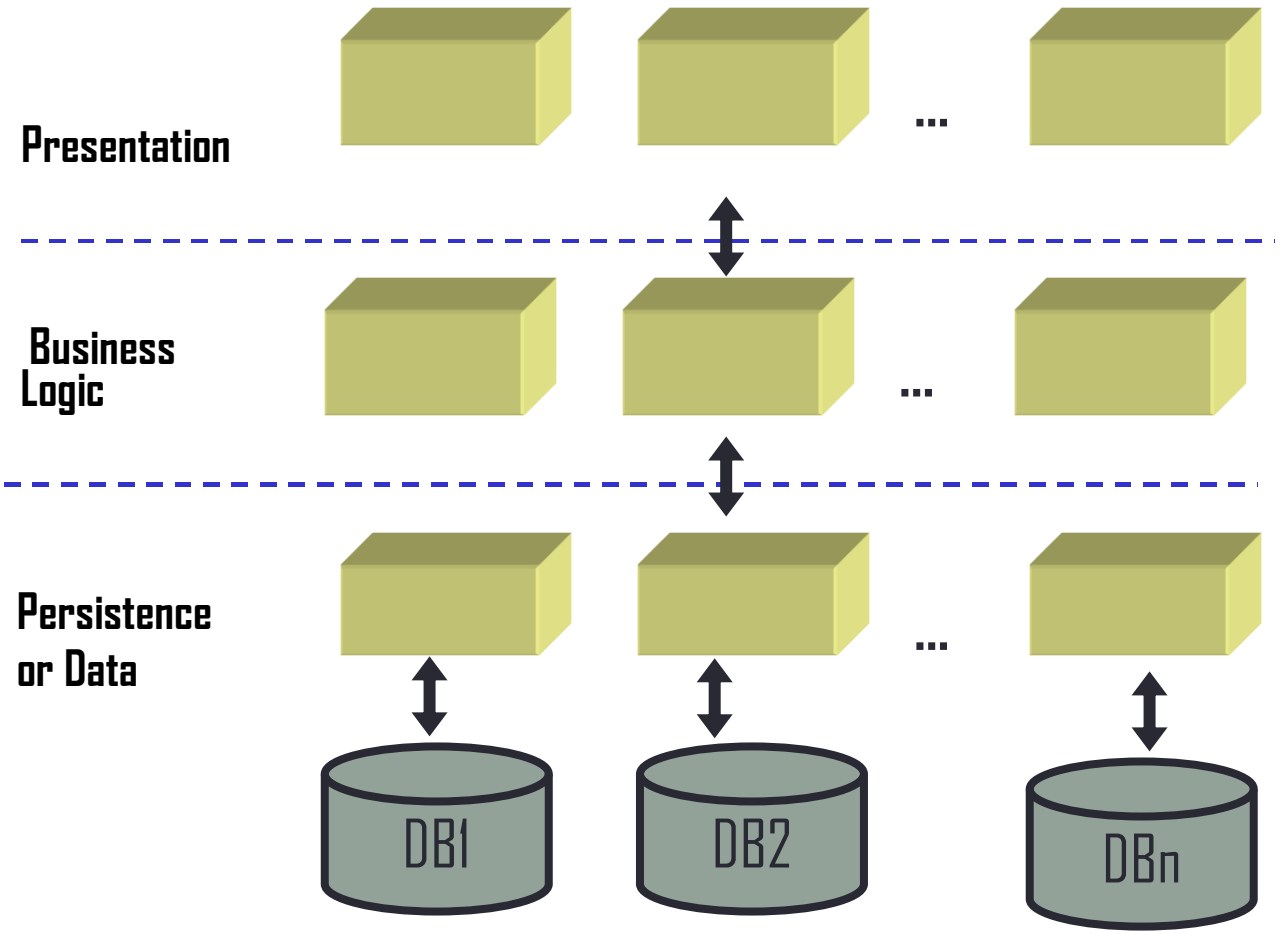
Advantages…



Three layered Architectures: variations



