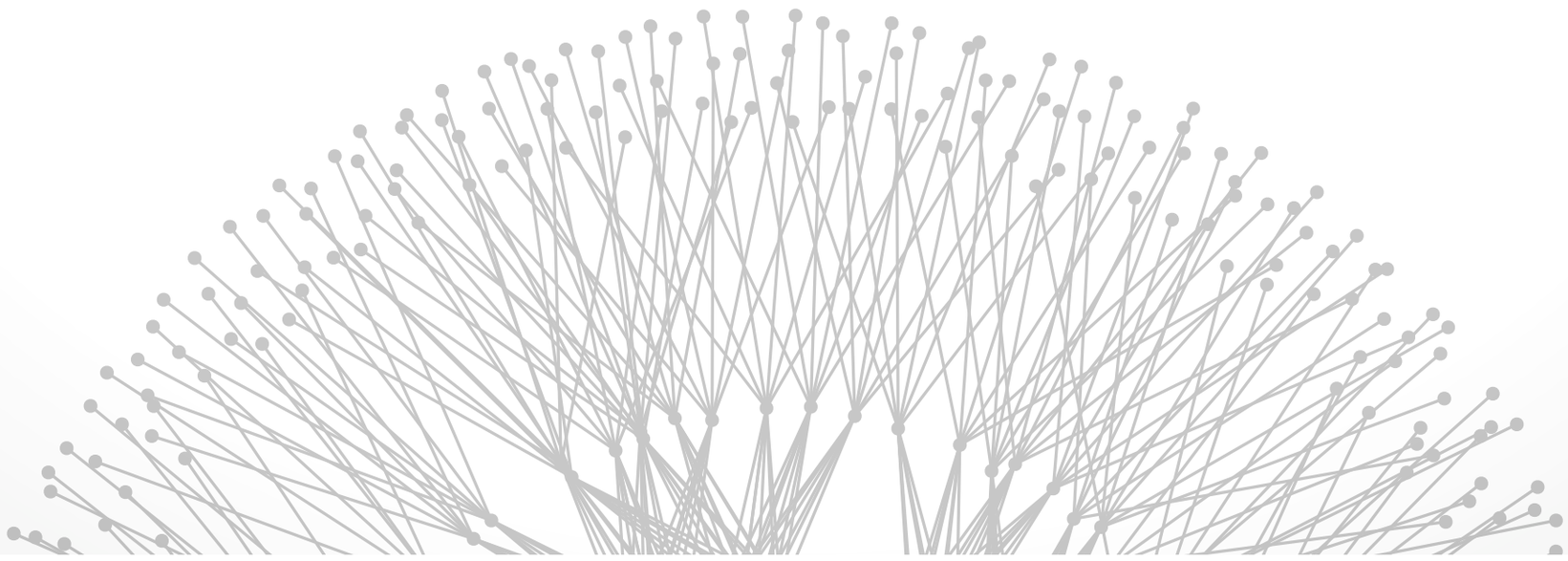
Architectural Principles

Brighten Godfrey

CS 538 September 6 2012



Cerf and Kahn: TCP/IP

Clark: TCP / IP design philosophy

Goals of the architecture

Interconnect existing networks

Survivability

Multiple communication services

Variety of networks

Distributed management

Cost effective

Easy host attachment

Resource usage accountability

Goals of the architecture

0. Interconnect existing networks

1. Survivability

2. Multiple communication services

3. Variety of networks

4. Distributed management

5. Cost effective

6. Easy host attachment

7. Resource usage accountability

0. Interconnect networks

Assumption: One common architecture

Technique: packet switching

- Met target application needs
- Already used in ARPANET, ARPA packet radio network

Interconnect with layer of gateways (packet switches)

1. Survivability

Definition: even with failures, endpoints can continue communicating without resetting high-level end-to-end conversation

- Except when?

Did this work?

1. Survivability

Key question for survivability:

Where is connection state stored?

