Distributed Systems

Define Distributed Systems

- Can you name some examples of distributed systems?

Some Possible Answers

- The Web
- Data Center
- The Internet
- A sensor network
- Kazaa (peer to peer overlays)
- Stock Trading System
- Cluster
- Grid
- Cloud computing system

What is a Distributed System?

Online Dictionary Definition

A collection of (probably heterogeneous) automata ***whose distribution is transparent*** to the user so that the system appears as one local machine. This is in contrast to a network, where the user is aware that there are several machines, and their location, storage replication, load balancing and functionality is not transparent. Distributed systems usually use some kind of client-server organization.

Textbook Definitions

- A distributed system is a collection of independent computers that appear to the users of the system as a single computer.

  [Andrew Tanenbaum]

- A distributed system is several computers doing something together. Thus, a distributed system has three primary characteristics: multiple computers, interconnections, and shared state.

  [Michael Schroeder]

Unsatisfactory

- Because we are interested in the insides of a distributed system
  - Design/Algorithms/Protocols
  - Implementation
  - Maintenance
  - Management

A Working Definition for Us

A distributed system is a collection of entities, each of which is autonomous, programmable, asynchronous and failure-prone, and which communicate through an unreliable communication medium.

- Entity=a process on a host
- Communication Medium=Wired or wireless network

Major Problems

- Basic Concepts [Synchronization,Consensus,…]
- Large-scale Systems [The Grid,Gnutella,Kazaa]
- Resource Management / Load Balancing
- Quality-of-Service
- Fault-Tolerance/anomaly management
- Automatic management
- Distributed file systems [NFS,AFS]
- Distributed storage
- Security

