Software Architecture Prototyping

Acknowledgement

- Some of the contents are adapted from material by Manzil-e-Maqsood

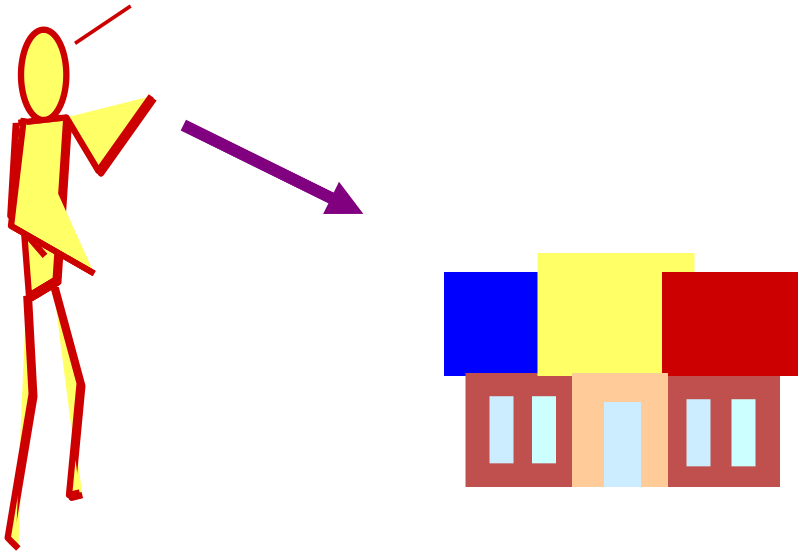
- Software Engineering – A practitioner’s approach by Roger S. Pressman and Bruce. R. Maxim

- Software Engineering by Ian Sommerville

Architecture of a building

customer requirements

"four bedrooms, three bathrooms,



architectural design

Architecture of a building

- Overall shape of the physical structure is the first thing that comes into mind
- In reality…
  - It is the manner in which various components of the building are integrated to form a cohesive whole

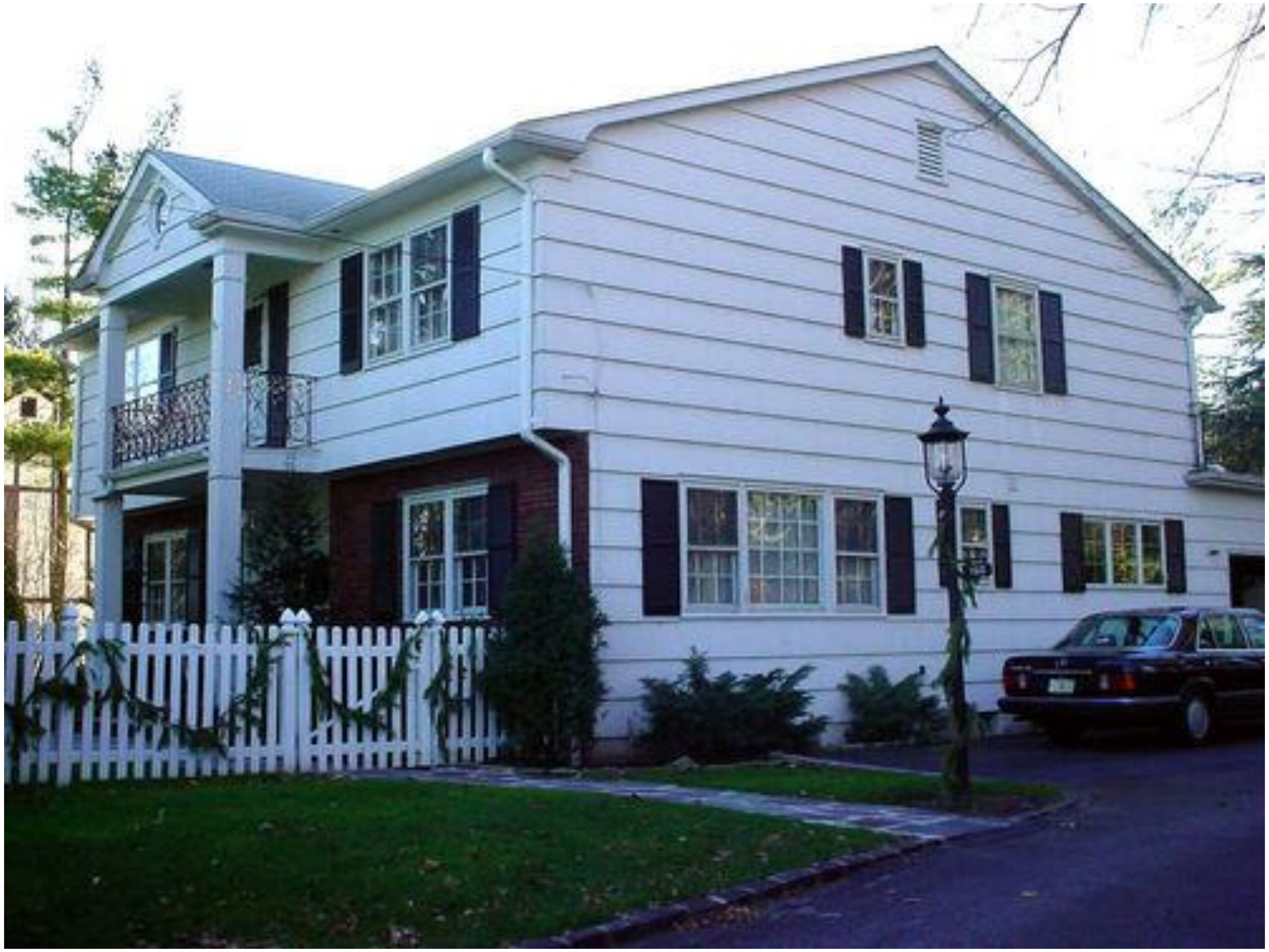
What is a Software architecture?