In network

So, must replicate

- Complicated
- Does not protect against all failures

On end hosts

Shared fate

- Simpler
- If state lost, then it doesn't matter

Conclusion: stateless network, datagram packet switching

2. Multiple types of service

Initially, just TCP

But some apps do not want reliability

- VoIP
- XNET debugging protocol

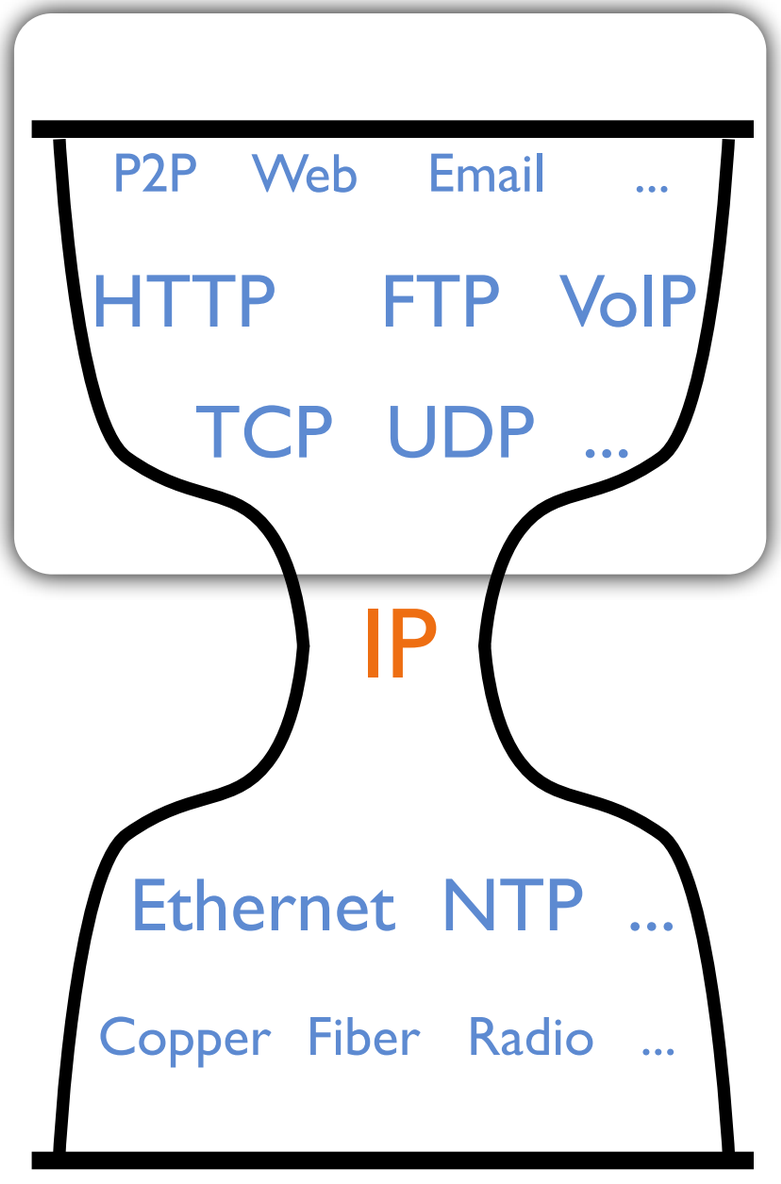
2. Multiple types of service

So, TCP/IP split

- Datagram is basic building block for many services

Still difficult to support low latency across all networks

- Hard to remove reliability if lower layer provides it



3. Variety of networks

Datagram is simple building block

Few requirements from underlying network technology

"IP over everything"

- D. Waitzman, "A Standard for the Transmission of IP Datagrams on Avian Carriers", RFC 1149