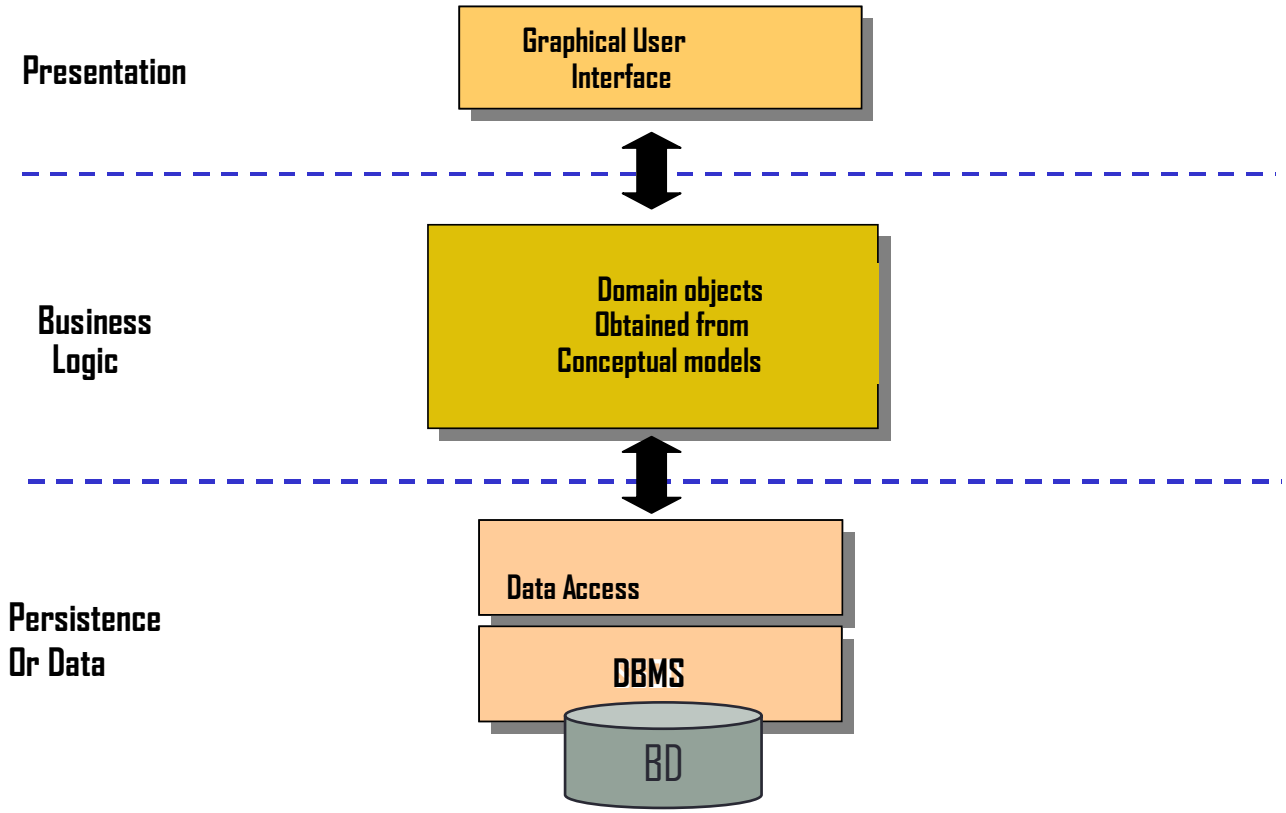
Three layered Architectures: variations



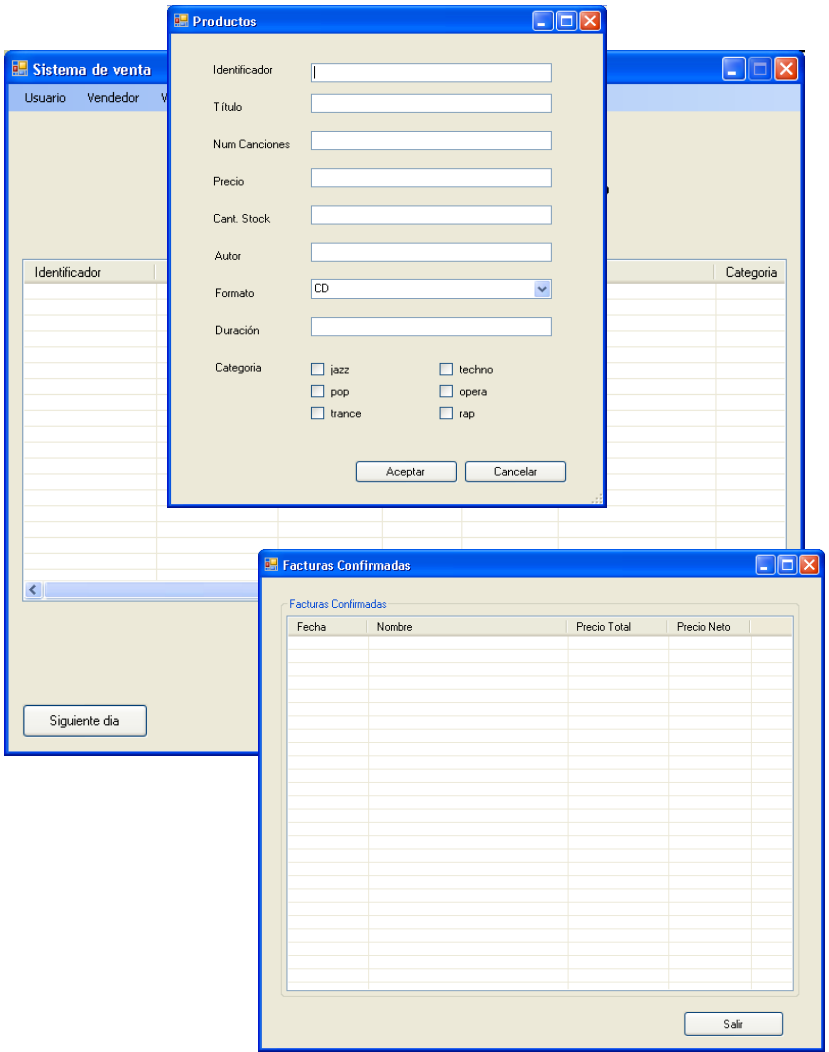
