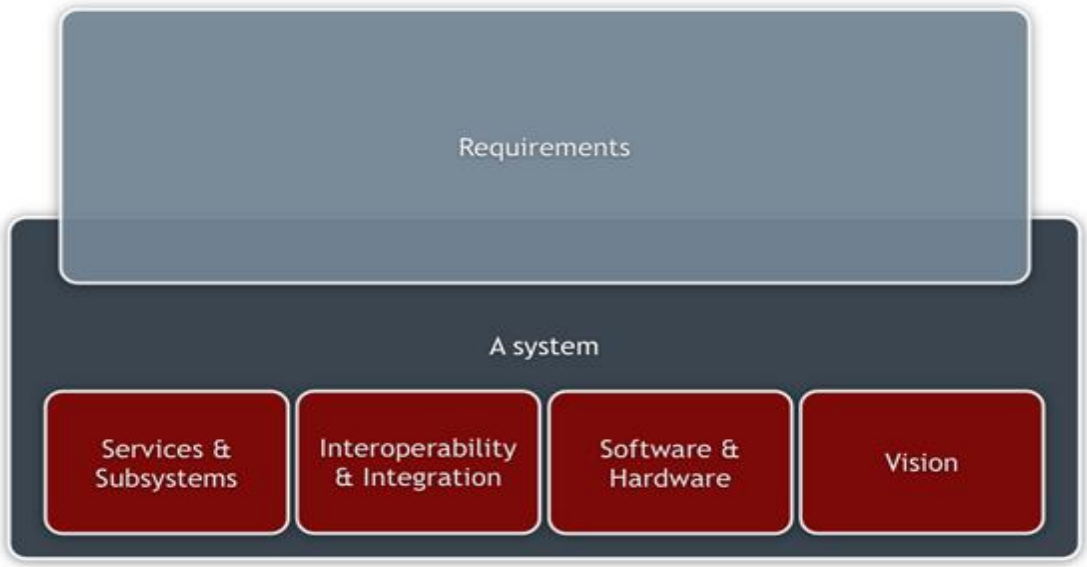
Lab 1
Computing Environment Architectures + SOA + Cloud Computing

System Architecture:

"System architecture" refers to the way in which desired functionality is met by hardware and software components as well as how these components relate to each other and the intended users of the system. The term "architecture" is often generically used to refer to the system architecture, at least within the context of software systems development. The system architecture can span several different business functions.



File Server Architecture:

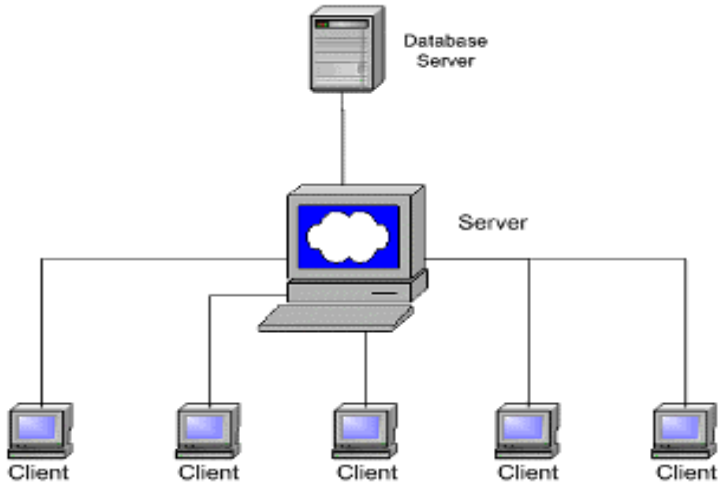
File Servers are useful for sharing information across the network. The client passes a request for file records over a network to the file server. This is the most primitive type of data service used for exchanging messages over the network to find the requested data. The file servers provide access to the remote server processors. In the typical implementation the software, shared data, databases and backups are stored on disk, tape and optical storage devices that are managed by the file server.

Client Server Architecture:

Client - A client is a single-user workstation that provides presentation services and the appropriate computing, connectivity and the database services and the interfaces relevant to the business need.