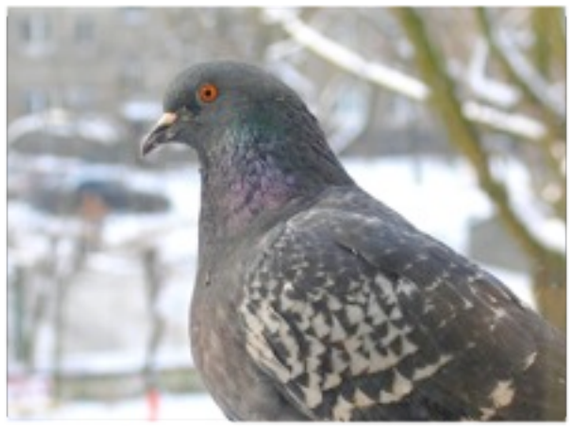
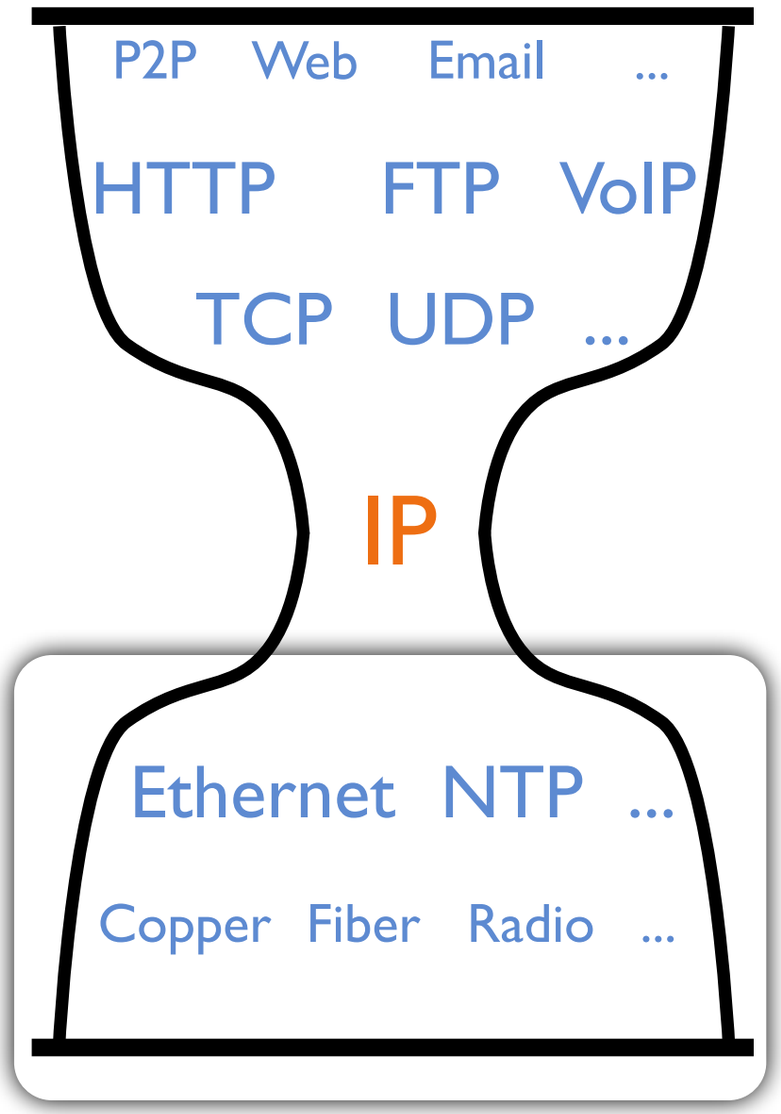


Photo: M. Betley / Wikimedia



4. Distributed management

> *... some of the most significant problems with the Internet today relate to lack of sufficient tools for distributed management, especially in the area of routing.*
>
> — David Clark, 1988

Still a problem 20+ years later!

Later in this course:
software-defined networks
ease distributed management

5. Cost effective

Inefficiencies:

- 40 byte header
- retransmission of lost packets
- How much do these matter now?

Many other sources of inefficiency

- Congestion control
- Load balancing
- Extra round trips in protocols
- ...

6. Easy host attachment

End-hosts must implement net services

Problems?

- end-host implementation complexity once caused concern to some people (end-hosts may be resource constrained)
- host misbehavior

7. Accountability

Difficult to account for who uses what resources

Today: inter-ISP transit service often priced based on 95th percentile of utilization

- Why is it only an approximation?

Both an economic and security issue

- Will return later in this course...

