Today's Agenda

- Group division (random?)
  - You can form your own group, NOW. Size at most 3.
- Internet:
  - Service perspective
  - Component perspective
  - Basic architecture
- Internet Philosophy & Design Principles
  - "end-to-end" argument: then and now

Service Perspective

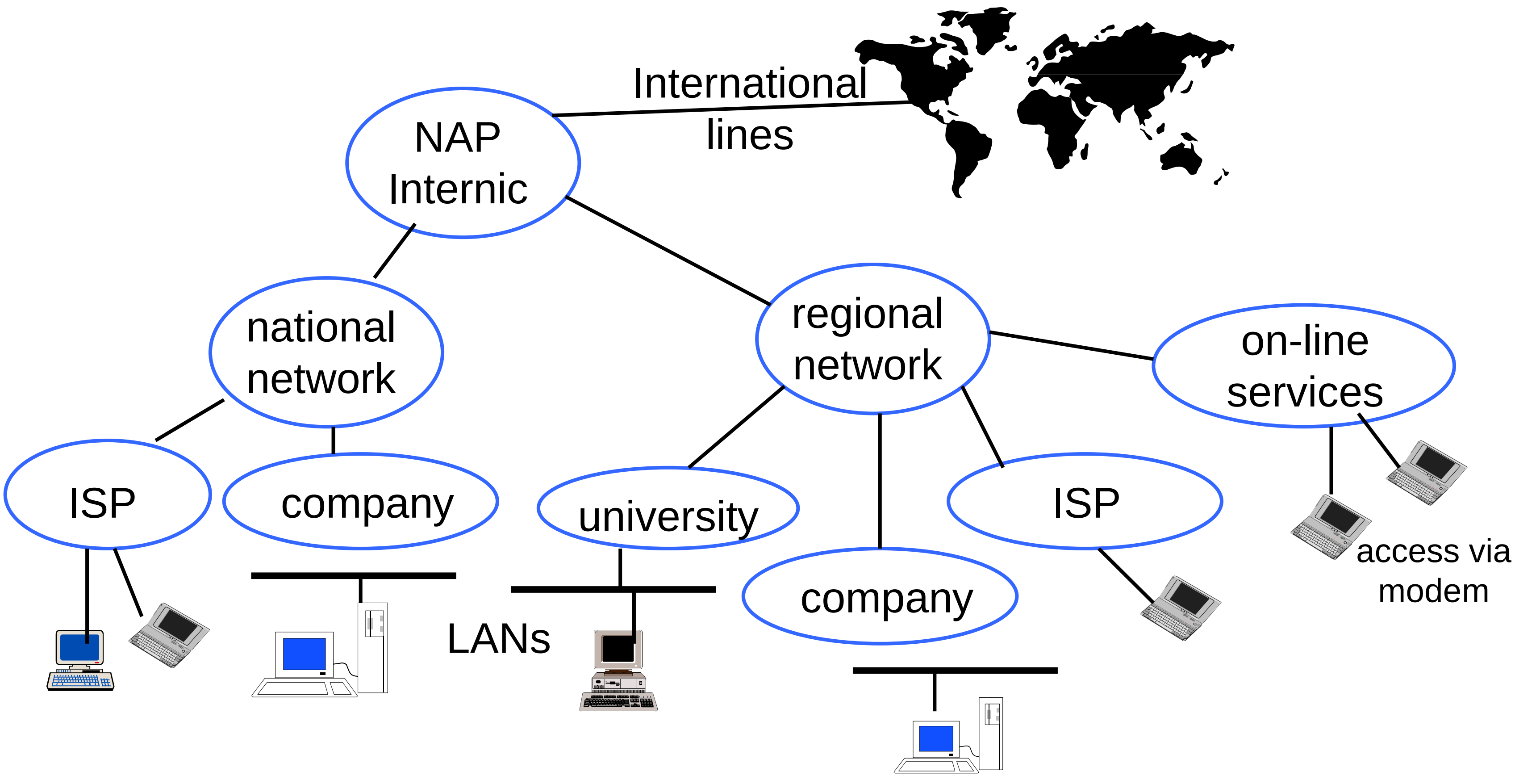
- Postal: deliver mail/package from people to people
  - First class, express mail, bulk rate, certified, registered, ...
- Telephone: connect people for talking
  - You may get a busy dial tone
  - Once connected, consistently good quality, unless using cell phones
- Internet: transfer information between people/machines
  - Reliable connection-oriented or unreliably connectionless services!
  - You never get a busy dial tone, but things can be very slow!
  - You can't ask for express delivery (not at the moment at least!)