EXAMPLE

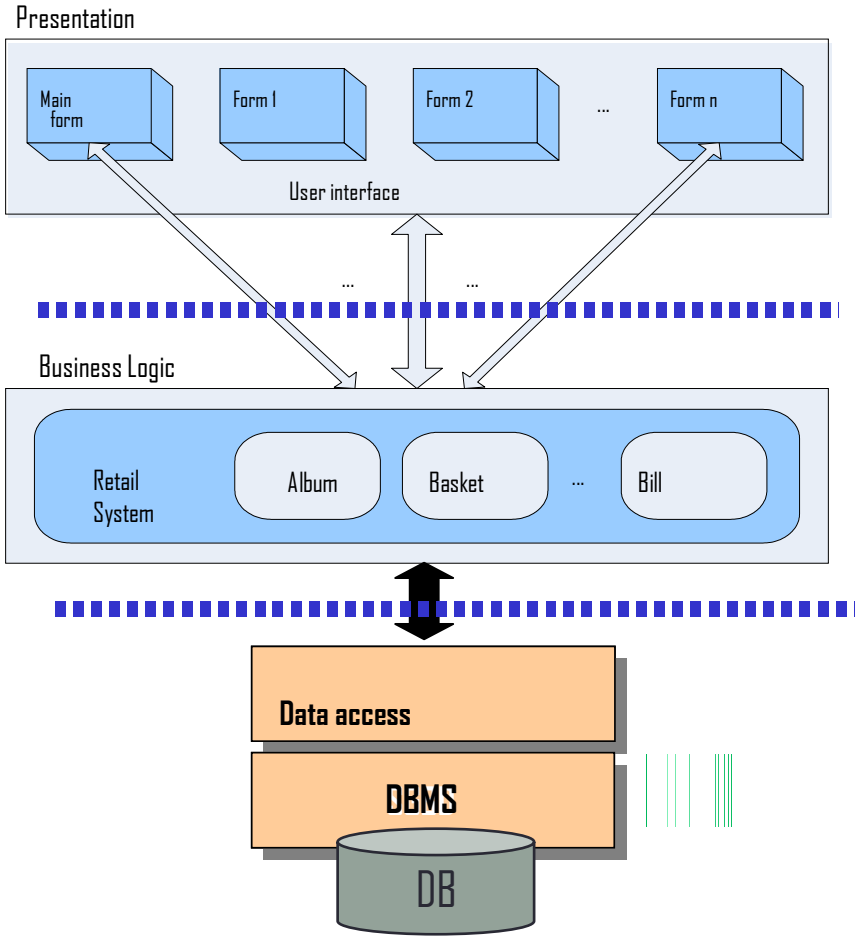
Three layered architecture: persistence

(Example)