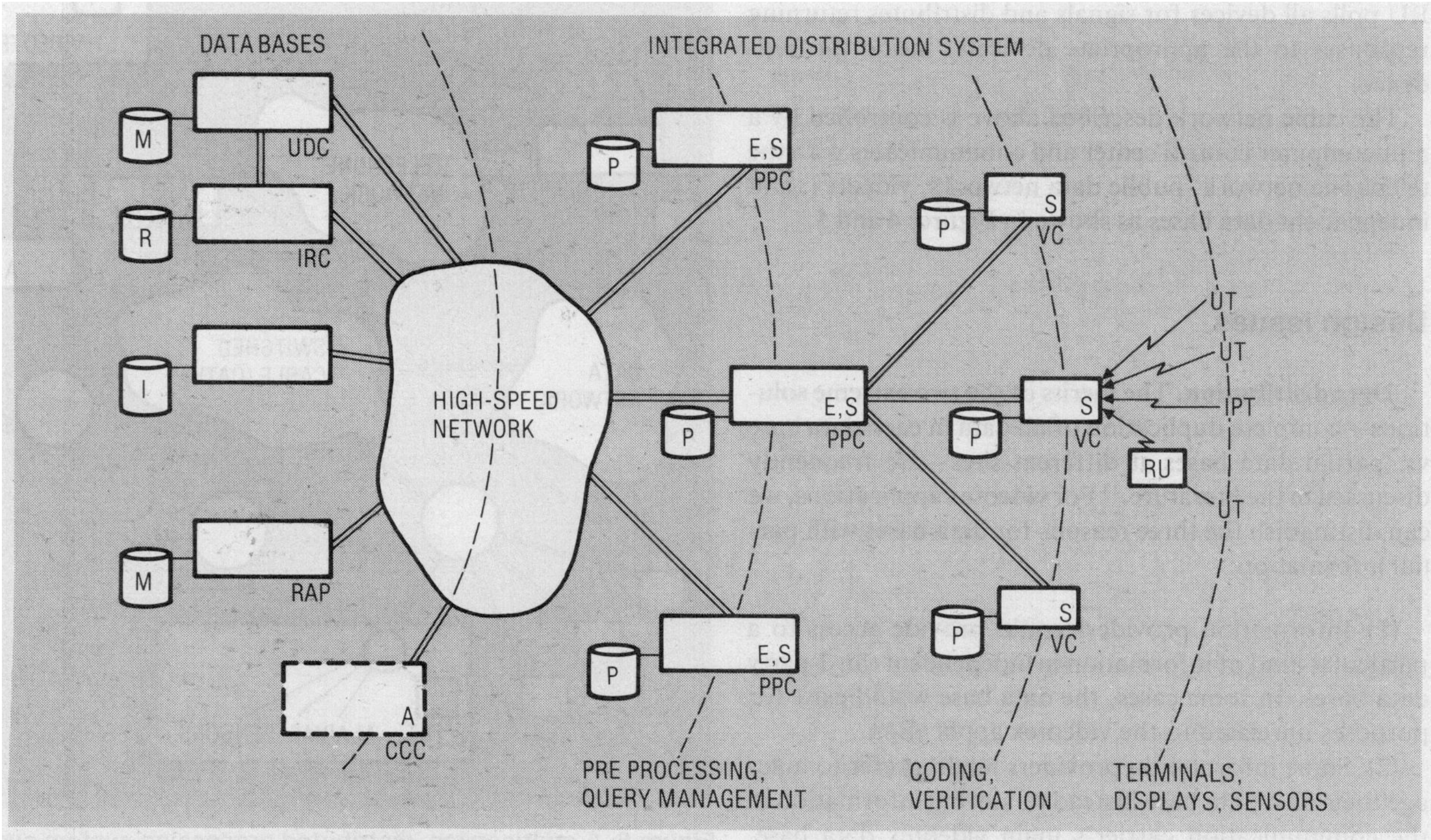
What it doesn't do

"The architecture tried very hard not to constrain the range of service which the Internet could be engineered to provide."

Extremely successful! But not as good at:

- Reporting failure ("potential for slower and less specific error detection")
- Resource management (next week!)
- Multipath forwarding
- Full illusion of reliability during failures
- Security
  - Host misbehavior and accountability discussed briefly
  - Other aspects missing

What kind of system is this?



Discussion

How would the network have been designed if the Internet were commercial?

A commercial 'internet'