Component Perspective

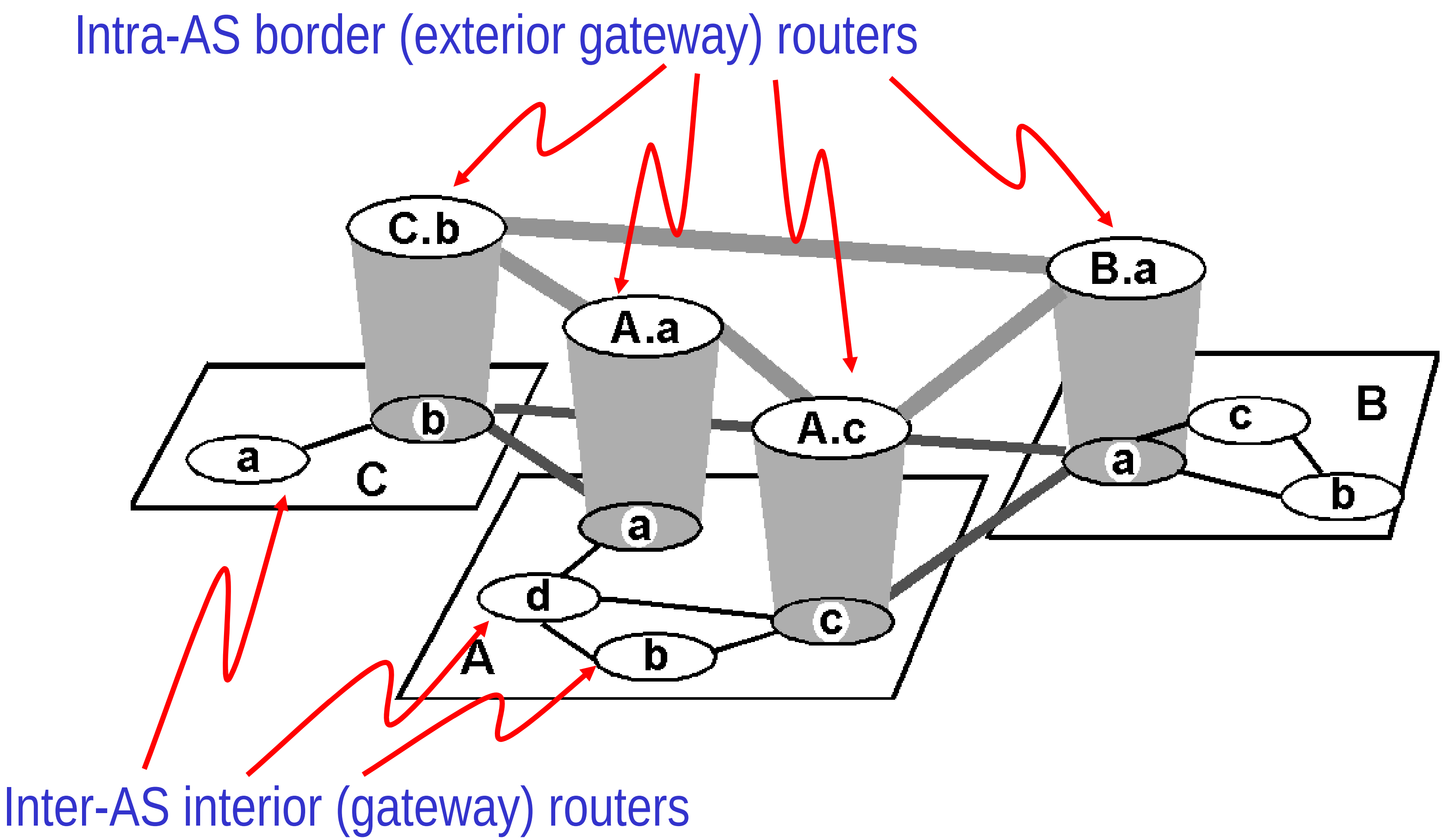
- Nodes:
  - Hosts (or end systems): PCs, laptops, servers, PDAs, ...
  - Switches: routers, hubs, switches ...
- Links:
  - Coaxial, twisted pair cables, optical fibers, wireless
  - Point to point or multiple access
- Nodes connected via links to form a network
  - LAN, WAN, MAN
- Internet: network of networks

