BITS Pilani

BITS Pilani
Hyderabad Campus

Dr. Manik Gupta
Associate Professor
Department of CSIS

innovate achieve lead

BITS Pilani
Hyderabad Campus

Distributed Data Systems (CS G554)
Lecture 3

Monday, 12th August 2024

Previous lecture recap

- What is a distributed system?
- What are characteristics, design goals of distributed systems?
- Examples of distributed systems

Today's agenda

- Distributed system architectures

Architectures

- Distributed systems are complex pieces of software of which the components are dispersed across multiple machines
- Different ways to organize the distributed systems
  - Logical organization of collection of software components
  - Physical realization
- **Software architecture** tell us how various software components are to be organized and how they should interact
- **System architecture** is the final instantiation of a software architecture

Architectural styles

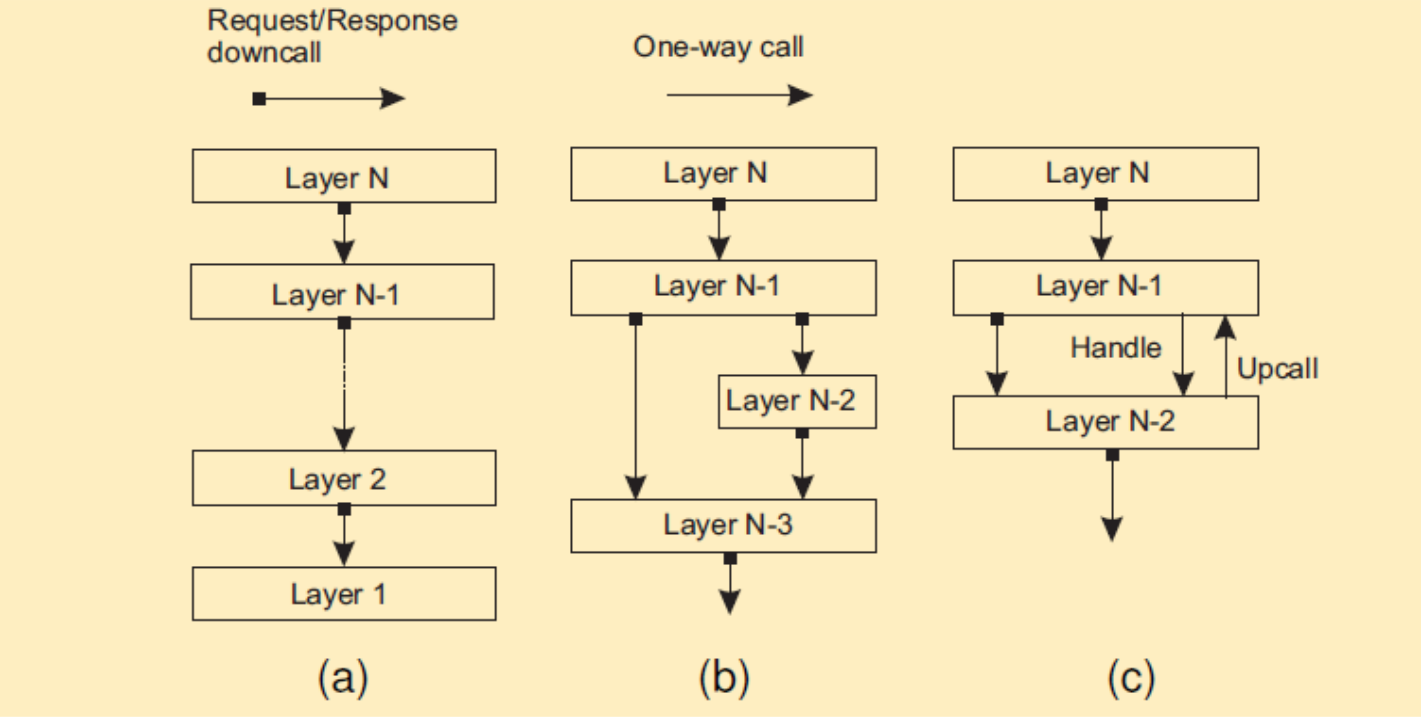
- A style is formulated in terms of
  - (replaceable) **components** with well-defined interfaces
  - the way that **components are connected** to each other
  - the **data** exchanged between components
  - how these components and connectors are jointly configured into a system
- Components
  - Modular unit with **well defined required and provided interfaces** that is replaceable within its environment.
- Connector
  - A mechanism that **mediates communication, coordination, or cooperation** among components. Example: facilities for (remote) procedure call, messaging, or streaming.