…"the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them"

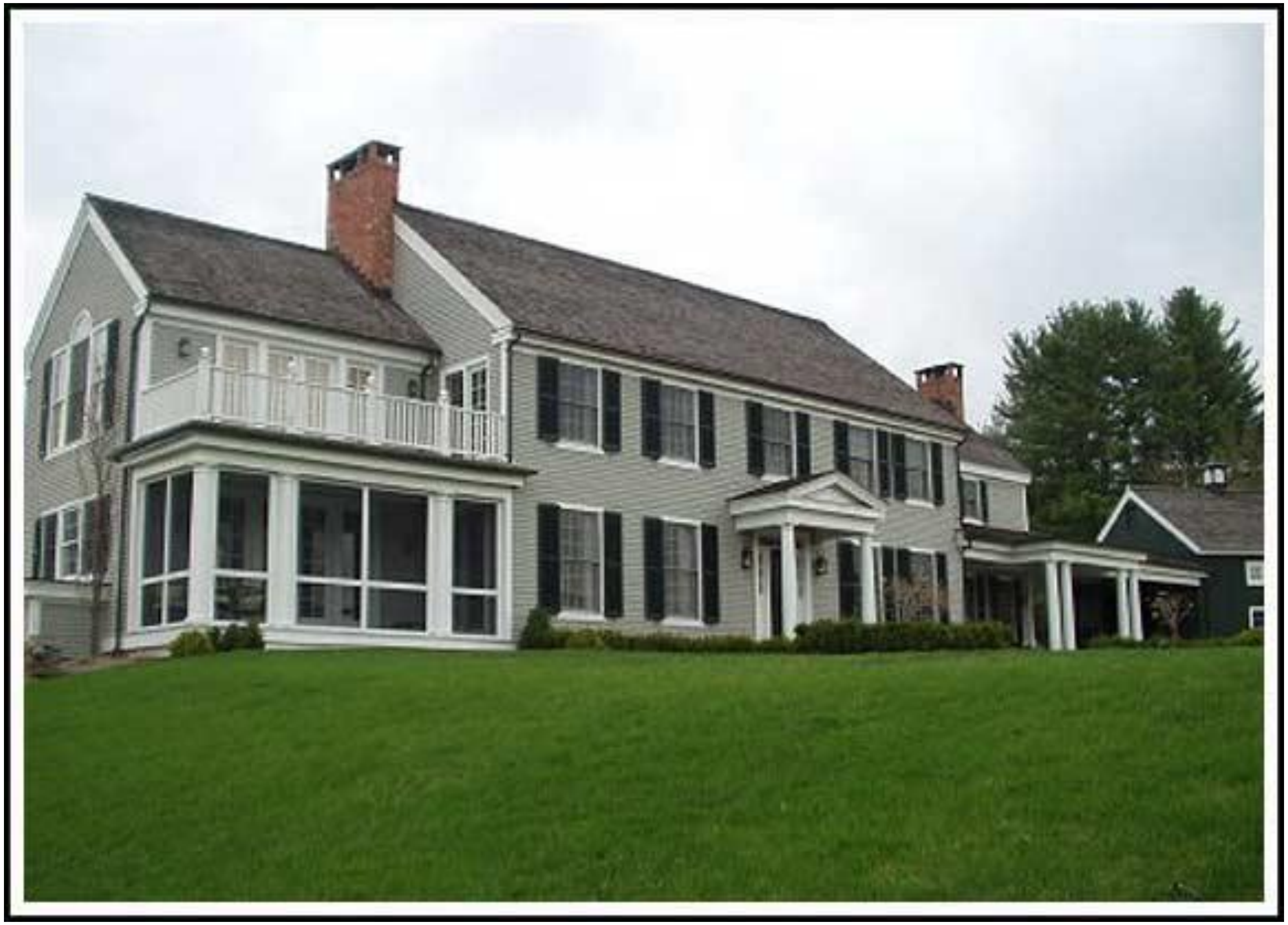
Architectural Styles

When a builder uses the phrase “center hall colonial” to describe a house, most people familiar with houses in the United States will be able to conjure a general image of what the house will look like and what the floor plan is likely to be. The builder has used an architectural style as a descriptive mechanism to differentiate the house from other styles (e.g., A-frame, raised ranch, Cape Cod). But more important, the architectural style is also a pattern for construction. Further details of the house must be defined, its final dimensions must be specified, customized features may be added, building materials are to be determined, but the pattern, a “center hall colonial” – guides the builder in his work.