Server- A server is one or more multi-user processors with share memory providing computing, connectivity and the database services and the interfaces relevant to the business need.

The Client/Server computing is an environment that satisfies the business need by appropriately allocating the application processing between the client and the server processors. The protocol is the client requests the services from the server; the server processes the request and returns the result to the client. The communication mechanism is a message passing Inter-Process communication (IPC) that enables the distributed

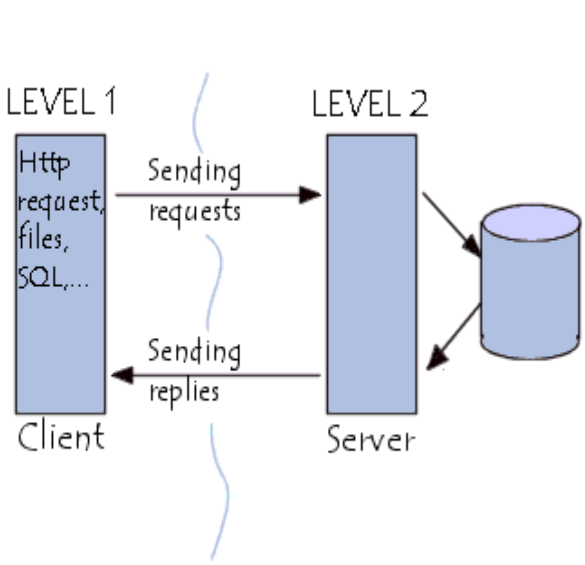


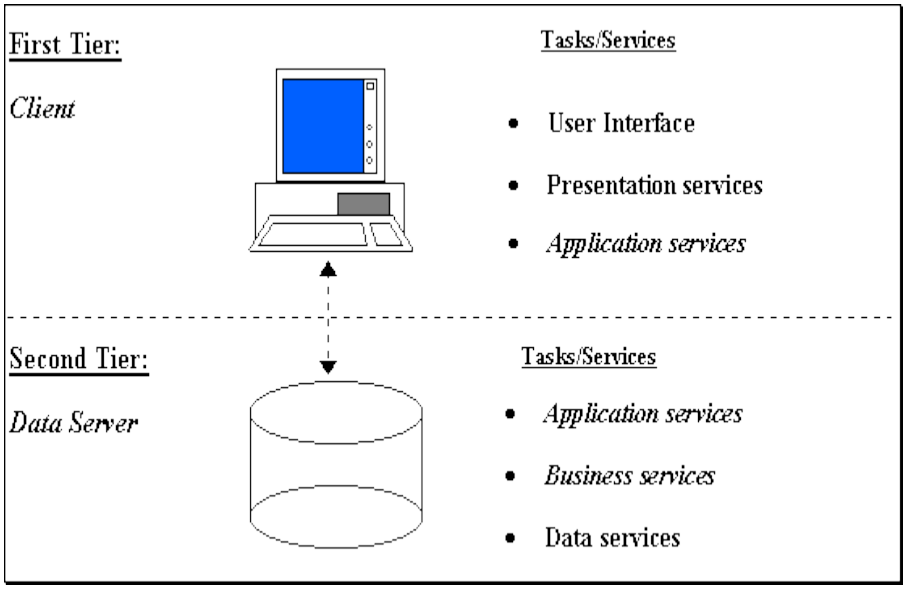
Types of Client-Server Architecture:

- **Two-Tier Architecture:**

Although there are several ways to design a two-tier client/server system, we will focus on examining what is overwhelmingly the most common implementation. In this implementation, the three components of an application (presentation, processing, and data) are divided among two software entities (tiers): client application code and database server. A robust client application's development language and a versatile mechanism for