Architectural styles

- Layered architectures
- Object based architectures
- Resource centred architectures
- Event based architectures

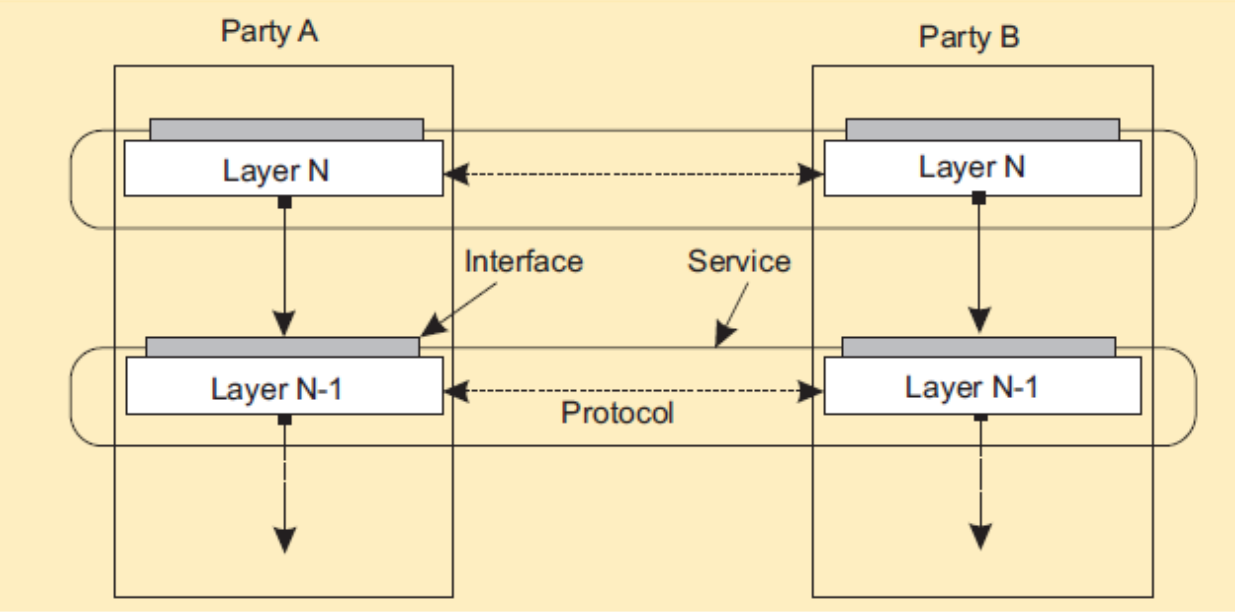
Layered architecture

- Pure layered organization
- Mixed layer organization
- Layered organization with up calls



Example: Communication protocols

- Layered communication protocol stack
  - Each layer offers an **interface** specifying the functions that can be called.
  - The interface should completely hide the implementation of a **service**
  - **Protocol** defines the rules that the parties follow in order to exchange information