Different priorities

- accountability first
- survivability & interconnection last

Example: Videotex networks

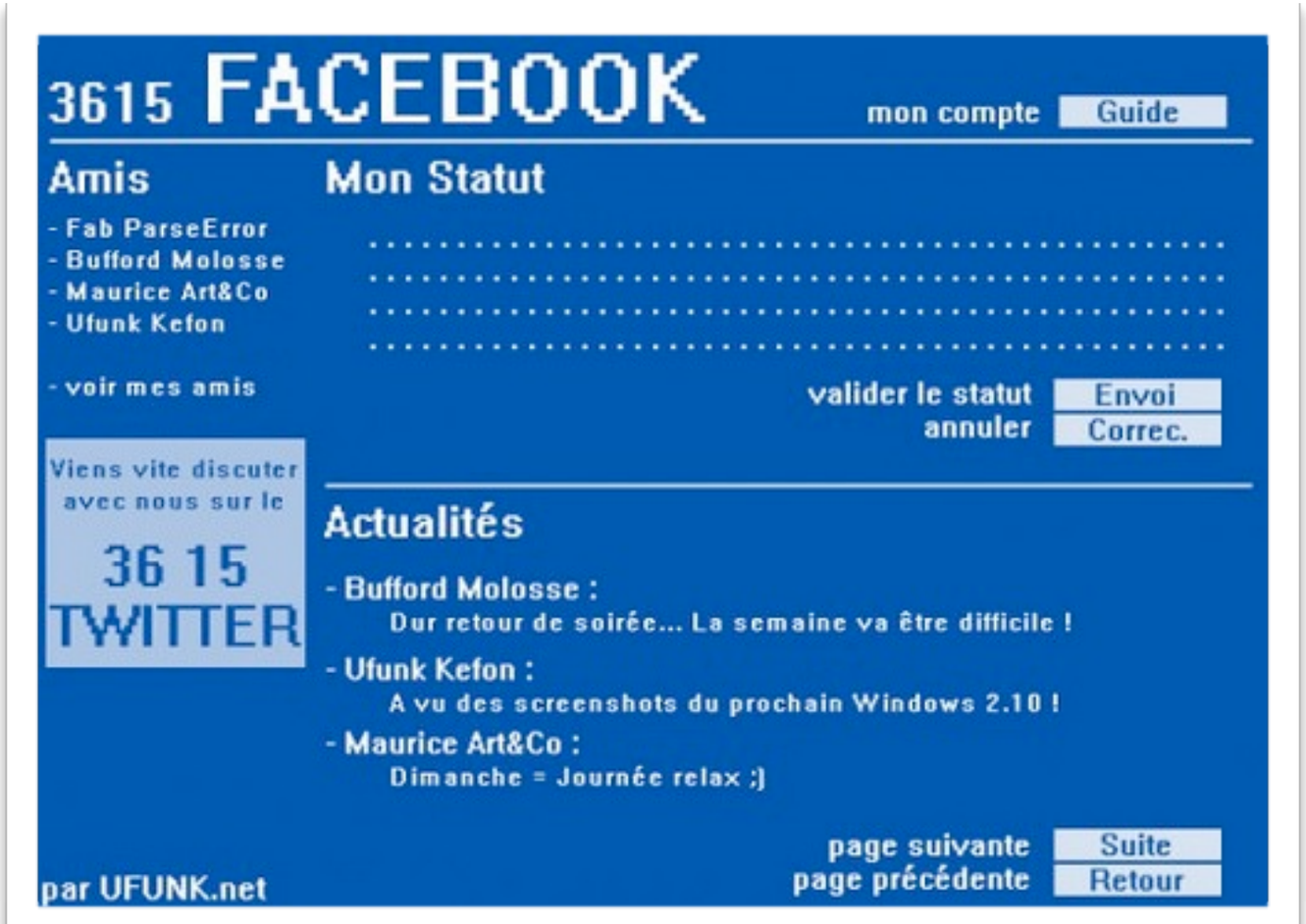
- e.g., France Telecom's Minitel

photo: wikimedia

Minitel

History

- 1972: launched
- 1995: 20 million users
- 2012 June: Terminated



[Emilie Ogez]

Services

- banking
- news
- train reservations
- adult chat
- stock transactions
- + 25,000 more services in 1995