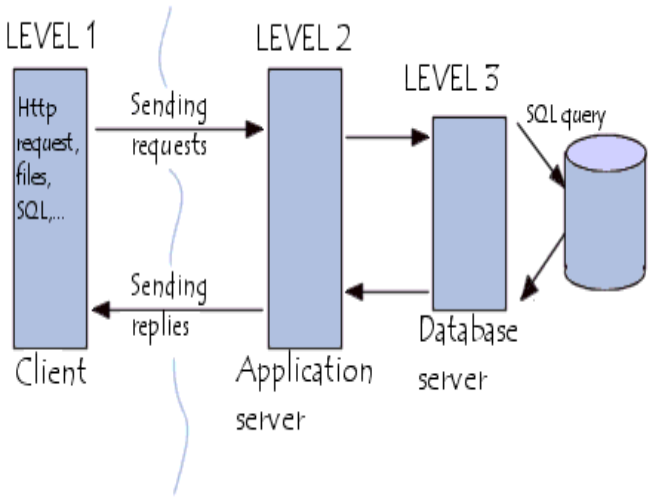
transmitting client requests to the server are essential for a two-tier implementation. Presentation is handled exclusively by the client, processing is split between client and server, and data is stored on and accessed via the server.

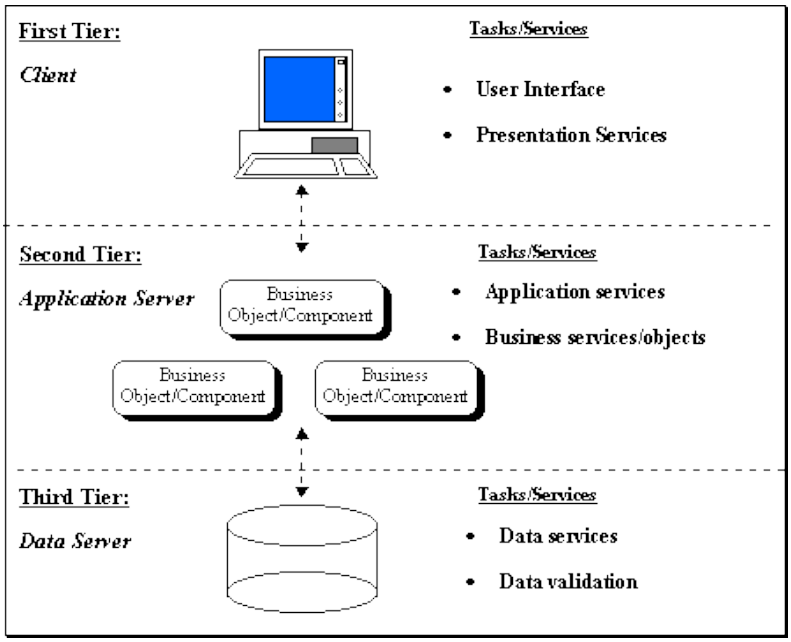




- **Three-Tier Architecture:**

The tree tier architecture attempts to overcome some of the limitations of the two-tier scheme by separating presentation, processing, and data into separate, distinct software entities (tiers). The same types of tools can be used for presentation as were used in a two-tier environment; however these tools are now dedicated to handling just the presentation. When the presentation client requires calculations or data access, a call is made to a middle tier functionality server. This tier can perform calculations or can make requests as a client to additional servers. The middle tier servers are typically coded in a highly-portable, non-proprietary language such as C. Middle-tier functionality servers may be multi-threaded and can be accessed by multiple clients, even those from separate applications.





Service-Oriented Architecture:

service-oriented architecture (SOA) is a set of principles and methodologies for designing and developing software in the form of interoperable services. These services have well-defined business functionalities that are built as software components (discrete pieces of code and/or data structures) which can be reused for different purposes. SOA design principles are used during the phases of systems development and integration.