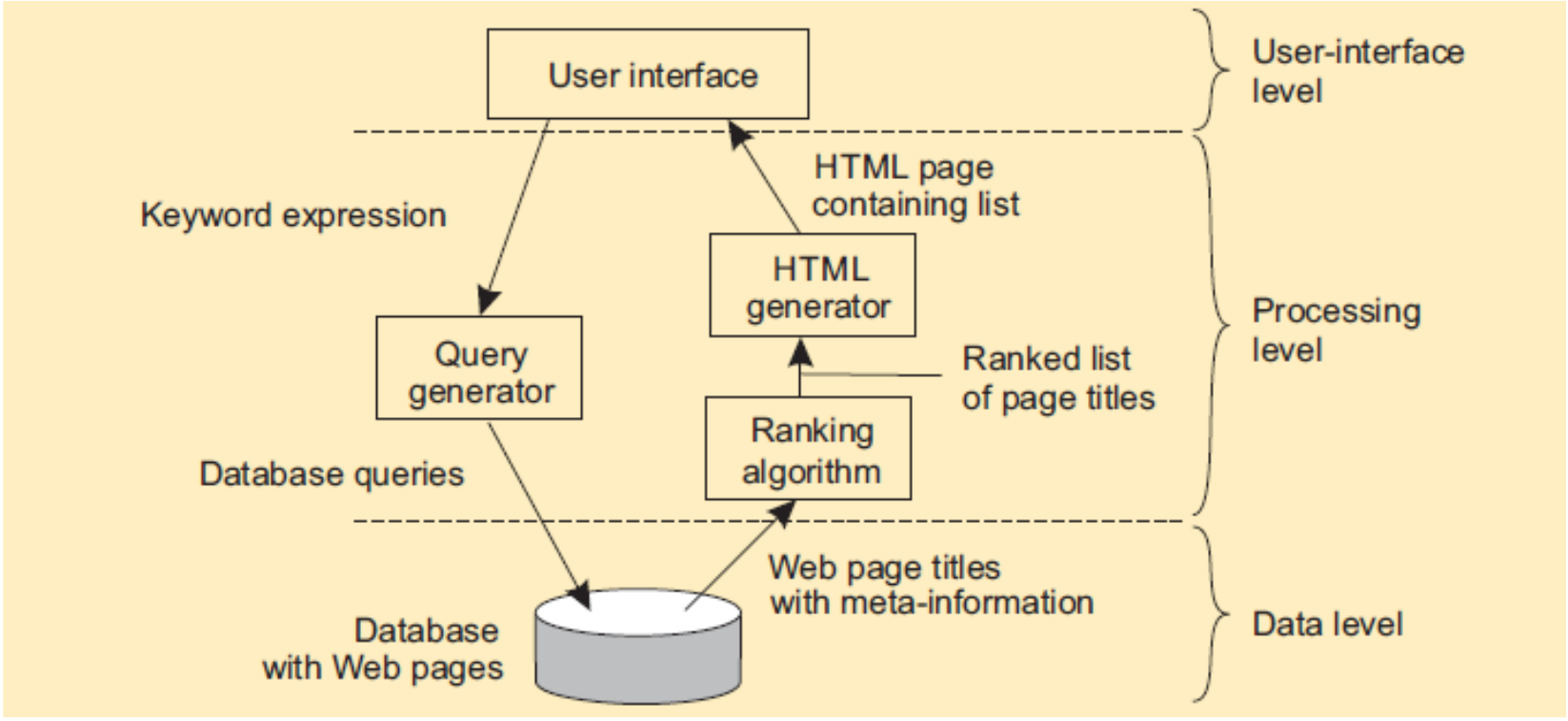


Application layering

- Traditional three-layered view
  - Application-interface layer contains units for interfacing to users or external applications
  - Processing layer contains the functions of an application, i.e., without specific data
  - Data layer contains the data that a client wants to manipulate through the application components

Application layering

A simple search engine

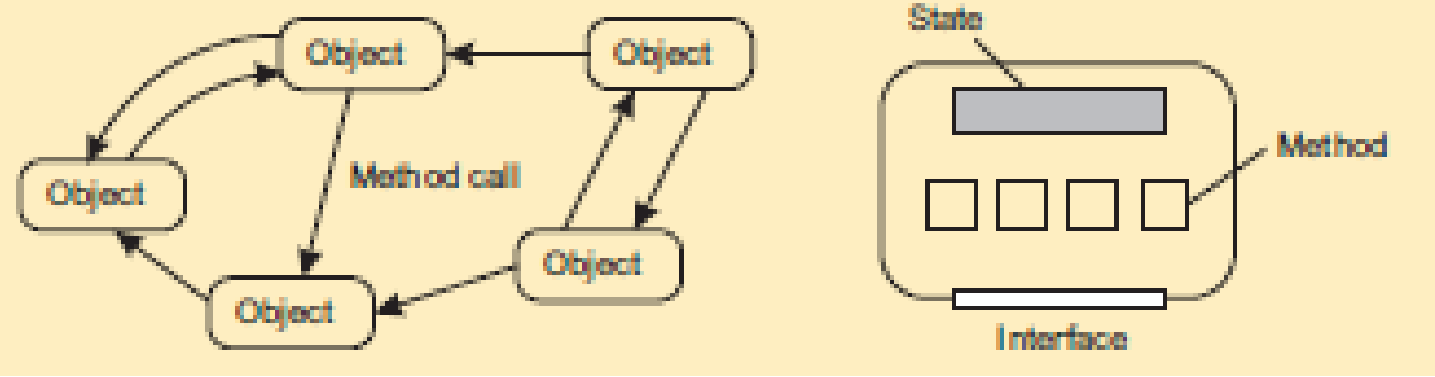


Discussion

- Can you think of how a decision support system for stock brokerage be designed using the layered architecture?

Object-based style

- Components are objects connected to each other through procedure calls.
- Objects may be placed on different machines; calls can thus execute across a network.



- What is the advantage of object based architectures?
  - Objects are said to encapsulate data and offer methods on that data without revealing the internal implementation.
  - Paving the road towards Service Oriented Architectures!

https://medium.com/@SoftwareDevelopmentCommunity/what-is-service-oriented-architecture-fa894d11a7ec#:~:text=Service%2DOriented%20Architecture%20(SOA),of%20vendors%20and%20other%20technologies.