Minitel

Architecture

- reliable
- per-minute fee
- centralized, closed
- out-evolved by the Internet

photo: wikimedia

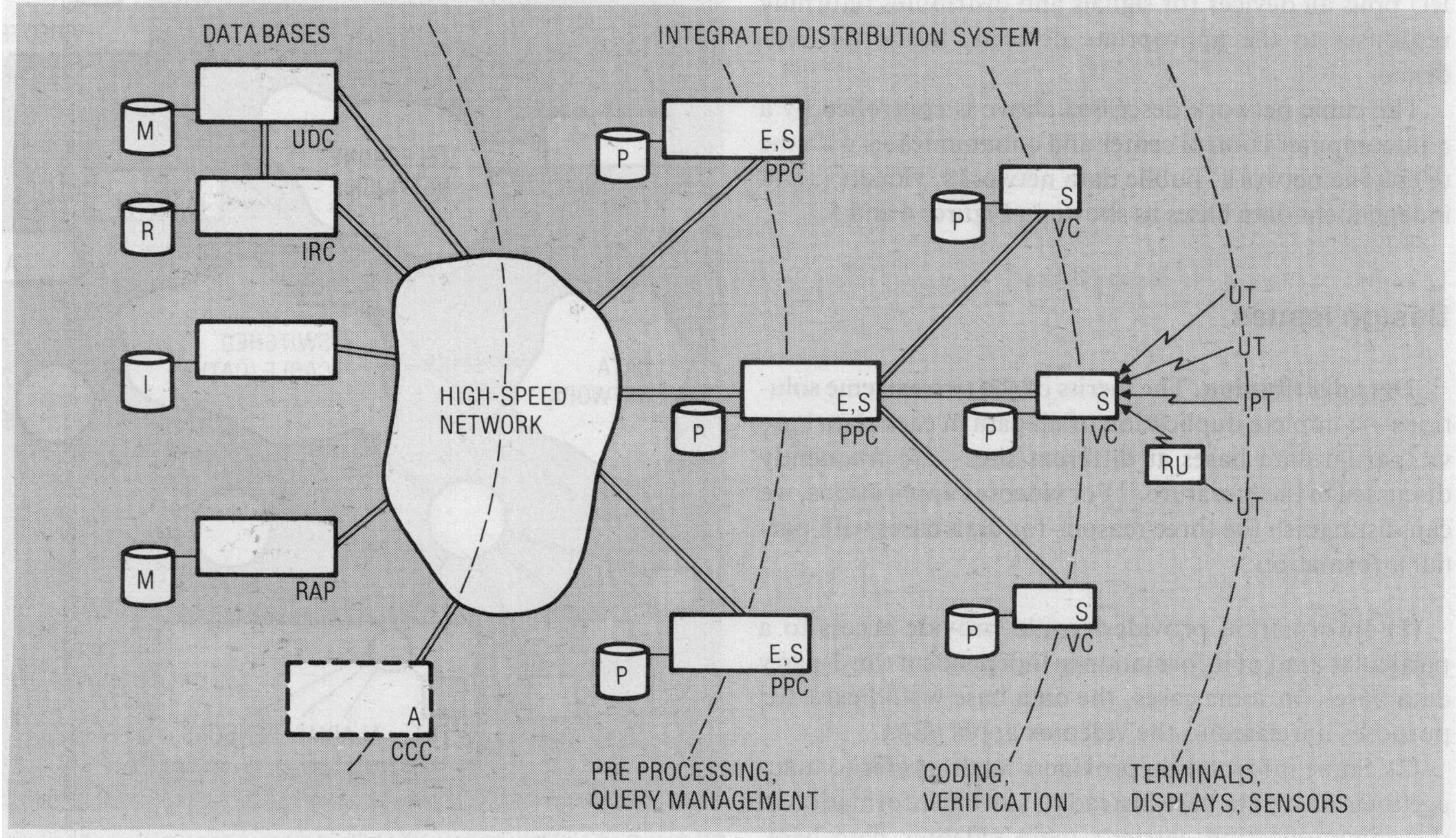


Figure 7. Generic Videotex system.

[A. J. S. Ball, G.V. Bochmann, and J. Gecsei. Videotex networks. IEEE Computer Magazine, 13(12):8–14, December 1980]

Two Architectural Principles

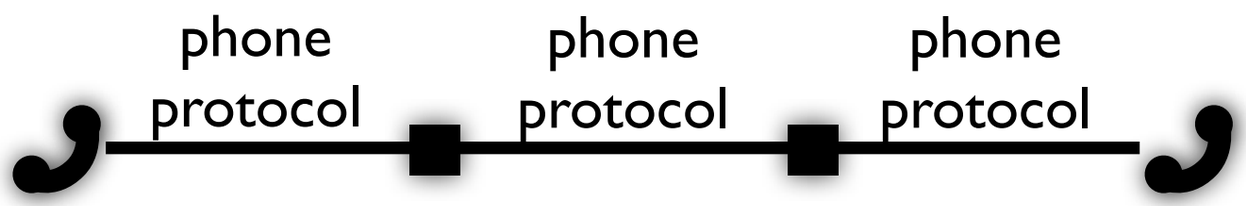
Question...

What was the key to the diversity of innovation that the Internet enabled?

- Packet switching for efficiency?
- Packet switching for resilience to nuclear attack?
- Ability to connect computers?
- Government funding?
- ...

Let's take a step back (in time)

PSTN network architecture

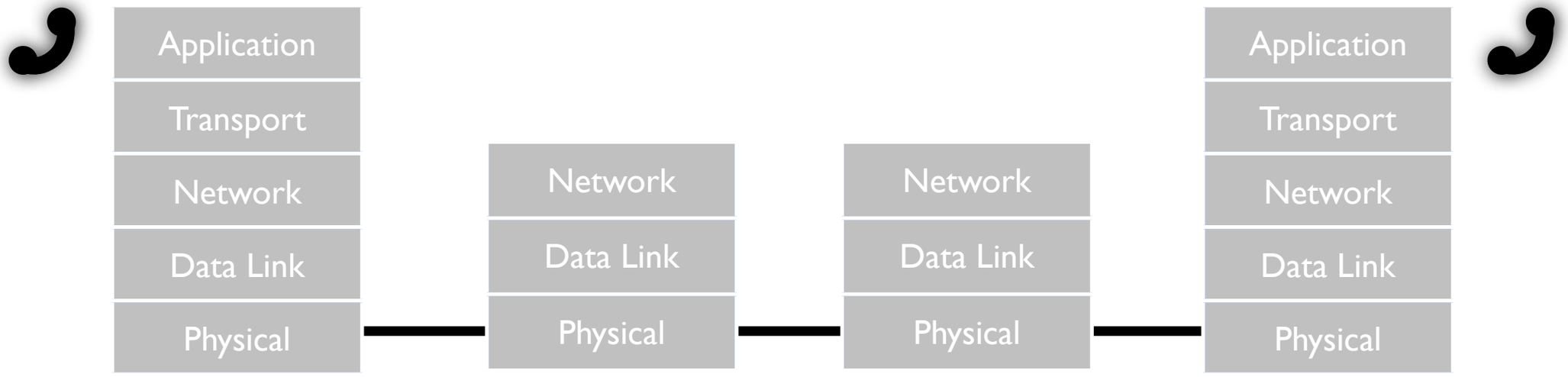


One protocol spoken by all devices

One application

What principle changed this picture?