A Simplified Picture of the Internet

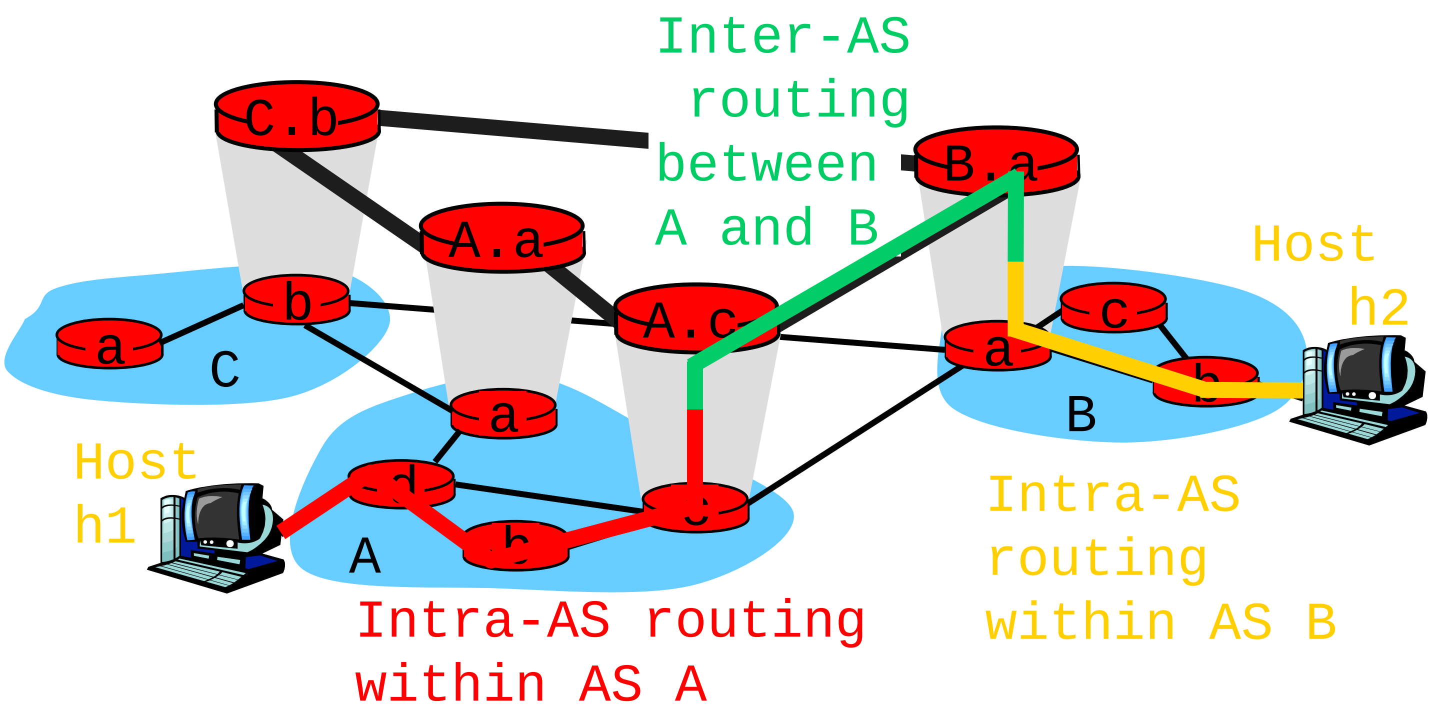
Internet: "network of networks"!