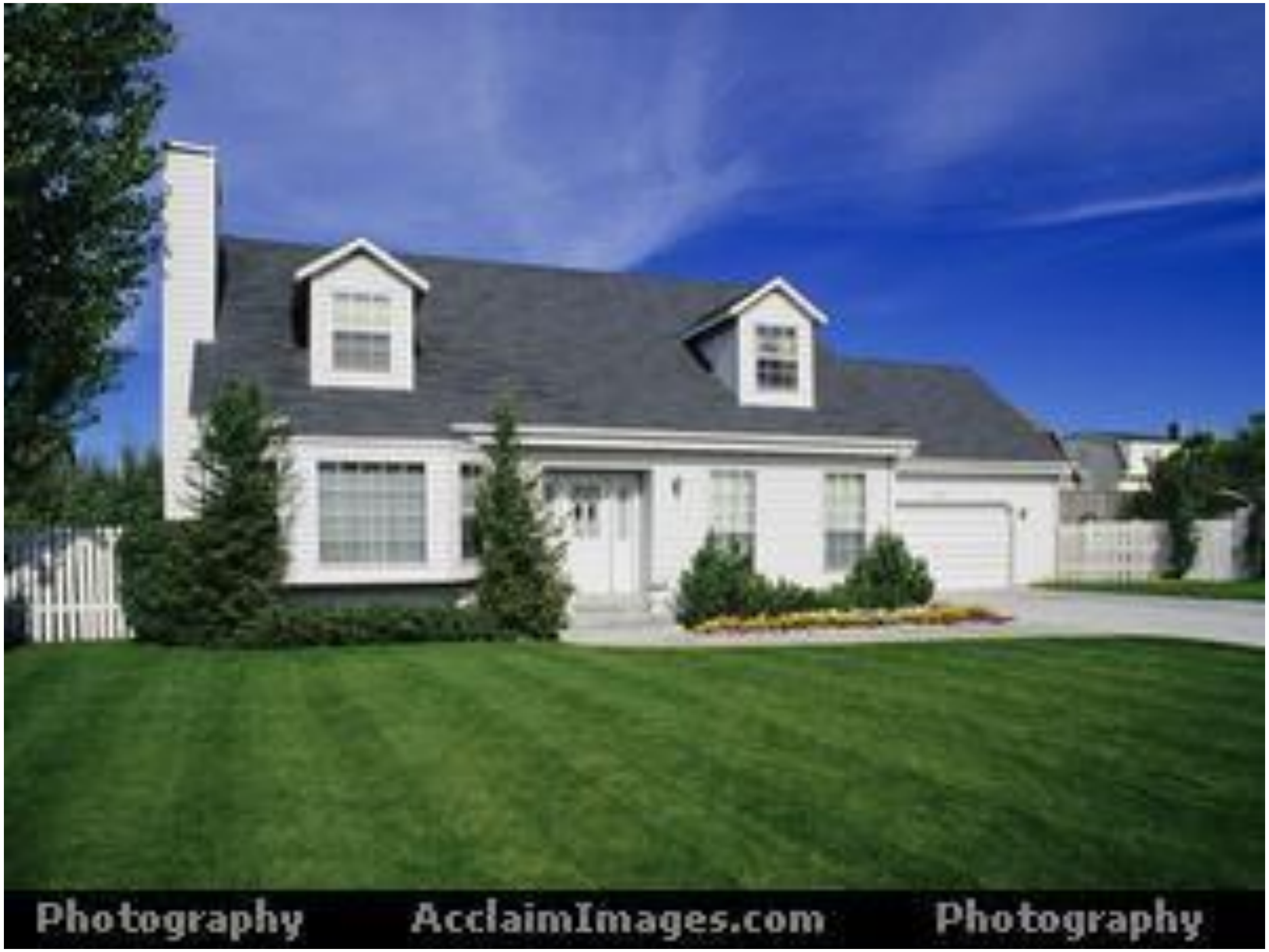
Center Hall Colonial-1



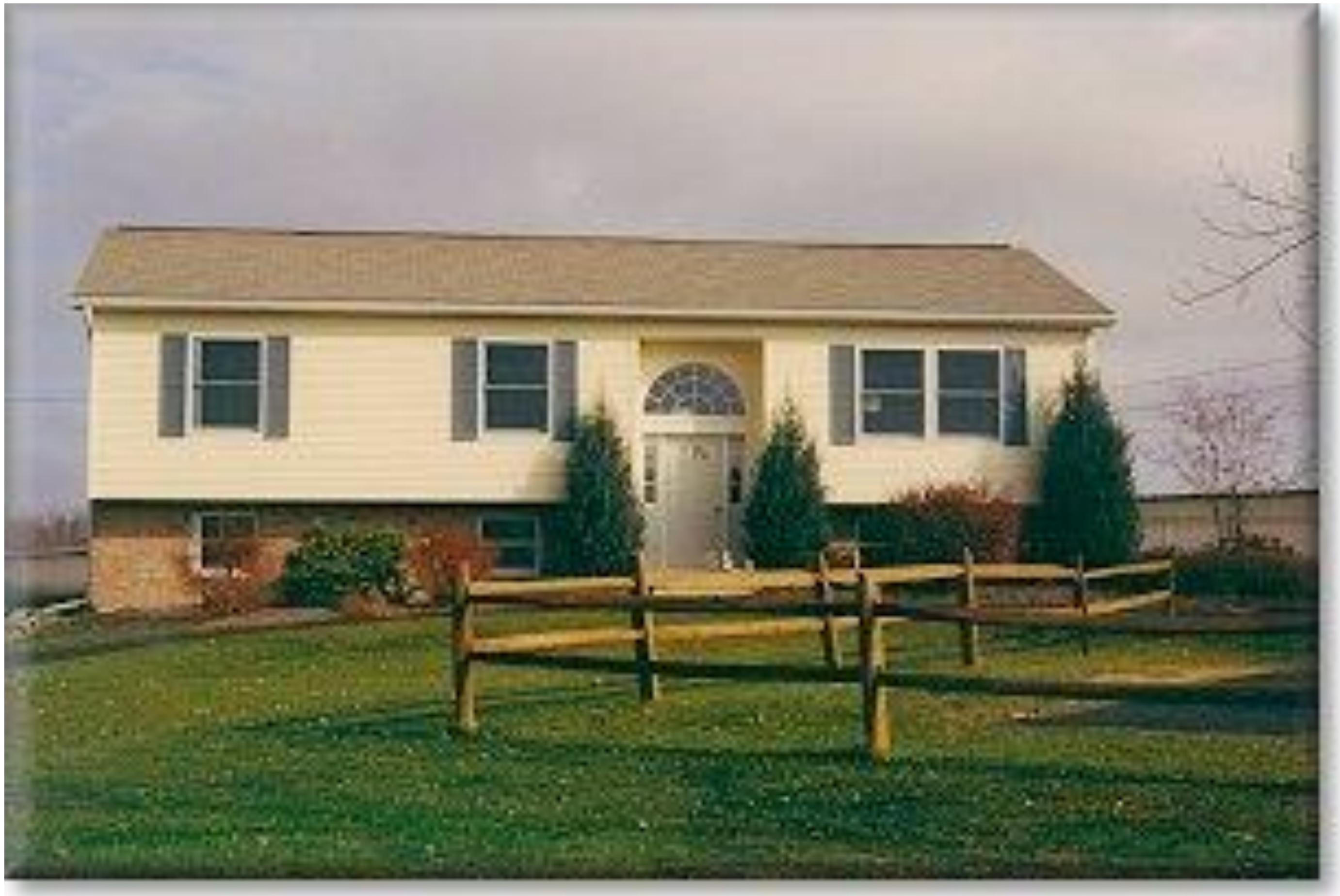
Center Hall Colonial-2



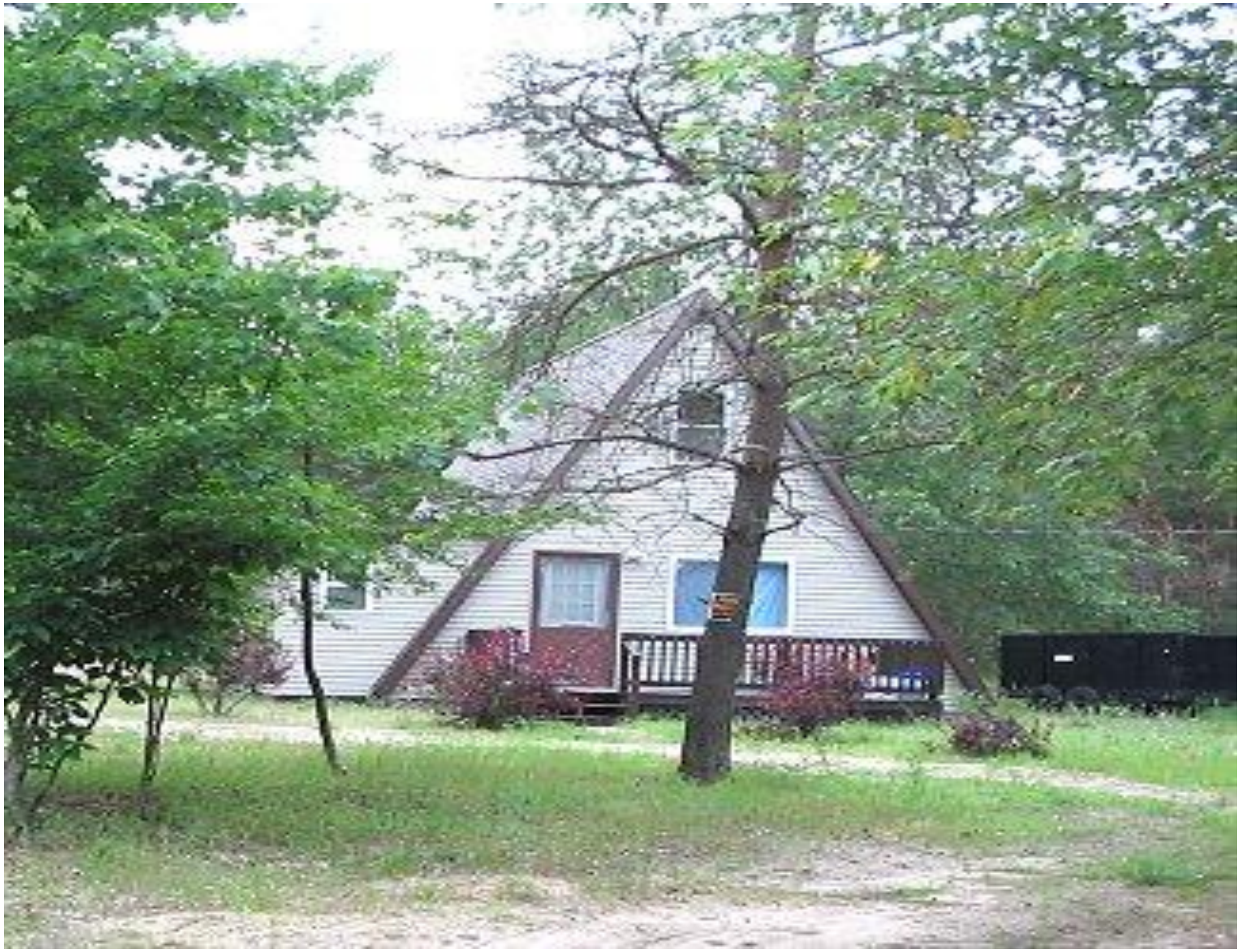
Cape Cod



Raised Ranch



A-Frame



Background

- Christopher Alexander, an architect observed that in buildings the same things are repeated with slight variations.
- These things can be reused again and again and can help make better architectures without the need to re-experiment.
- These, he called STYLES, or PATTERNS.