RESTful architectures

- View a distributed system as a collection of **resources**, individually managed by components.
- Resources may be added, removed, retrieved, and modified by (remote) applications.
- Key characteristics
  - Resources are identified through a single naming scheme
  - All services offer the same interface consisting of four operations (PUT, GET, DELETE, POST)
  - Messages sent to or from a service are fully self-described
  - After executing an operation at a service, that component forgets everything about the caller – stateless execution

Operation	Description
PUT	Create a new resource
GET	Retrieve the state of a resource in some representation
DELETE	Delete a resource
POST	Modify a resource by transferring a new state

Example: Amazon's Simple Storage Service

- Objects (i.e., files) are placed into buckets (i.e., directories). Buckets cannot be placed into buckets.
- Operations on ObjectName in bucket BucketName require the following identifier:
  - *http://BucketName.s3.amazonaws.com/ObjectName*
- All operations are carried out by sending HTTP requests:
  - Create a bucket/object: PUT, along with the URI
  - Listing objects: GET on a bucket name
  - Reading an object: GET on a full URI

- https://hub.packtpub.com/defining-rest-and-its-various-architectural-styles/
- https://medium.com/@audira98/why-should-we-choose-rest-client-server-model-to-develop-web-apps-c3bb2451b13a

Publish subscribe architecture

- Requirement for an architecture in which the dependencies between processes becomes as loose as possible as the systems grow and processes can join/leave more easily
- Strong separation between **processing** and **coordination**
  - View the system as a **collection of autonomously operating processes**
  - Coordination consists of both **communication** and **cooperation** between processes

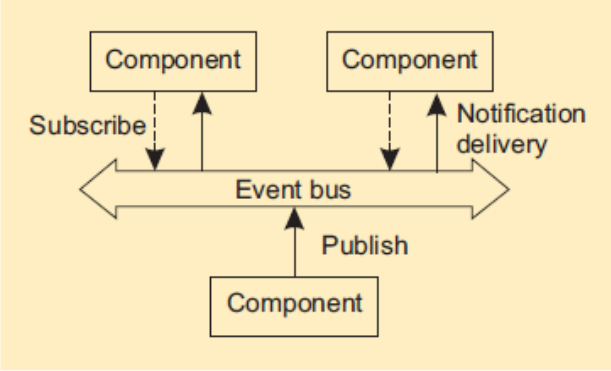
Coordination models

- Taxonomy of coordination models that can be applied to distributed systems can be distinguished along two different dimensions - **Temporal** and **referential** coupling between the processes

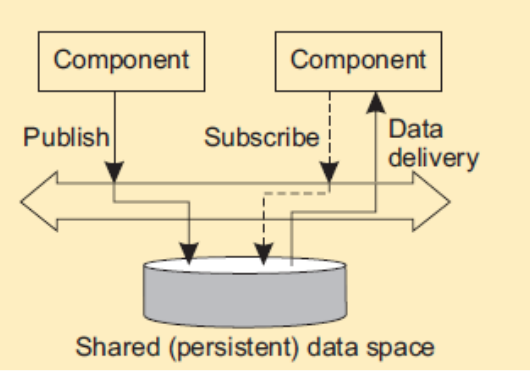
	Temporally coupled	Temporally decoupled
Referentially coupled	Direct	Mailbox
Referentially decoupled	Event-based	Shared data space

Coordination models

- Temporally and referentially coupled – **Direct communication**
- Temporally decoupled and referentially coupled – **Mailbox communication**
- Temporally coupled and referentially decoupled – **Event based communication**



- Temporally and referentially decoupled – **Shared Data space**



Exercise

Read and find out more about:

- Types of publish subscribe systems. What are the design issues involved?
- Why are pub-sub architectures preferred for large scale distributed systems?

- https://medium.com/@adriennedomingus/distributed-systems-an-introduction-to-publish-subscribe-pub-sub-6bc72812a995
- https://www.ably.io/concepts/pub-sub

System architecture

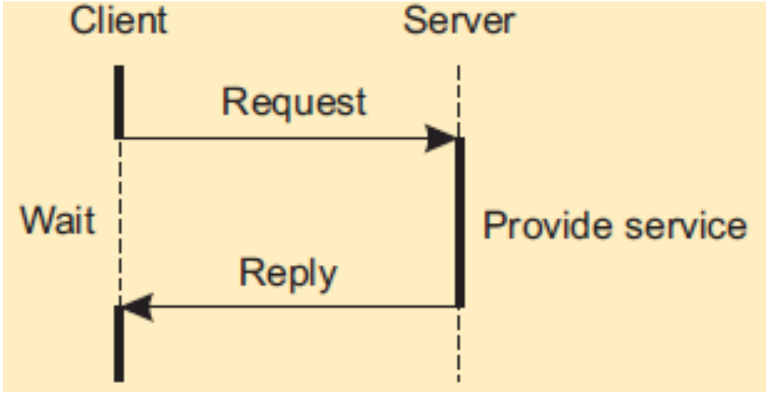
- Deciding on software components, their interaction and their placement leads to an instance of a software architecture known as a system architecture.