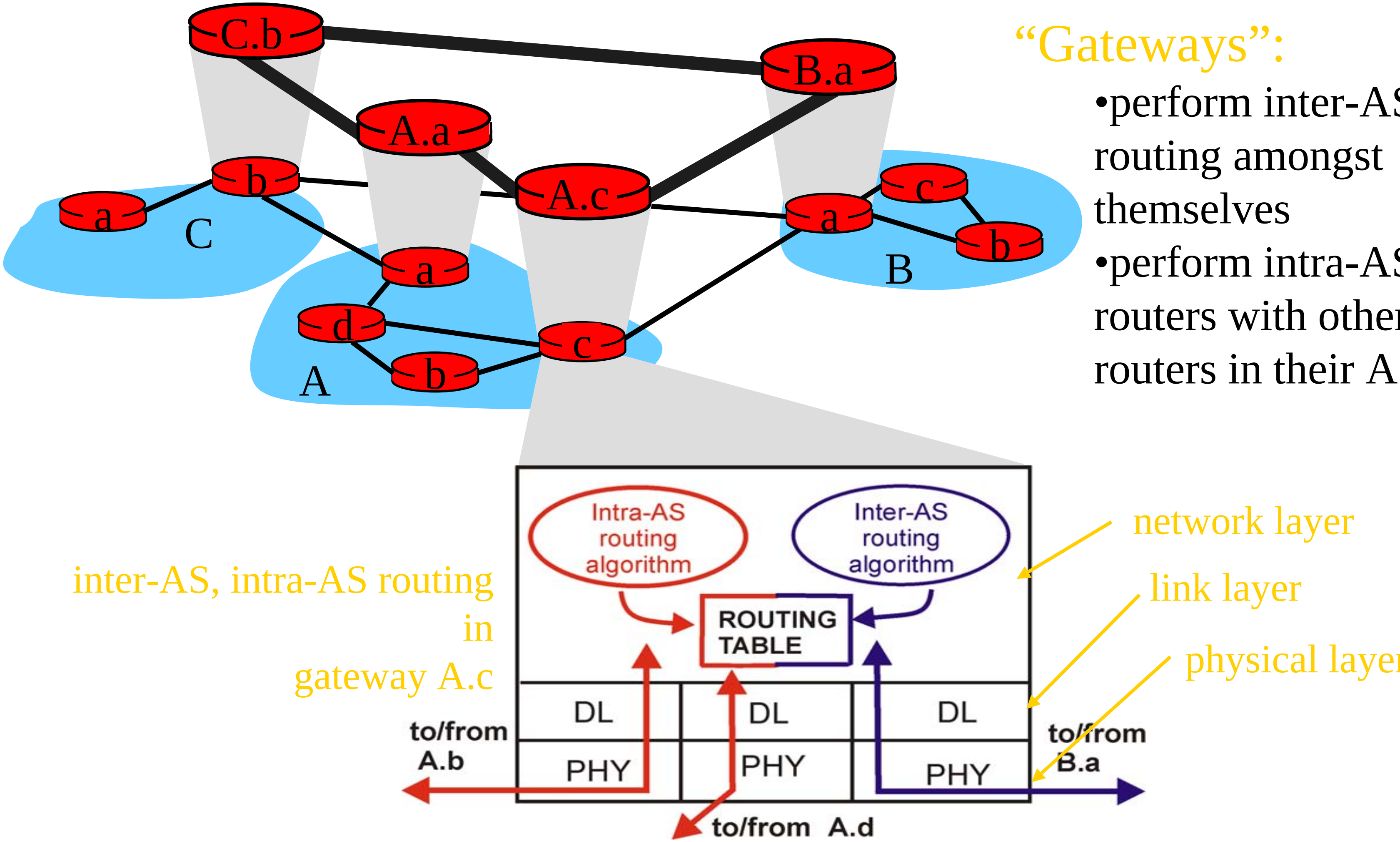
Internet AS Hierarchy



Intra-AS vs. Inter-AS Routing



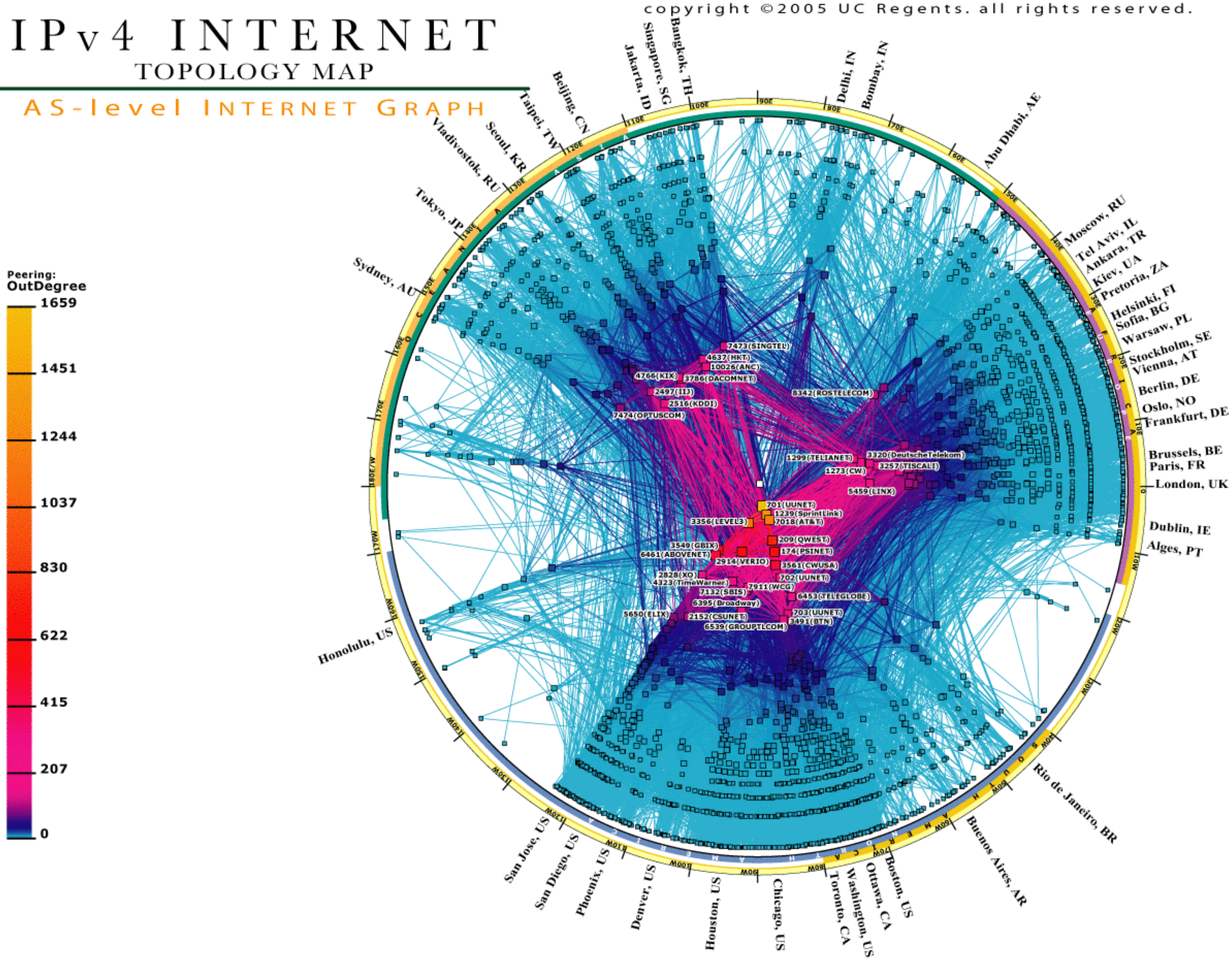
Intra-AS and Inter-AS Routing



"Gateways":