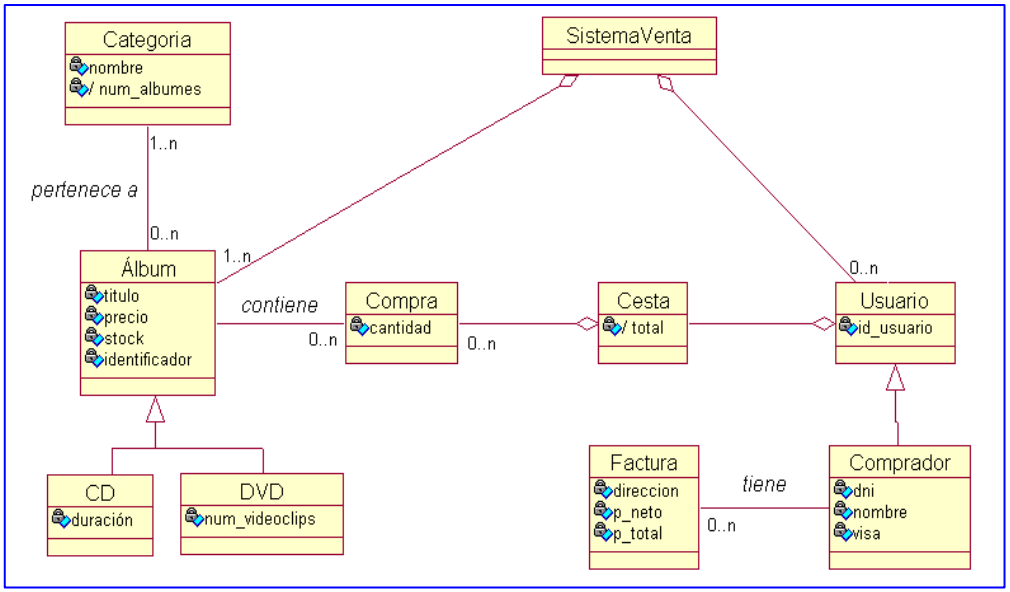


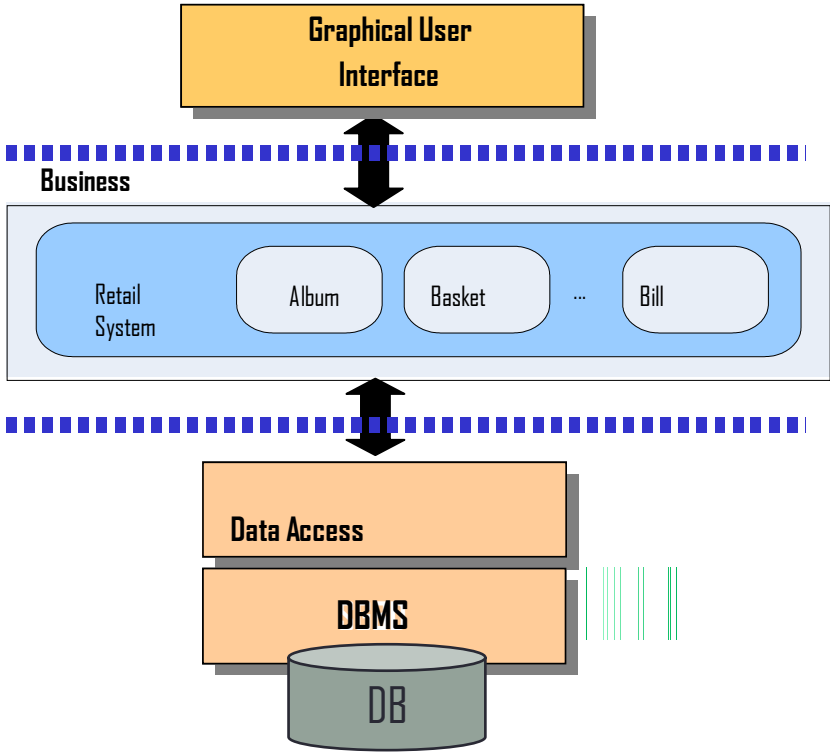
Three layered Architecture: Presentation