Centralized system architectures

Basic Client–Server Model

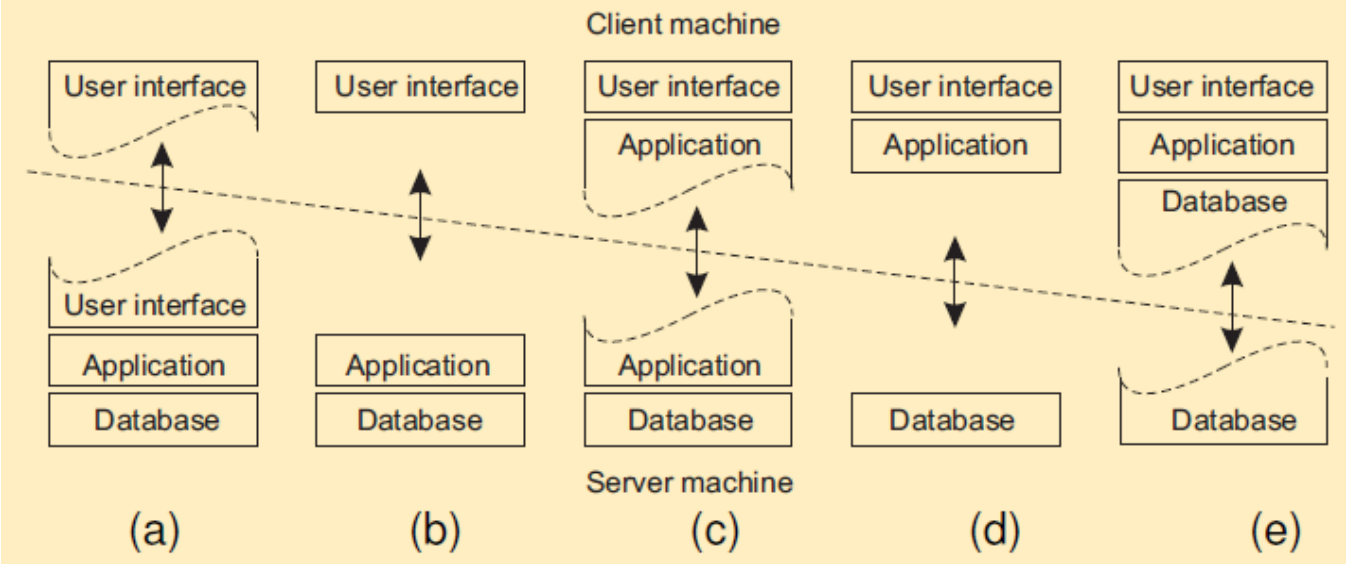
Characteristics:

- There are processes offering services (servers)
- There are processes that use services (clients)
- Clients and servers can be on different machines
- Clients follow request/reply model with respect to using services



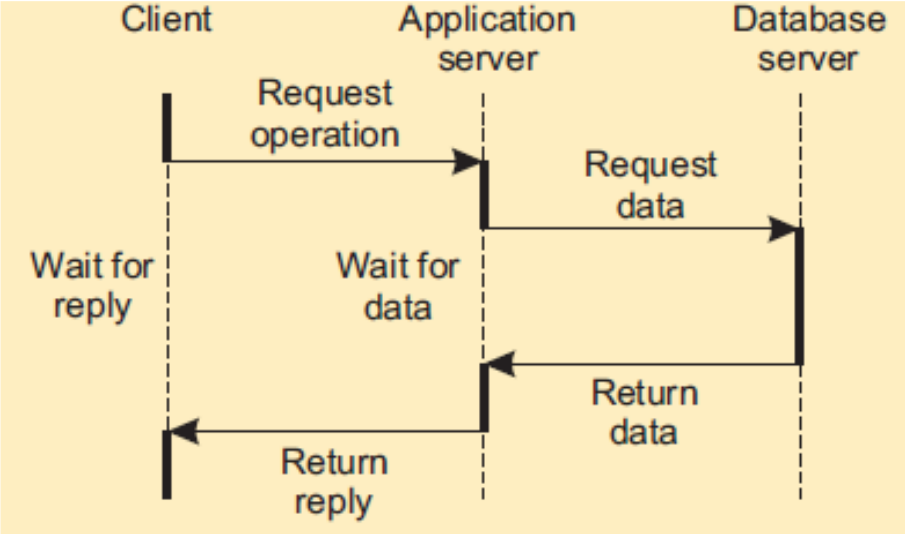
Multi-tiered centralized system architectures