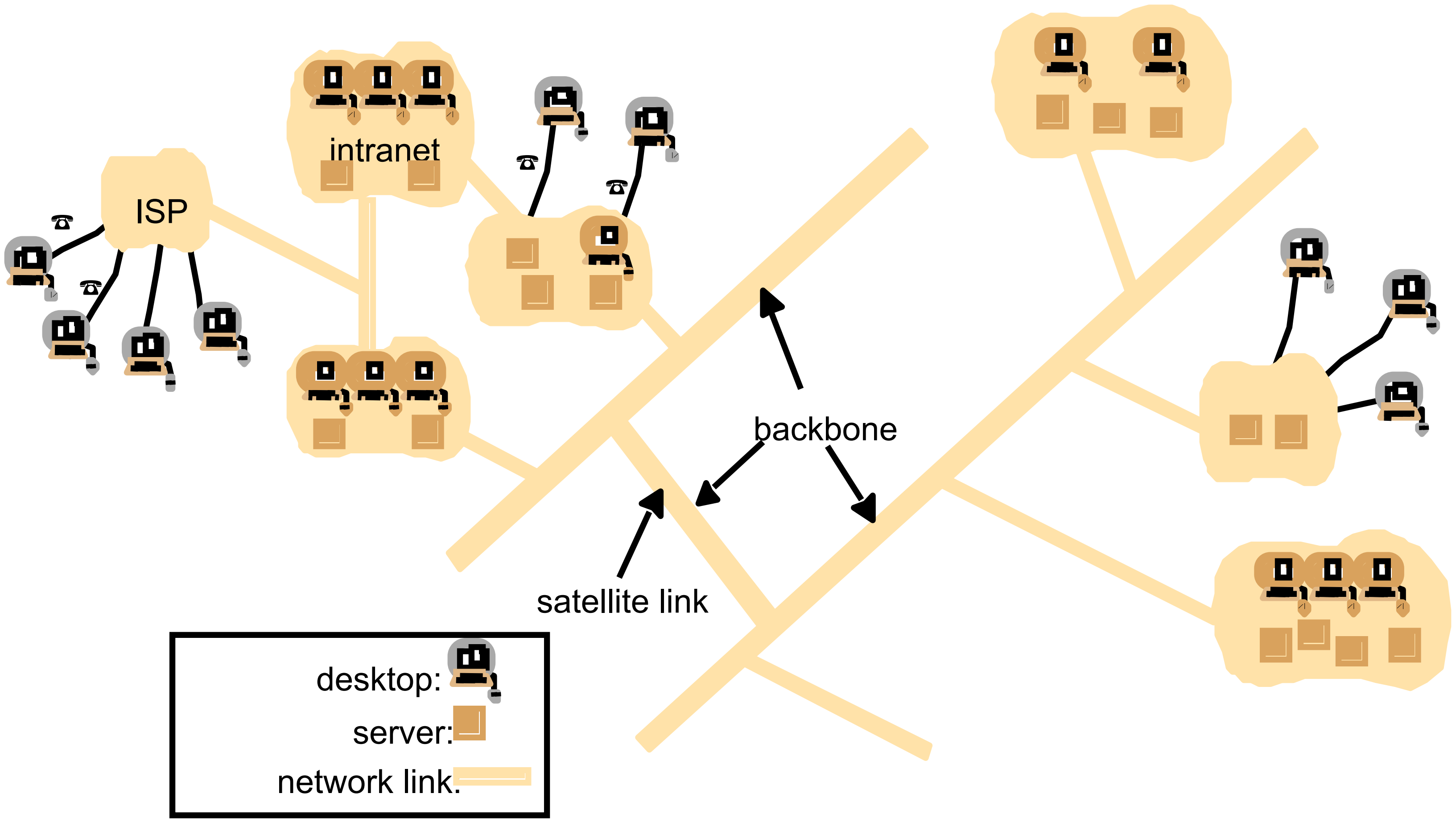
Distributed System Design Goals

- Reliability – is the system resilient to host crashes and failures, and to the network dropping messages?
- Availability – are data, services always there for clients?
- Transparency – can the system hide its internal workings from the users? i.e., Operating in such a way as to not be perceived by users.
- Heterogeneity – can the system handle different types of devices?