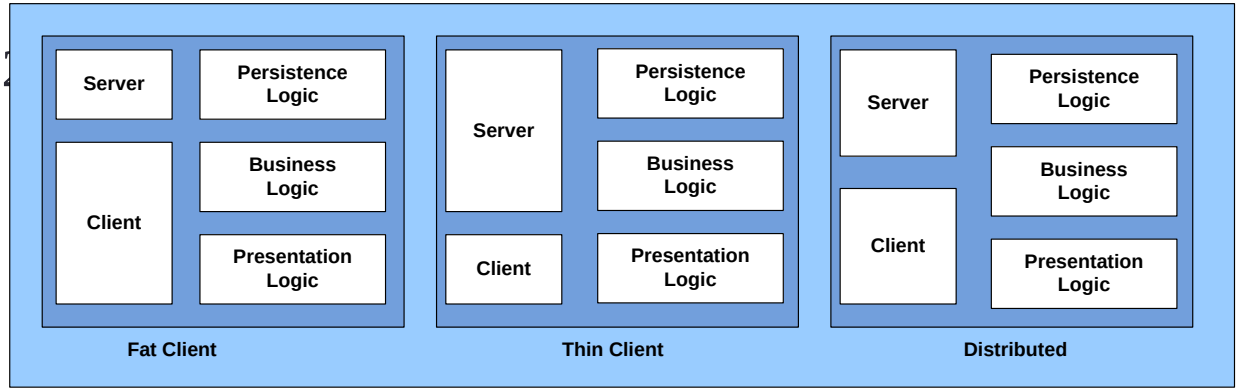


Three layered Architecture: Business Logic

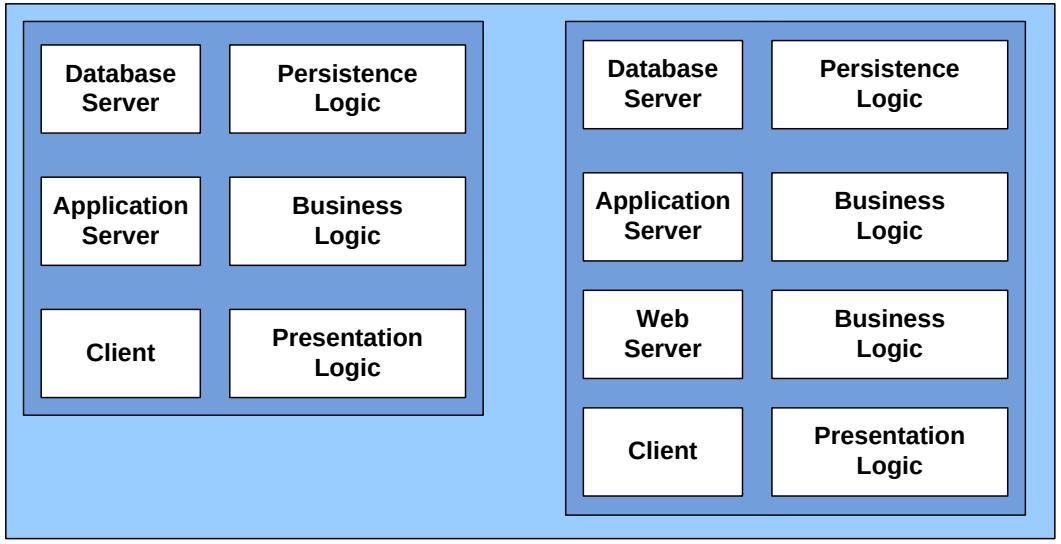




Summary: Business Logic distribution in architectures



- 3-n layers:



References

- **Alonso et al., Web Services: Concepts, Architectures and Applications, Springer, 2004**
  - **Chapters 1 & 2**
- **Sommerville. “Software Engineering”. Chapter 6**
- **David Parnas, "On the Criteria to Be Used in Decomposing Systems Into Modules“. Communications of the ACM, December 1972.**