- Three logical layers can be distributed physically across number of machines
  - Single-tiered: dumb terminal/mainframe configuration
  - Two-tiered: client/single server configuration
  - Three-tiered: each layer on separate machine
- Client server organization in two-tiered architectures



Being client and server at same time

- Three tier architecture (user interface layer, processing layer, data layer)



- Can you think of an example?
  - Transaction processing monitor
  - Organization of Web sites

Decentralized organizations

Peer-to-peer (P2P) architectures

- Processes are all equal
- Interaction between processes is symmetric and each process will act as a client and a server at the same time

- How to organize the processes in an overlay network?
  - A network in which nodes are formed by the processes and links represent the possible communication channels

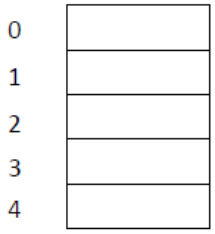
- Can you think of some advantages of P2P architectures?

Structured P2P

- Nodes are organized in an overlay that adheres to a specific, deterministic topology : binary tree, grid etc.
- Topology is used to efficiently look up data
- Each data item is uniquely associated with a key, in turn used as an index. Common practice: use a hash function

  key(data item) = hash(data item's value):
- P2P system now responsible for storing (key,value) pairs and system is seen to implement a distributed hash table

A hash tables has b buckets
- Any item x is put into bucket h(x)
- h(x) must be at most b for all x

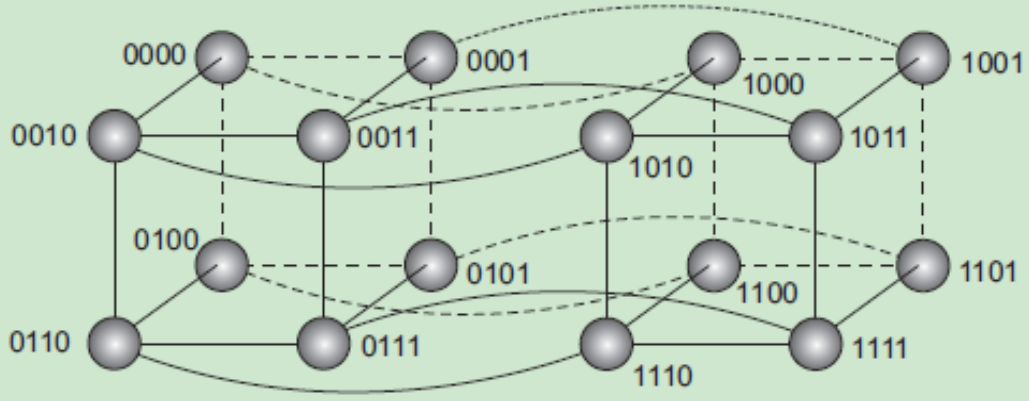


Example: a hash table of 5 buckets
- Any item x is put into bucket x mod 5
- Insert numbers 3, 5, 12, 116, 211

Example

Example of simple P2P system organised as a hypercube

Each data item is associated with a node
Hash the value of the data item to a key – 0 to 15