Distributed Systems Design Goals

- Concurrency – can the server handle multiple clients simultaneously?
- Efficiency – is the system fast enough?
- Scalability – can the system handle 100 million nodes? (nodes=clients and/or servers)
- Security – can the system withstand hacker attacks?
- Openness – is the system extensible?

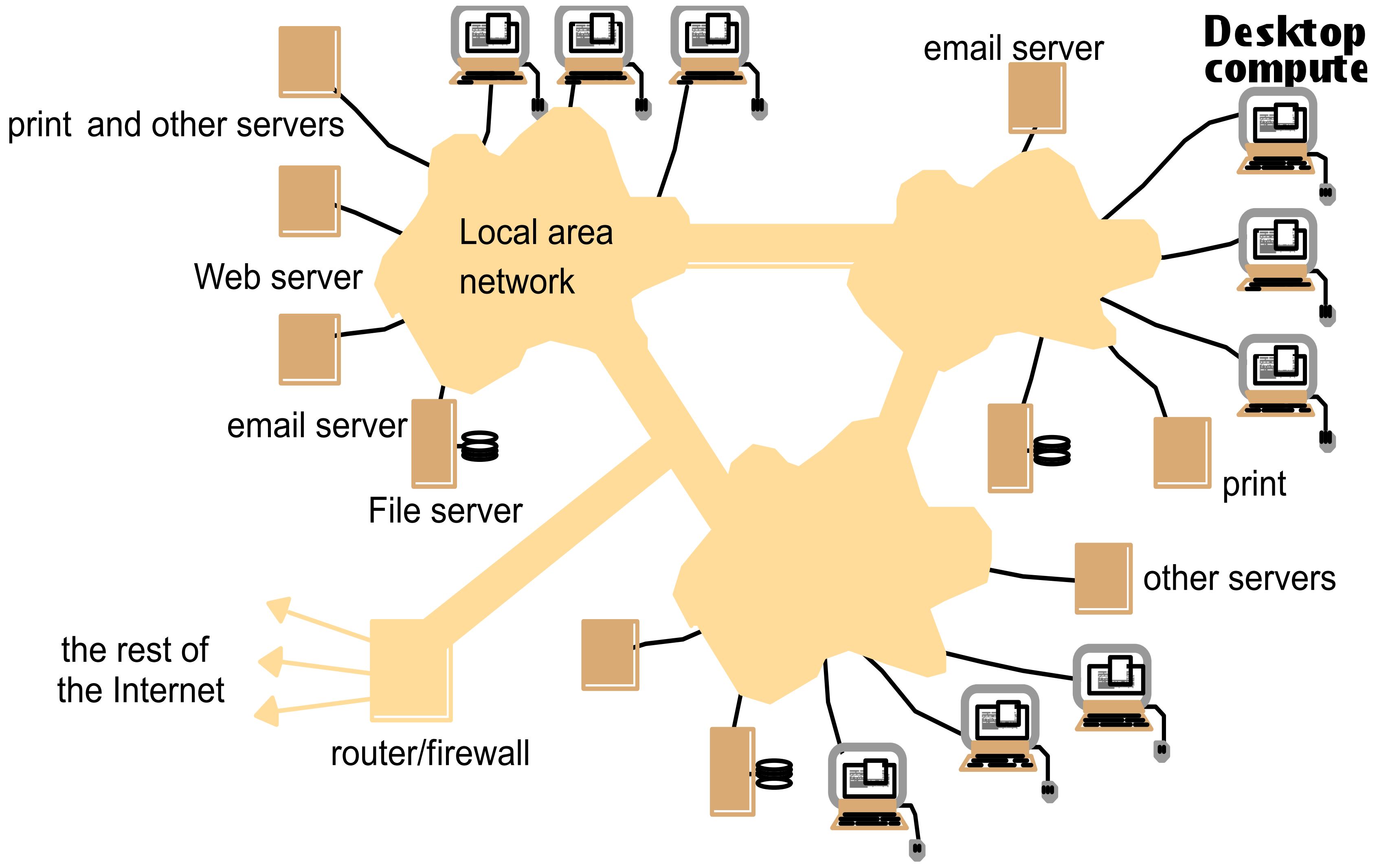
Distributed System Example -- the Internet