Layering



A kind of modularity

Functionality separated into layers

- Layer *n* interfaces only with layer *n*-1
- Hides complexity of surrounding layers: enables greater diversity and evolution of modules
- (IP) connectivity becomes a commodity

Layering and innovation

Layering modularized the architecture with flexible open interfaces which helped spur innovation.

Layering before IP

Layering in ARPANET

- *"Along with the basic host-host protocol, we also envisioned a hierarchy of protocols, with Telnet, FTP and some splinter protocols as the first examples. If we had only consulted the ancient mystics, we would have seen immediately that seven layers were required."* – Stephen Crocker on the 1969 development of ARPANET [RFC1000, 1987]

Layering in computer systems

- examples?

Discussion

Layering is a guiding principle, not a law

When is layering violated? (layer *n* interacts with layers other than *n*-1 and *n*+1)

- Web-based authentication for wireless networks
- NATs
- Web caches
- ...

Organizing the layers

Layering doesn't tell you what services each layer should provide

What is an effective division of responsibility between layers?

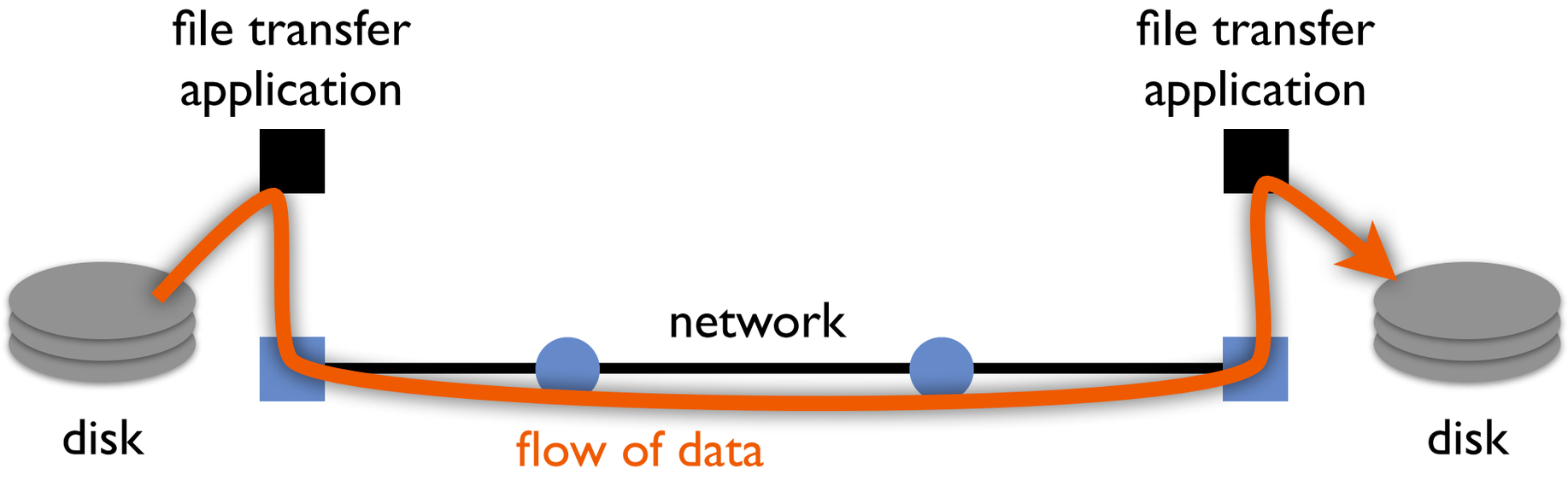
End-to-end principle

(a slight rephrasing of the paper)

If a function can completely and correctly be implemented only with the knowledge and help of the application standing at the endpoints of the communication system,

then providing that function as a feature of the communication system itself is not possible.

Example: file transfer



Suppose the link layer is reliable. Does that ensure reliable file transfer?

Suppose the network layer is reliable. Does that ensure reliable file transfer?

Interpretation

Assume the condition (if ...) holds. Then...

End-to-end implementation

- Correct
- Simplifies, generalizes lower layers

In-network implementation

- Insufficient
- May help – or hurt – performance. Examples?

Be wary to sacrifice generality for performance!

Where should these be?