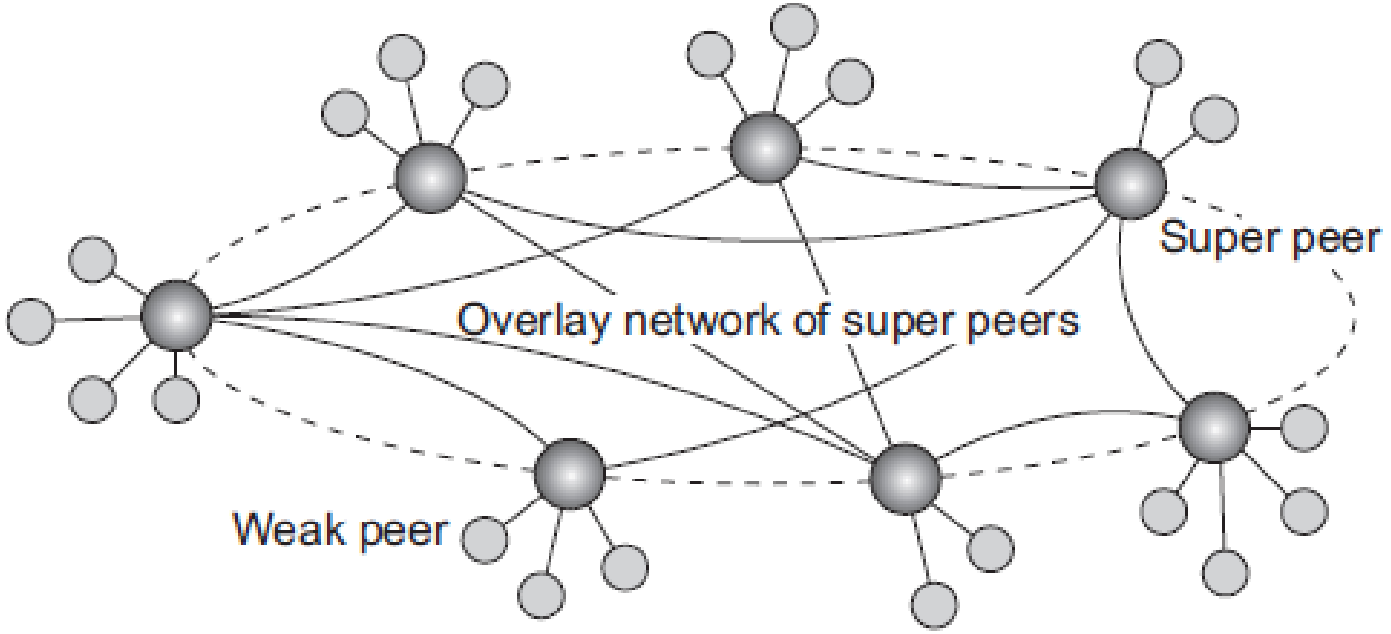
Looking up d with key $k \in \{0,1,2,\ldots,2^4-1\}$ means routing request to node with identifier k.

Unstructured P2P

- Each node maintains an adhoc list of neighbours and the resulting overlay resembles a random graph
- There is no deterministic way of routing a lookup request to a specific data item, so a node can resort to search for a request by means of flooding or randomly walking through the network

Hierarchically organized P2P networks

- In case of content delivery system, nodes can offer a storage service to host copies of web documents
  - Special nodes that maintain an index called **super peers** organised in a peer to peer network
  - Every regular peer is connected as a client to a super peer

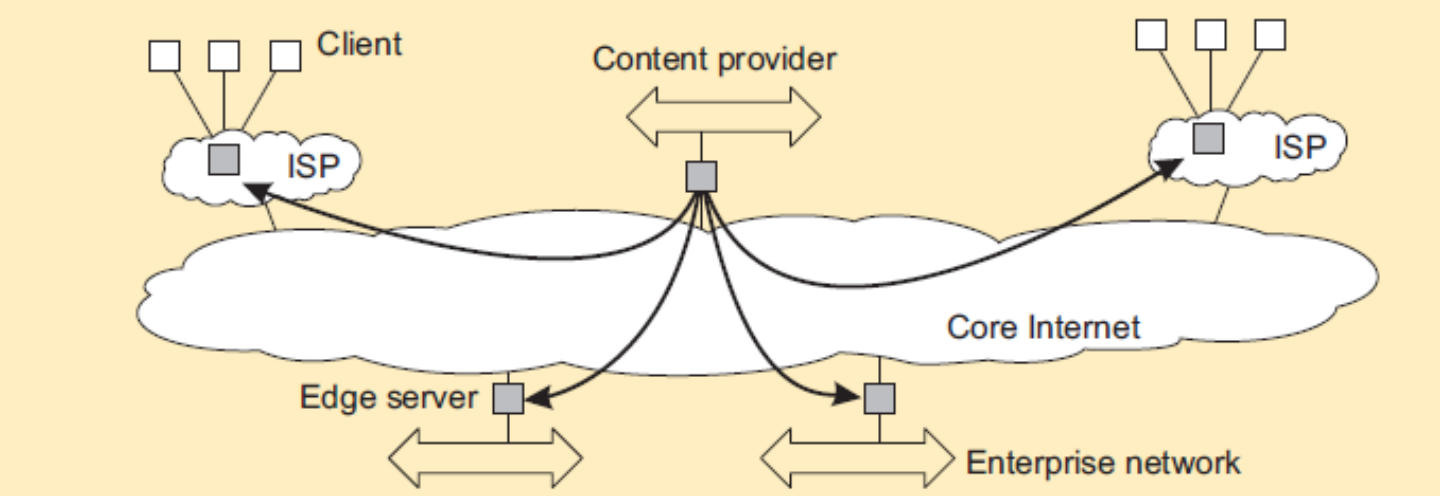


Hybrid architectures

- Specific class of distributed systems in which client server solutions are combined with decentralized architectures