The Internet

- A vast interconnected collection of computer networks of many types.
- Intranets – subnetworks operated by companies and organizations.
- ISPs – companies that provide modem links and other types of connections to users.
- Intranets are linked by backbones – network links of large bandwidth, such as satellite connections, fiber optic cables, and other high-bandwidth circuits.

A Typical Intranet



Internet Apps: Their Protocols and Transport Protocols

Application	Application layer protocol	Underlying transport protocol
e-mail	smtp [RFC 821]	TCP
remote terminal access	telnet [RFC 854]	TCP
Web	http [RFC 2068]	TCP
file transfer	ftp [RFC 959]	TCP
streaming multimedia	proprietary (e.g. RealNetworks)	TCP or UDP
remote file server	NFS	TCP or UDP
Internet telephony	proprietary (e.g., Skype)	typically UDP

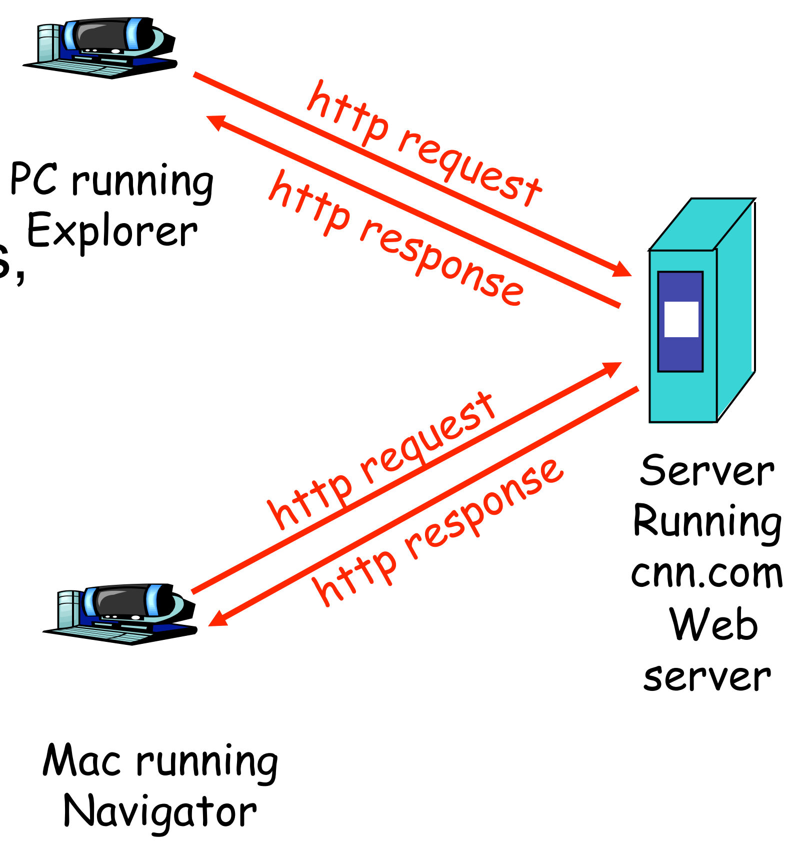
TCP=Transmission Control Protocol
UDP=User Datagram Protocol