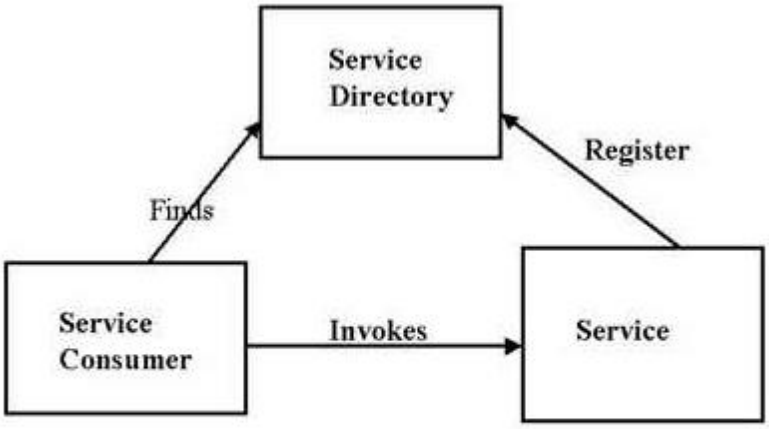


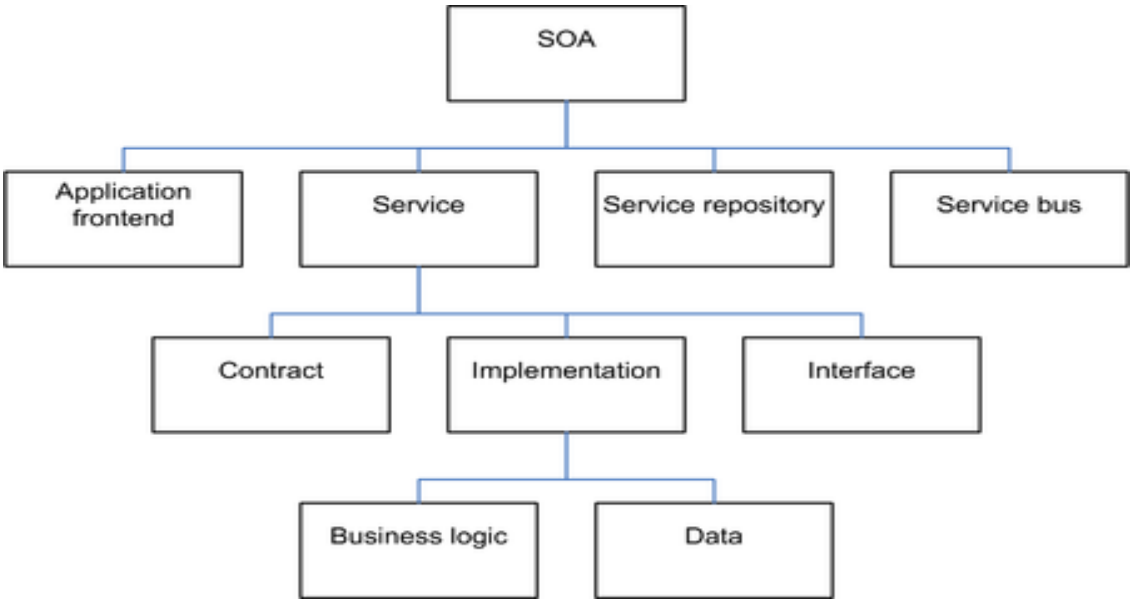
Fig 1. Service Oriented Architecture

SOA generally provides a way for consumers of services, such as web-based applications, to be aware of available SOA-based services. For example, several disparate departments within a company may develop and deploy SOA services in different implementation languages; their respective clients will benefit from a well-defined interface to access them. XML is often used for interfacing with SOA services.

Service-orientation requires loose coupling of services with operating systems and other technologies that underlie applications. SOA separates functions into distinct units, or services,[1] which developers make accessible over a network in order to allow users to combine and reuse them in the production of applications. These services and their corresponding consumers communicate with each other by passing data in a well-defined, shared format, or by coordinating an activity between two or more services

Web services can implement a service-oriented architecture. Web services make functional building-blocks accessible over standard Internet protocols independent of platforms and programming languages. These services can represent either new applications or just wrappers around existing legacy systems to make them network-enabled.