Edge-server architecture

- Systems deployed on the Internet where servers are placed at the edge of the network: the boundary between enterprise networks and the actual Internet.
- In a distributed system, edge servers act as intermediaries between central servers (or cloud data centers) and the end-user devices. They can handle tasks such as content caching, data preprocessing, and running applications that need to be closer to the user.



- Rise of Fog Computing!

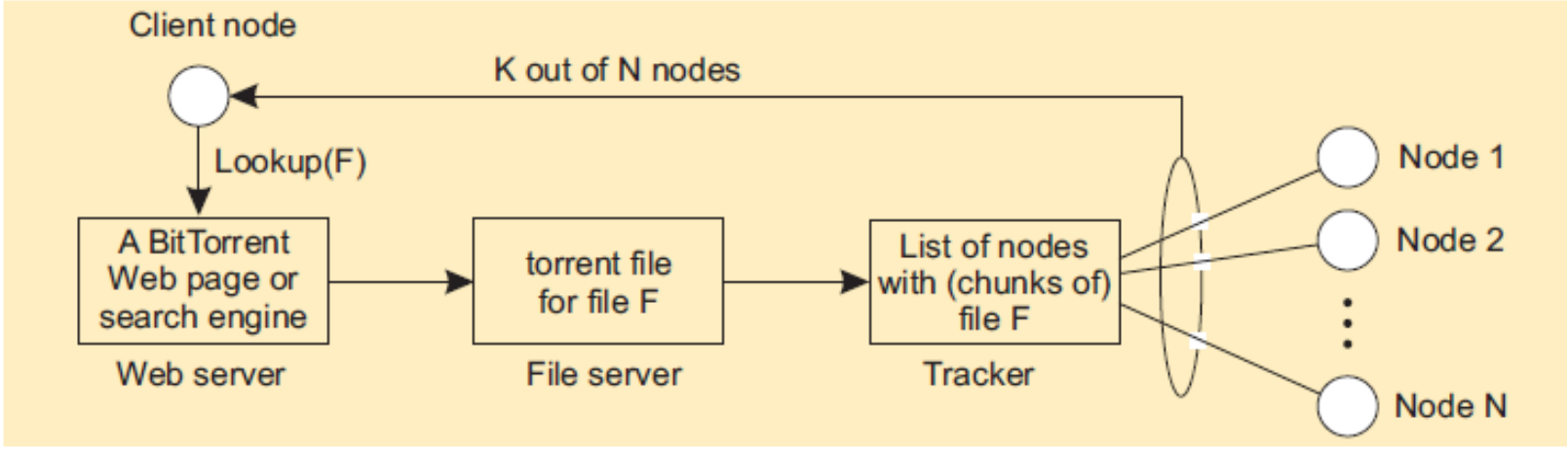
Collaborative distributed systems

- BitTorrent file sharing system – P2P file downloading system
- Design goal was **collaboration –**
  - A file can be downloaded only when the downloading client is providing content to someone else

Working principle of BitTorrent

Search for a file F:

- Lookup file at a global directory that returns the torrent file
- Torrent file contains reference to tracker: a server keeping an accurate account of active nodes that have chunks of F
- P can join swarm, get a chunk for free, and then trade a copy of that chunk for another one with a peer Q also in the swarm



Exercise

- Read more and find out
  - How edge server systems have given rise to Fog Computing?
  - How BitTorrent works?
    - https://www.beautifulcode.co/blog/58-understanding-bittorrent-protocol
  - Research about the architecture of web based distributed systems

Questions

Study architectures and system design for the following-

- Whatsapp –
  - https://www.geeksforgeeks.org/designing-whatsapp-messenger-system-design/
  - https://hackernoon.com/understanding-whatsapp-architecture
- Netflix

- UPI
  - https://www.geeksforgeeks.org/designing-upi-system-design/
  - https://www.linkedin.com/pulse/unified-payment-interface-technologists-perspective-vivek-anand/

Lecture summary

innovate achieve lead

Topics covered

- Distributed system architectures
  - Architectural styles
  - System architectures

Essential Readings

- Chapter 2 Tanenbaum
- Chapter 2 Coulouris

Thanks…

Next Lecture

- Review of RDMS concepts

Questions??