Implemented via network "sockets". Basic primitive that allows machines to send messages to each other

WWW: the HTTP Protocol

HTTP: hypertext transfer protocol

- WWW's application layer protocol
- client/server model
  - *client:* browser that requests, receives, and "displays" WWW objects
  - *server:* WWW server stores the website, and sends objects in response to requests



The HTTP Protocol: More

http: TCP transport service:

- client initiates a TCP connection (creates socket) to server, port 80
- server accepts the TCP connection from client
- http messages (application-layer protocol messages) exchanged between browser (http client) and WWW server (http server)
- TCP connection closed

http is "stateless"

- server maintains no information about past client requests

aside

Protocols that maintain "state" are complex!

- past history (state) must be maintained
- if server/client crashes, their views of "state" may be inconsistent, and hence must be reconciled.

Distributed Software Systems

- Approaches:
  - Client/server model
  - Centralized control
  - Decentralized control
  - Peer-to-peer
  - Transactions
- ➢ Designing a distributed system is hard for several reasons. Consistency, Reliability, and availability, for instance, are hard to achieve.

Distributed Software Systems

- Lack of global knowledge and global time are particularly problematic
  - Event ordering
  - Consensus
  - Coordinated attack (e.g., DDoS) problems

Client/Server Computing

- Client machines: single-user PCs/workstations
  - user-friendly interface
- Each server provides
  - shared user services
- Server enables many clients
  - to share access to same database
  - to use high-performance computer system (manage database)

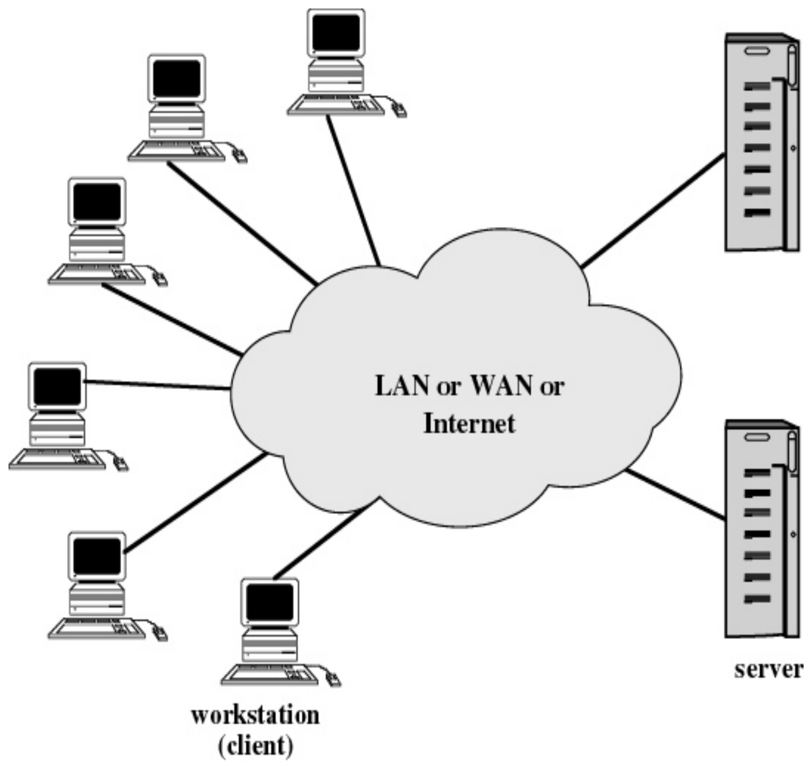
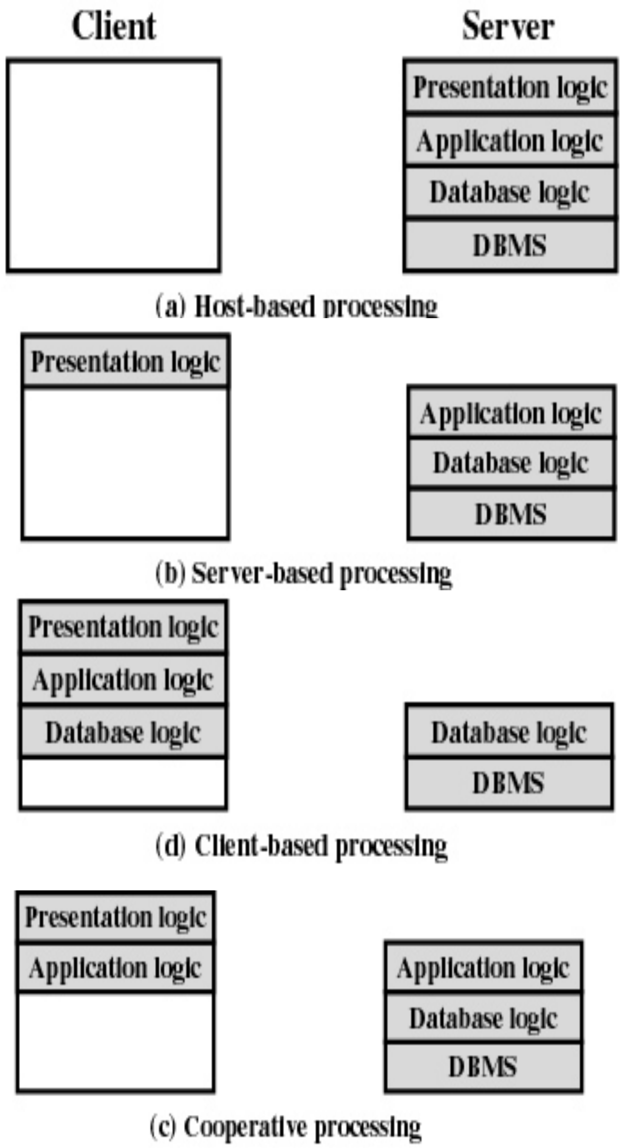