Some Architectural Styles…

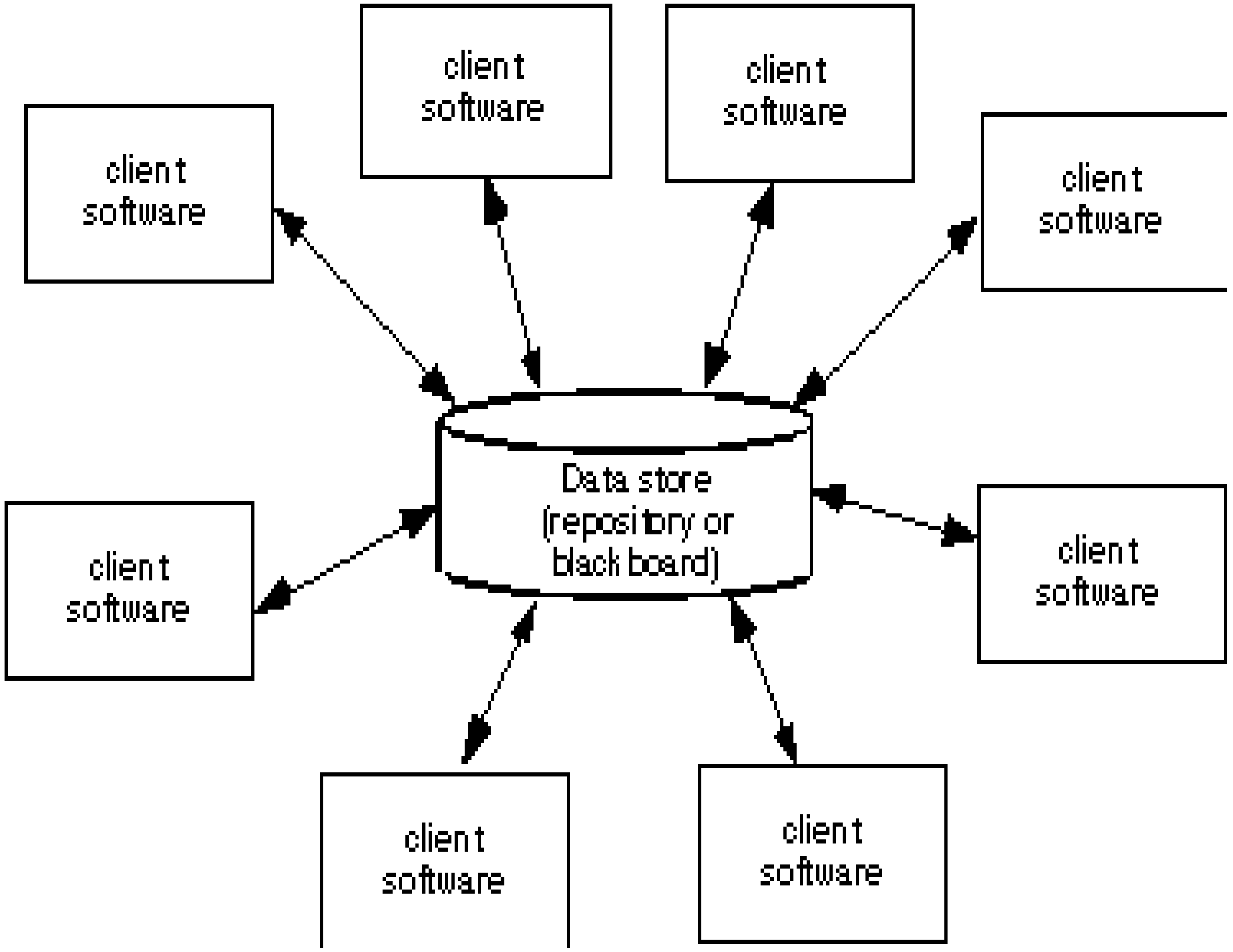
1. Data-centered architectures
2. Client Server Architecture
3. Data flow architectures
4. Layered architectures
5. Reference Architecture
6. Call and return architectures
7. Object-oriented architectures

Don't reinvent the wheel.. Use the things that are already there !!!

The Repository/Data-Centered Model

- Sub-systems must exchange data. This may be done in two ways:
  - Shared data is held in a central database or repository and may be accessed by all sub-systems
  - Each sub-system maintains its own database and passes data explicitly to other sub-systems
- When large amounts of data are to be shared, the repository model of sharing is most commonly used