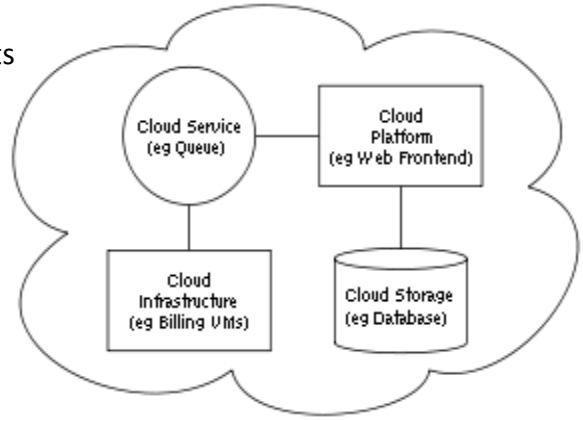
Elements of SOA:

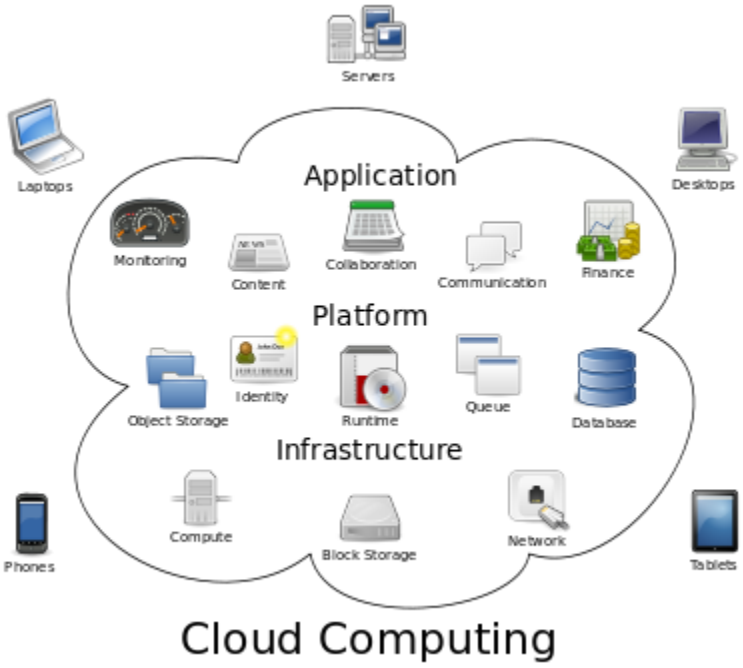


Cloud Computing:

Cloud Computing is a general term used to describe a new class of network based computing that takes place over the Internet.

Cloud computing architecture refers to the components and subcomponents required for cloud computing. These components typically consist of a frontend platform (e.g. fat client, thin client, mobile device), a back end platforms (servers, storage), a cloud based delivery, and a network (e.g. Internet, Intranet, Intercloud). All of which combined makes up cloud computing architecture.





Companies may make a number of considerations with regard to which cloud computing model they choose to employ, and they might use more than one model to solve different problems. An application needed on a temporary basis might be best suited for deployment in a public cloud because it helps to avoid the need to purchase additional equipment to solve a temporary need. Likewise, a permanent application, or one that has specific requirements on quality of service or location ofdata, might best be deployed in a private or hybrid cloud.

Cloud computing types:

(1) Public clouds

Public clouds are run by third parties, and applications from different customers are likely to be mixed together on the cloud's servers, storage systems, and networks. Public clouds are most often hosted away from customer premises, and they provide a way to reduce customer risk and cost by providing a flexible, even temporary extension to enterprise infrastructure.

(2) Private clouds

Private clouds are built for the exclusive use of one client, providing the utmost control over data, security, and quality of service. The company owns the infrastructure and has control over how applications are deployed on it. Private clouds may be deployed in an enterprise datacenter.

(3) Hybrid clouds

Hybrid clouds combine both public and private cloud models. They can help to provide on-demand, externally provisioned scale. The ability to augment a private cloud with the resources of a public cloud can be used to maintain service levels in the face of rapid workload fluctuations. This is most often seen with the use of storage clouds to support Web 2.0 applications. A hybrid cloud also can be used to handle planned workload spikes. Sometimes called "surge computing," a public cloud can be used to perform periodic tasks that can be deployed easily on a public cloud.