Figure 13.1 Generic Client/Server Environment

Classes of Client/Server Applications

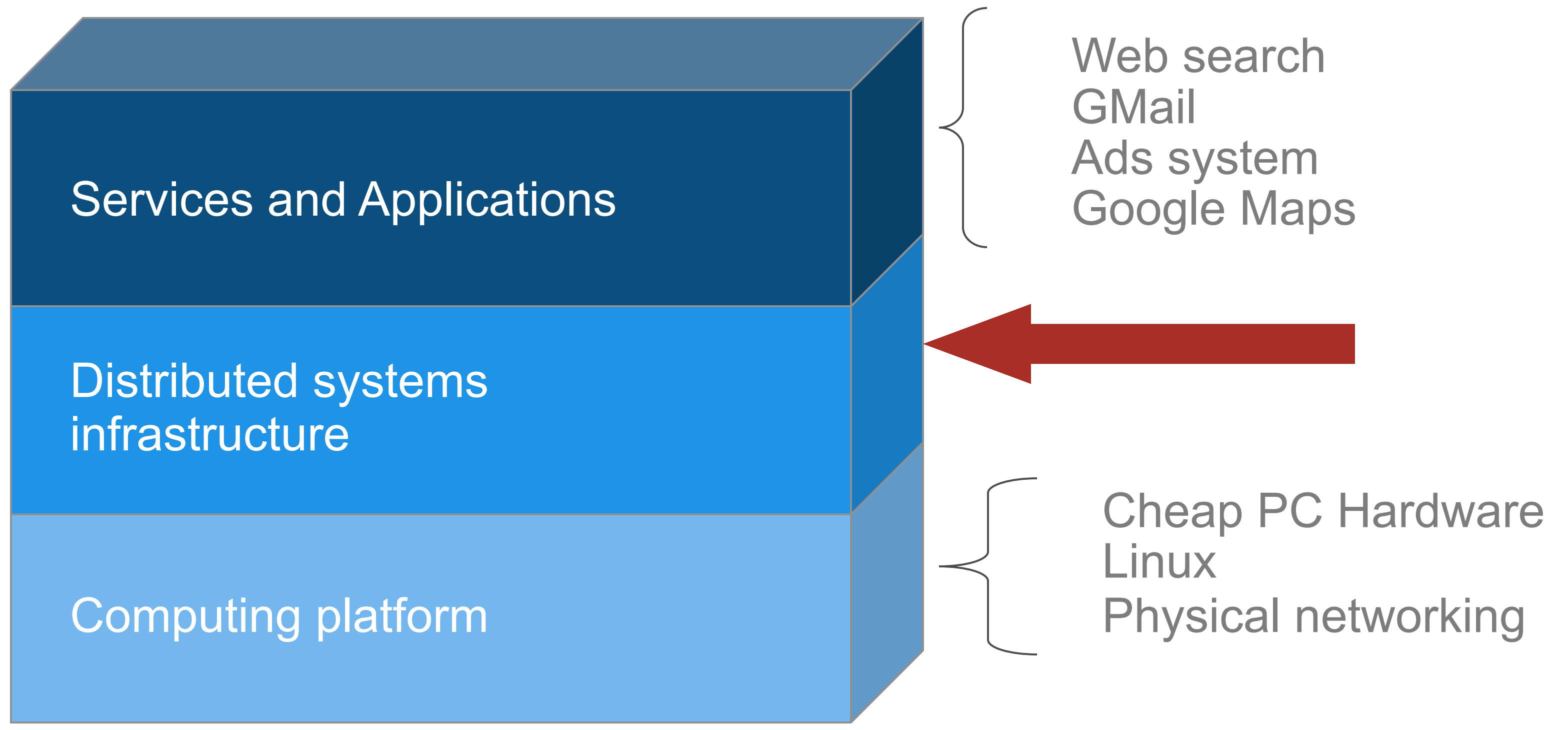
1. Host-based processing
   - not true client/server computing
   - traditional mainframe environment
2. Server-based processing
   - Server: all processing
   - User: provides graphical interface
3. Client-based processing
   - Client: all processing data
   - Server: validation /database logic
4. Cooperative processing
   - processing optimized between client and server
   - complex to set up and maintain