Failure avoidance

Congestion control

Routing

- Topology discovery
- Path selection

Caching web requests

Evolution of architecture

ARPANET

App | App

NCP: transport connection

Reliable network

Original Cerf & Kahn

App | App

TCP

Unreliable network

Final TCP/IP

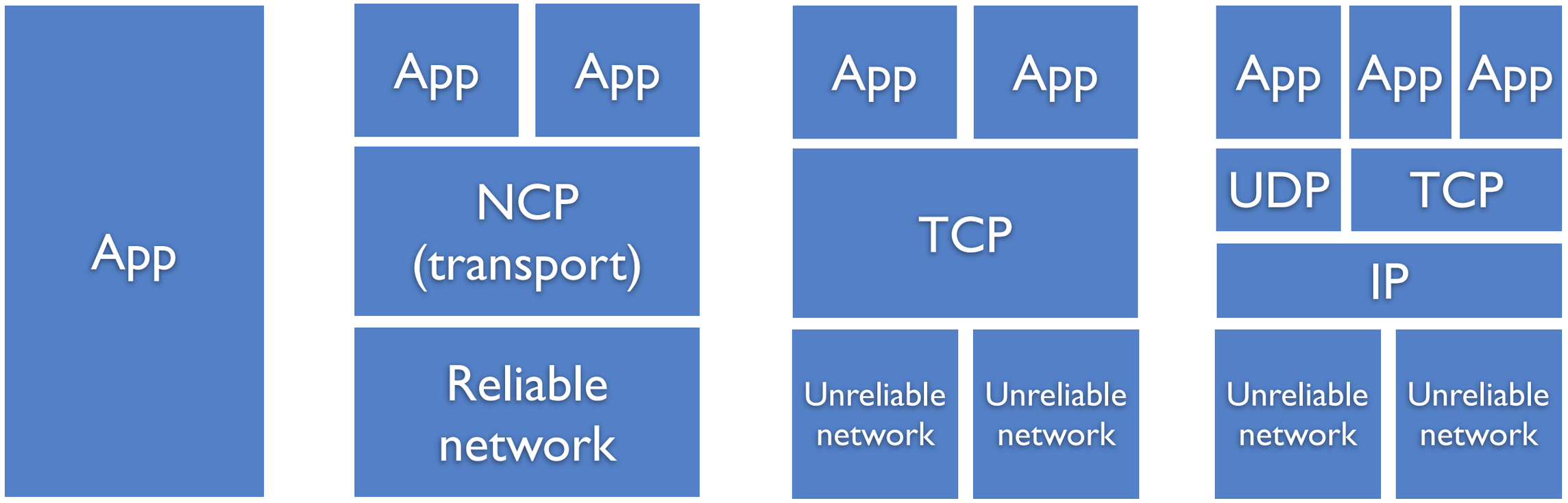
App | App | App

UDP | TCP

IP

Unreliable network

Evolution of architecture



We're done! ... right?

Two main principles

- Layering: a modular design
- End-to-end: guides what the modules should do

Is that a complete Internet architecture?

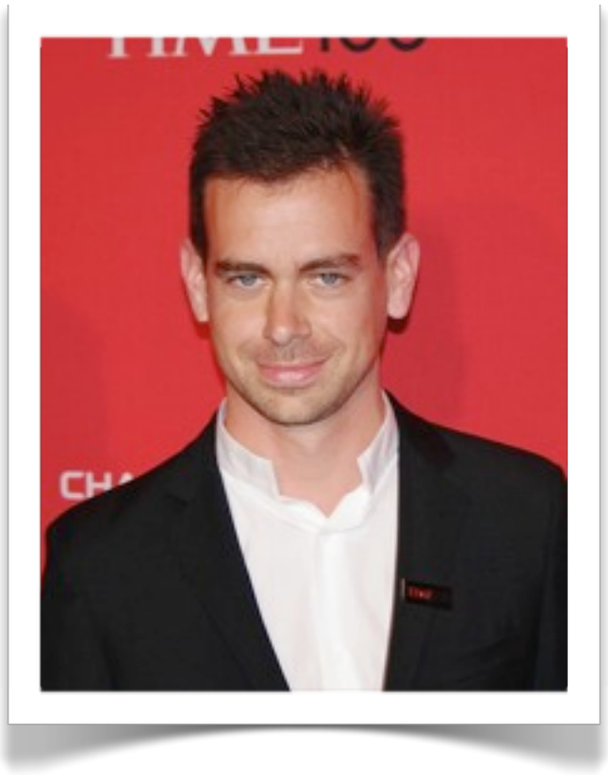
- Operations / control?
- Resource management?
- What are the right layers above, e.g. Naming?
- Routing? Security? Interaction among entities? ...

Internet experienced organic growth with fewer clear principles in other parts of the architecture

Announcements

Jack Dorsey on campus Tuesday

- Founder of Twitter, Square



Read over syllabus

Reviews for next time:

- Congestion Avoidance and Control (Jacobson 1988)
- Why flow-completion time is the right metric (RCP) (Dukkipati 2006)