- perform inter-AS routing amongst themselves
- perform intra-AS routers with other routers in their AS

IPv4 INTERNET
TOPOLOGY MAP
AS-level INTERNET GRAPH



Fundamental Issues in Networking

- Naming/Addressing
  - How to find name/address of the party (or parties) you would like to communicate with
  - Address: byte-string that identifies a node
  - Types of addresses
    - Unicast: node-specific
    - Broadcast: all nodes in the network
    - Multicast: some subset of nodes in the network
- Routing/Forwarding:
  - Process of determining how to send packets towards the destination based on its address
  - Finding out neighbors, building routing tables

Fundamental Problems in Networking

What can go wrong?

- Bit-level errors: due to electrical interferences
- Packet-level errors: packet loss due to buffer overflow/congestion
- Out of order delivery: packets may takes different paths
- Link/node failures: cable is cut or system crash

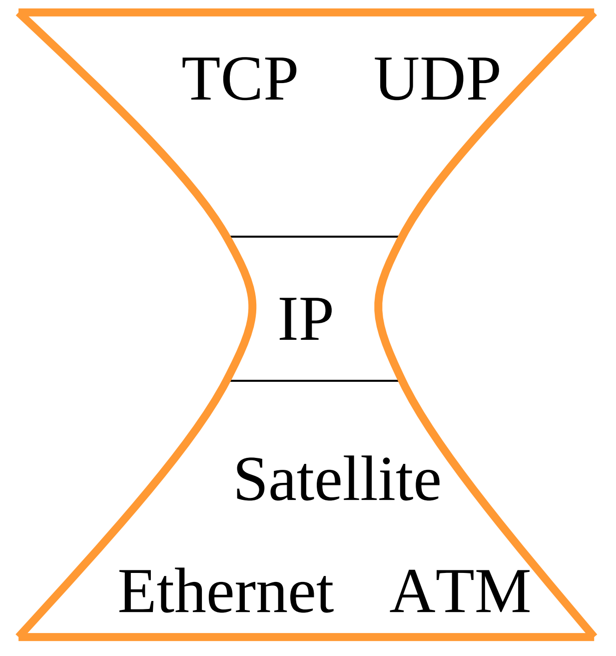
- What else?

Switching & Multiplexing

- Network is a shared resource
  - Provide services for many people at same time
  - Carry bits/information for many people at same time
- How do we do it?
  - Switching: how to deliver information from point A to point B?
  - Multiplexing: how to share resources among many users
- Current Internet:
  - Packet switching, statistical multiplexing
  - Circuit & virtual circuit switching at the core