Data-Centered/Repository Architecture



Repository/Data Centric Architecture Characteristics

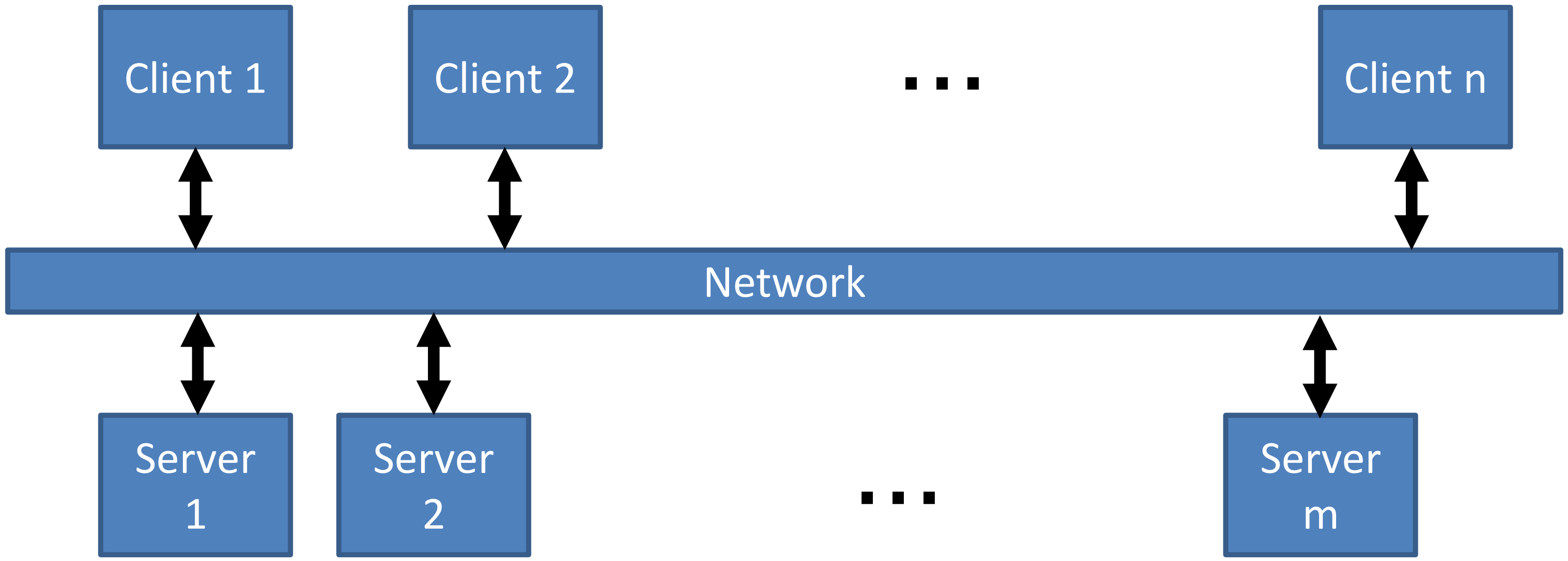
- **Advantages**
  - Efficient way to share large amounts of data.
  - Sub-systems need not be concerned with how data is produced.
  - Data security, backup and management is done at the central system that holds the data.
  - All sub systems know exactly what is present in the data repository; no data is hidden or abstracted, as it is a common repository.

- **Disadvantages**
  - Data evolution is difficult and expensive
    - increased coupling
    - increased ripple effects while bringing about a change)
  - Difficult to distribute efficiently; performance, scalability etc. are compromised.
  - Single point of failure.

Client/Server Architectures

- Set of servers which provide specific services such as printing, data management, etc.
- Set of clients which call on these services
- Network which allows clients to access servers

Client-Server Style



Client-server characteristics

- Advantages
  - Easy and straightforward distribution of data
  - Makes effective use of networked systems. May require cheaper hardware
  - Easy to add new servers or upgrade existing servers

- Disadvantages
  - Data interchange may be inefficient and affect performance of the system
  - Management of each server is difficult