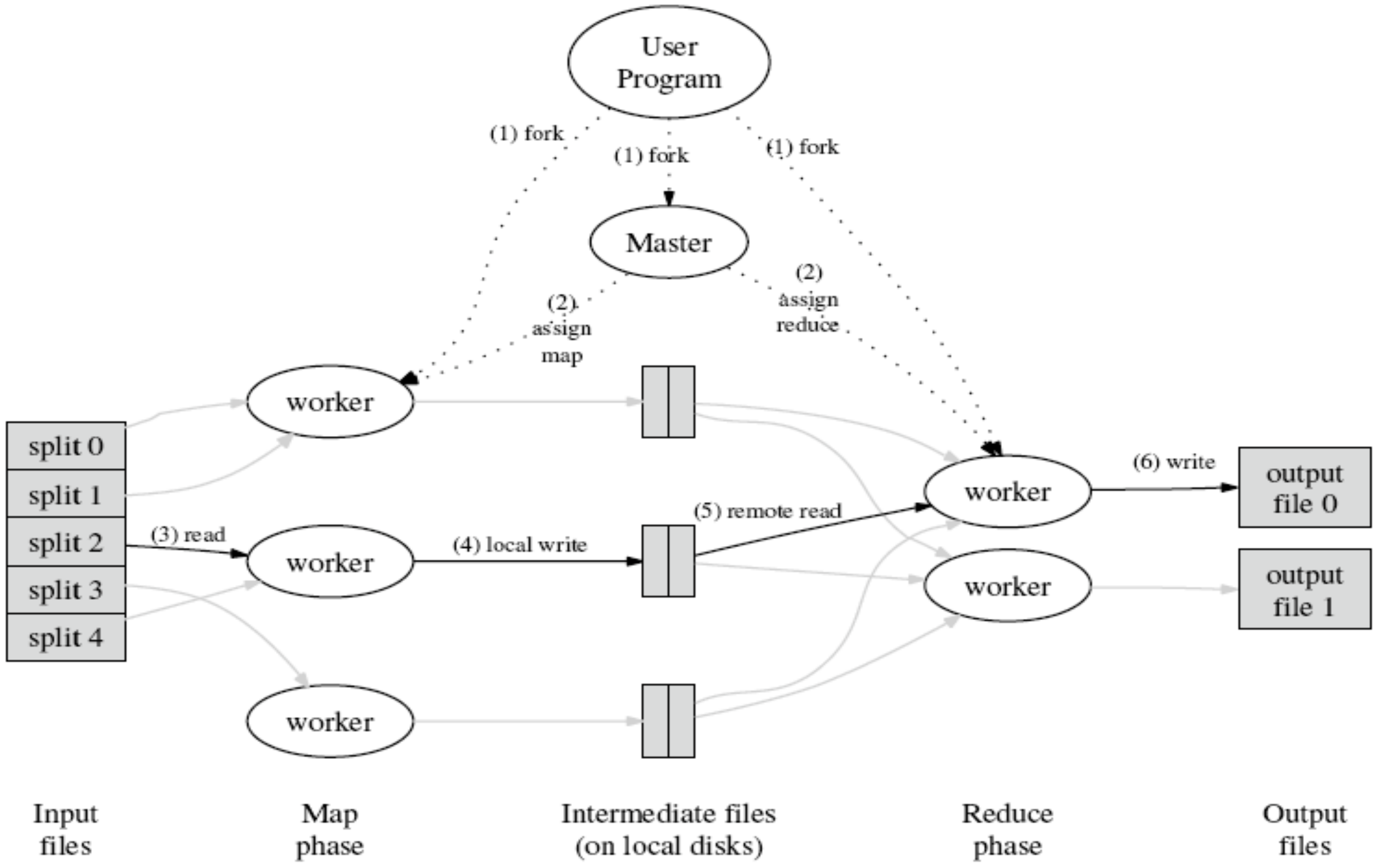
Distributed Systems in Real World

- Google infrastructure
- MapReduce massive data processing
- Multi-tier web hosting services
- Virtual Computing Labs (VCL)
- Amazon Elastic Cloud Computing (EC2)
- …

Google Technology Layers



Google's Map/Reduce Execution



Web Hosting Systems

- Application software distributed among three types of machines
  - User machine
    - thin client
  - Middle-tier server
    - Gateway
    - Convert protocols
    - Merge/integrate results from different data source
  - Backend server

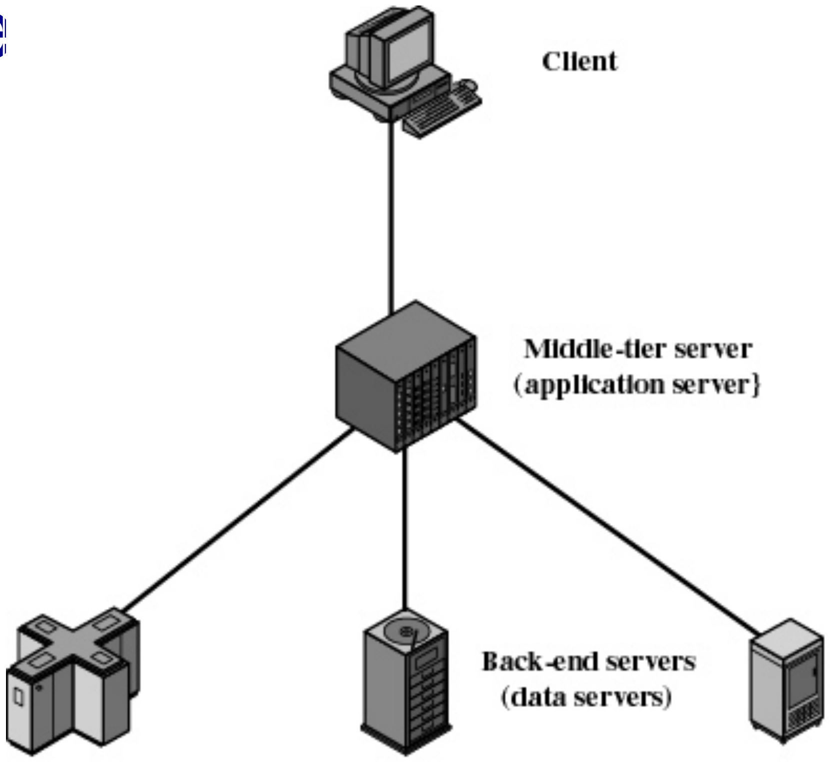


Figure 13.6 Three-tier Client/Server Architecture

Want to learn more?

- Please register CSC 724 advanced distributed systems class.