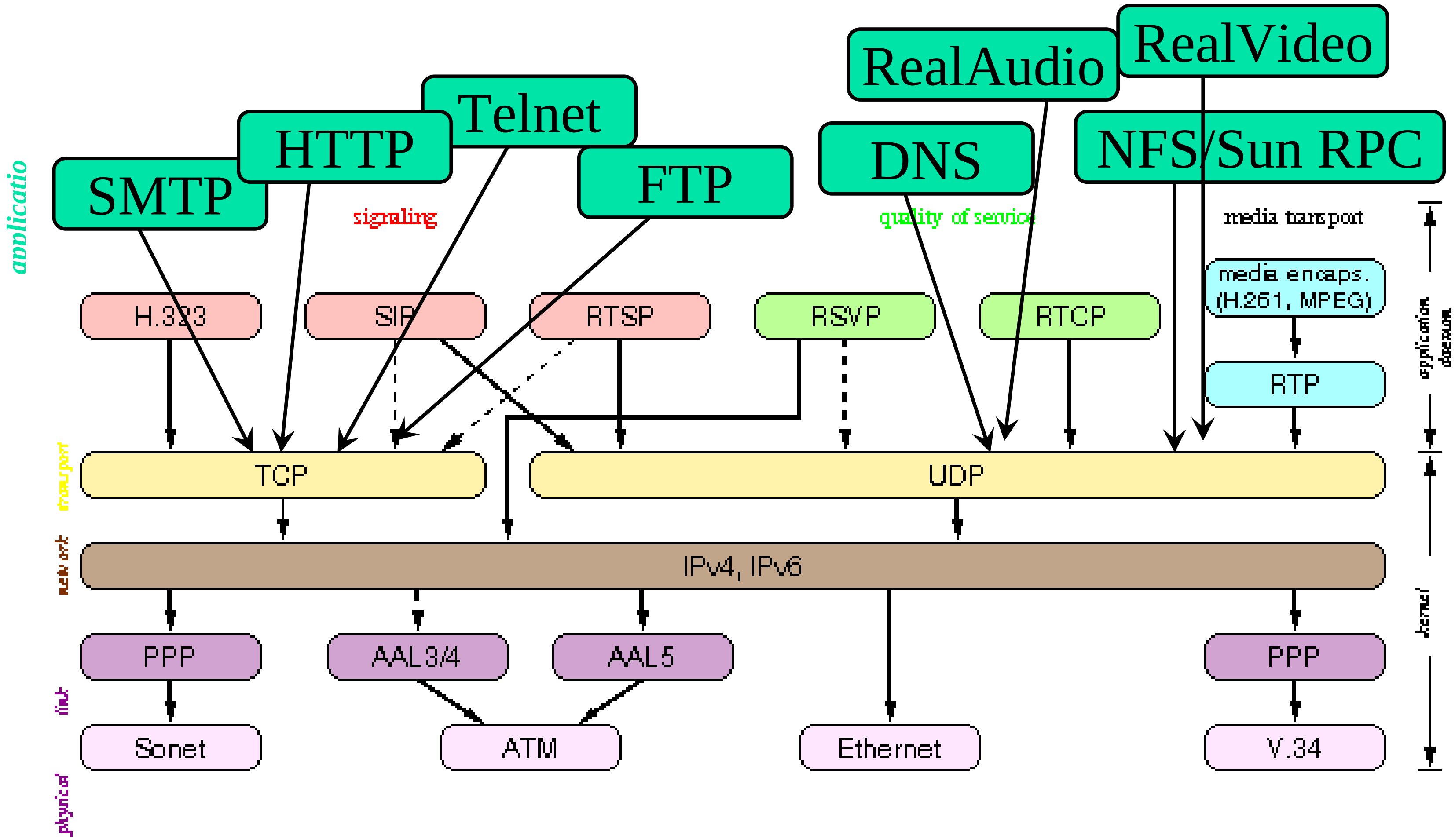
Internet Architecture

- Packet-switched datagram network
- IP is the glue (network layer overlay)
- IP hourglass architecture
  - all hosts and routers run IP
- Stateless architecture
  - No per flow state inside network