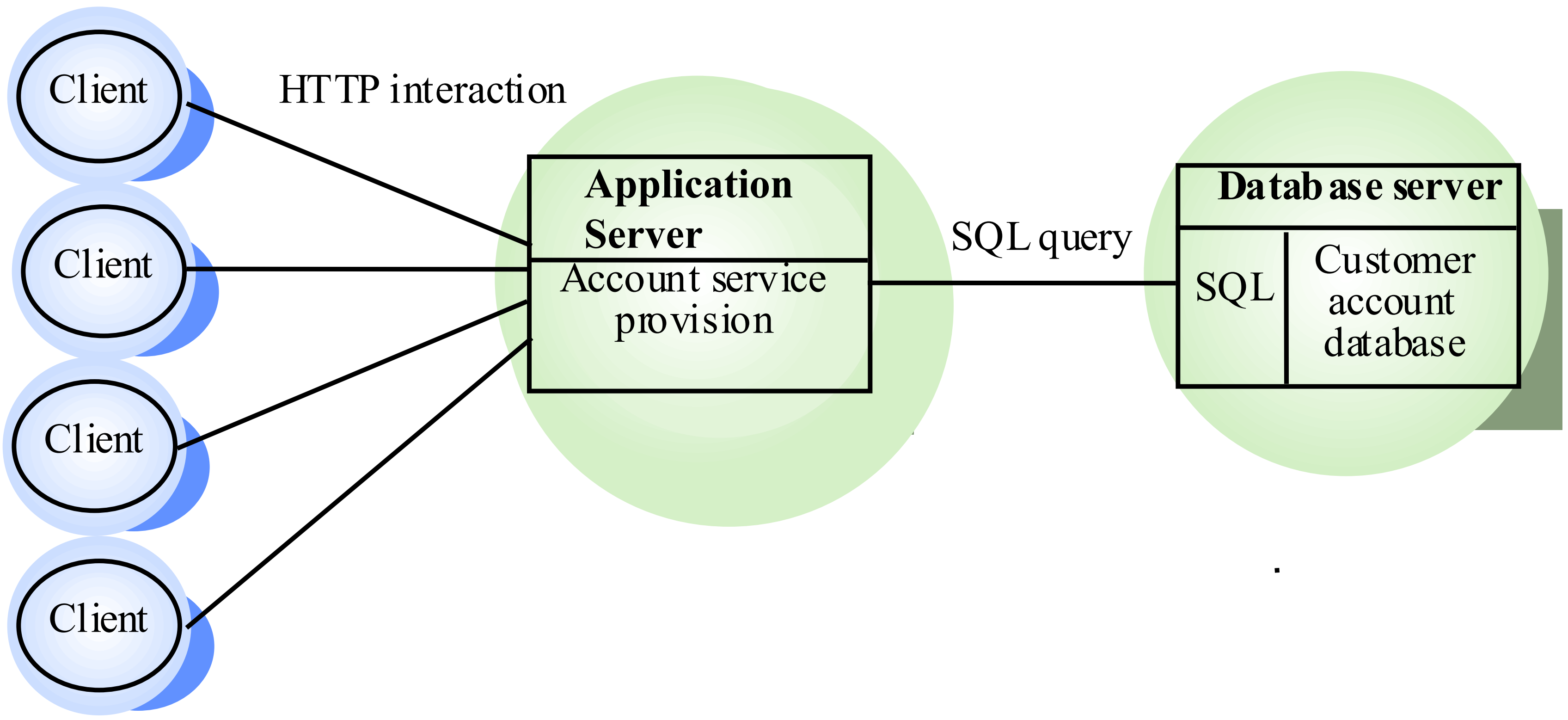
Three-tier Architecture

- Tier = layers / structures that are applicable to distributed environments.
- Data, applications, clients are all distributed.
- Each application architecture layers (presentation, application, database) may run on separate processors
- So, client machines has interface, servers had application logic and database servers served as data repositories.

Three-tier Architecture…

- Allows for better performance
  - Logic and data are independent of each other as logic is placed in application server machines and data is in data servers.
  - You can change the application without worrying about changing the database.
- Simpler to manage
- Highly scalable (you can add more servers)

Three-tier Architecture…



Prototyping

- What is a Prototype?
- Different types of prototypes

The problem

- Requirements are often poorly understood, uncertain, unclear and ambiguous

- Current requirements remain only partially understood until users actually use the system
  - they may change the requirements if they don’t get what they wanted
  - Results in wasted effort on your part

Why does this problem occur?

- Because many times users or customers only have a vague.
- The ideas are not concrete and are difficult to capture
- To capture the requirements for such vague, uncertain ideas, we use prototyping

So, What is Prototyping?

A technique of constructing a partial implementation, **MOCK-UP**, of a system so that customers, users or developers can **LEARN** more about a problem or a solution to that problem

More about Prototypes

- Prototypes are mockups of the real application. They are NOT real systems
- They show:
  - The type of application we will develop
- Users provide feedback after using the prototypes
- This helps in clear understanding of user requirements and will help in making the actual application

Primary Schools of Thought

- Throw-away Prototyping
  - The concept is to construct a prototype in order to learn more about the requirements and discard it after the desired knowledge is gained.
  - Also called quick-dirty and throw-away prototype

What are the quick-and-dirty characteristics?

- No design
- No comments
- No test plans
- No proper documentation

Primary Schools of Thought

- Evolutionary Prototyping
  - The concept is to evolve the initial prototype until it becomes the real system.

Next Lecture

- Software Testing