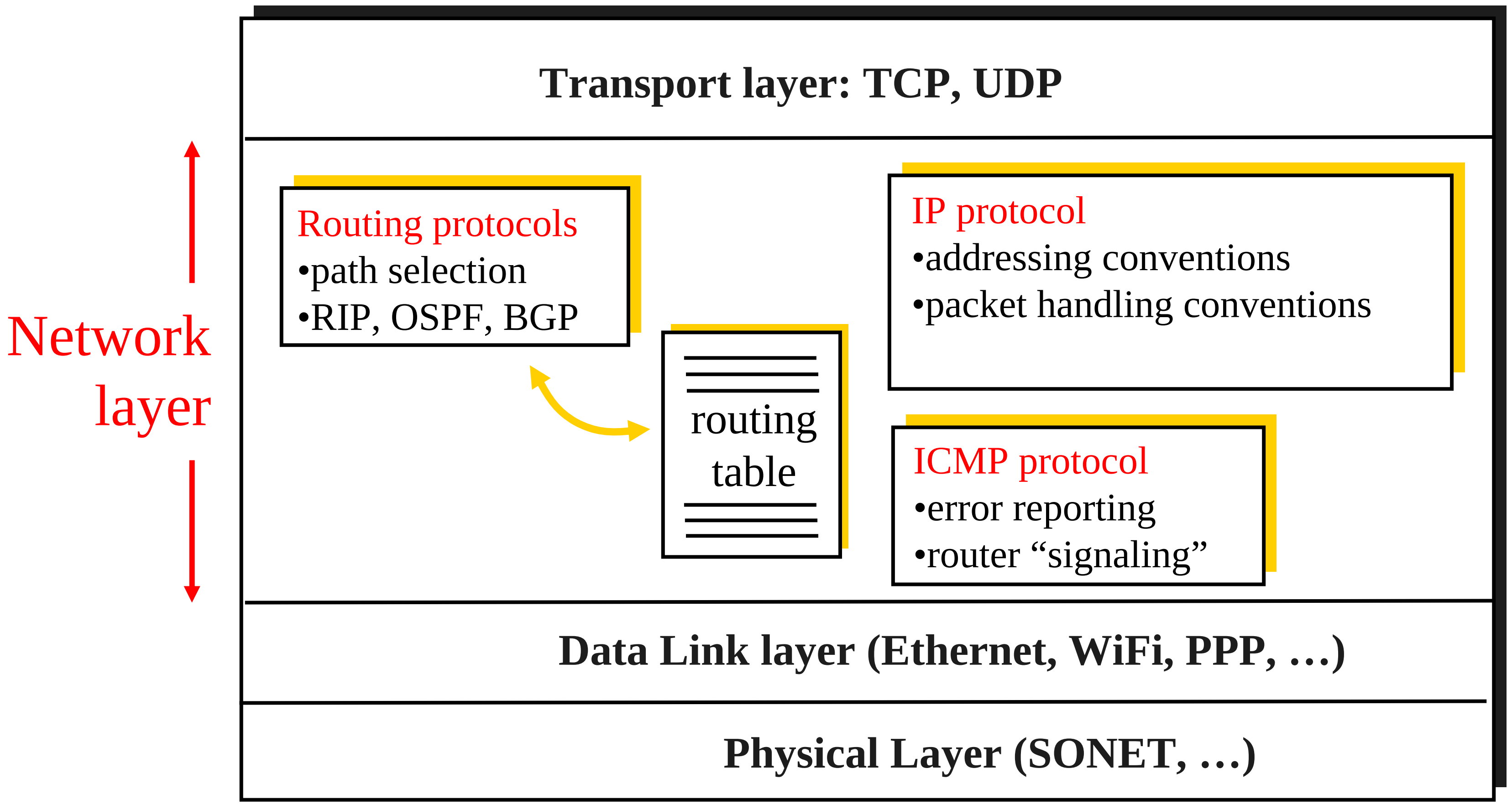


IP hourglass

Internet Protocol “Zoo”



The Internet Network layer



Tips and tricks 6

- What is IP Smurfing?

Internet Philosophy and Design Principles

Goals:

- identify, study principles that can guide network architecture
- "bigger" issues than specific protocols or implementation tricks

Key Questions:

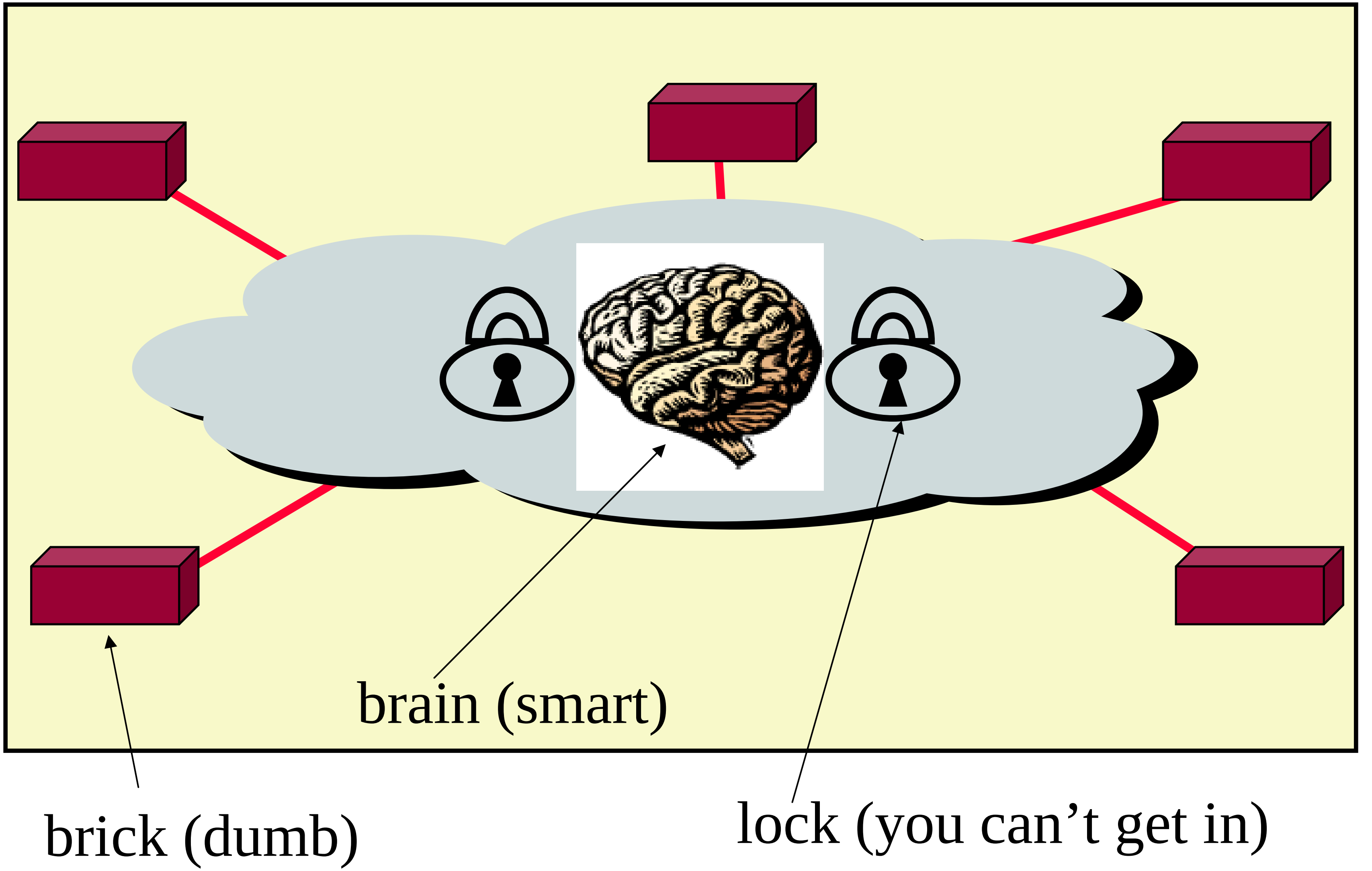
- How to decompose the complex system functionality into protocol layers?
- Why even do layering in the first place?
- *Which* functions placed *where* in network, at which layers?
- Can a function be placed at multiple levels ?

Internet End-to-End Argument/Principle

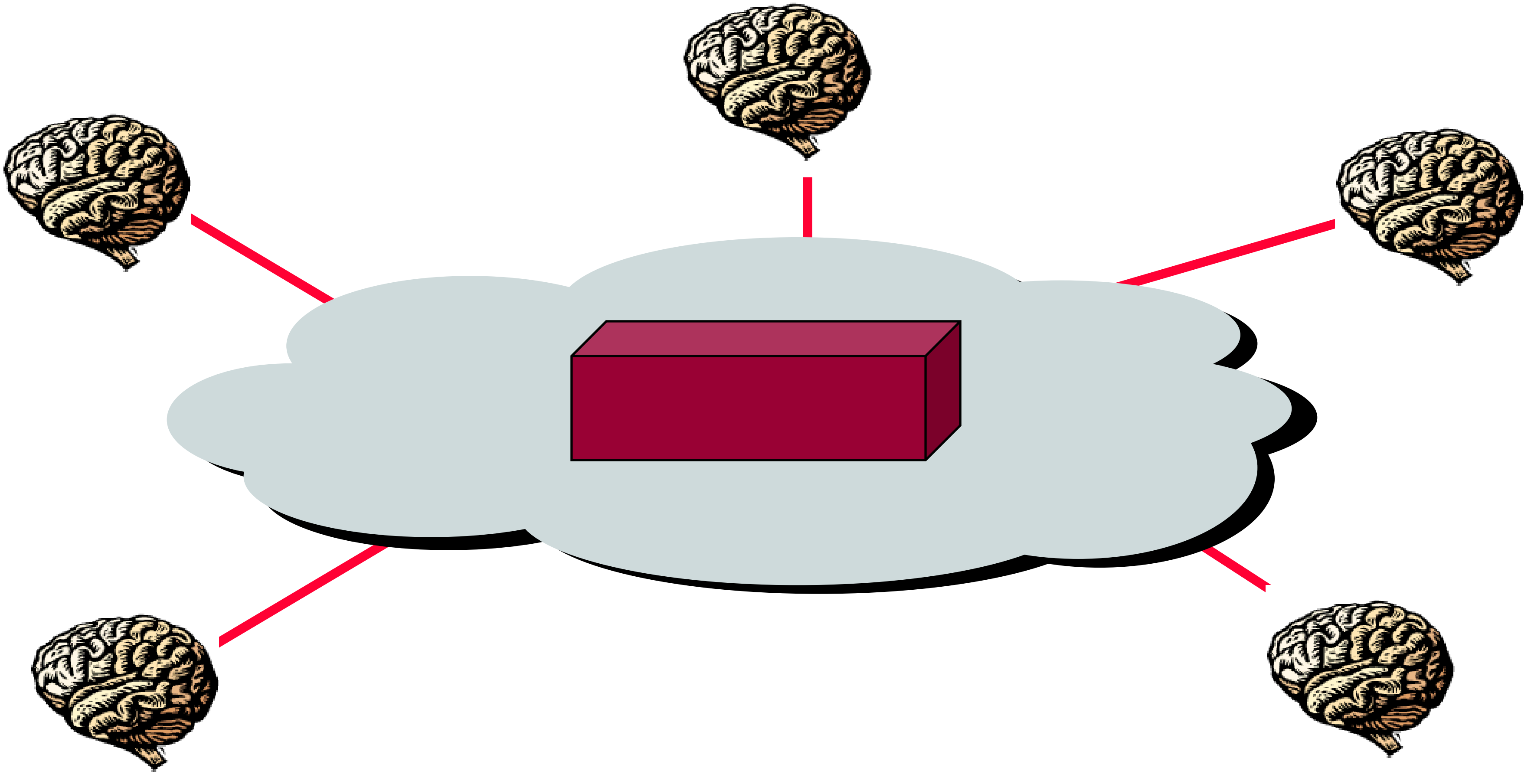
Saltzer, Reed, and Clark (Conf. 1981 – Jour. 1984)

- "...functions placed at the lower levels may be *redundant* or of *little value* when compared to the cost of providing them at the lower level..."
- "...sometimes an *incomplete* version of the function provided by the communication system (lower levels) may be useful as a *performance enhancement*..."
- This leads to a philosophy diametrically opposite to the telephone world of dumb end-systems (the telephone) and intelligent networks.

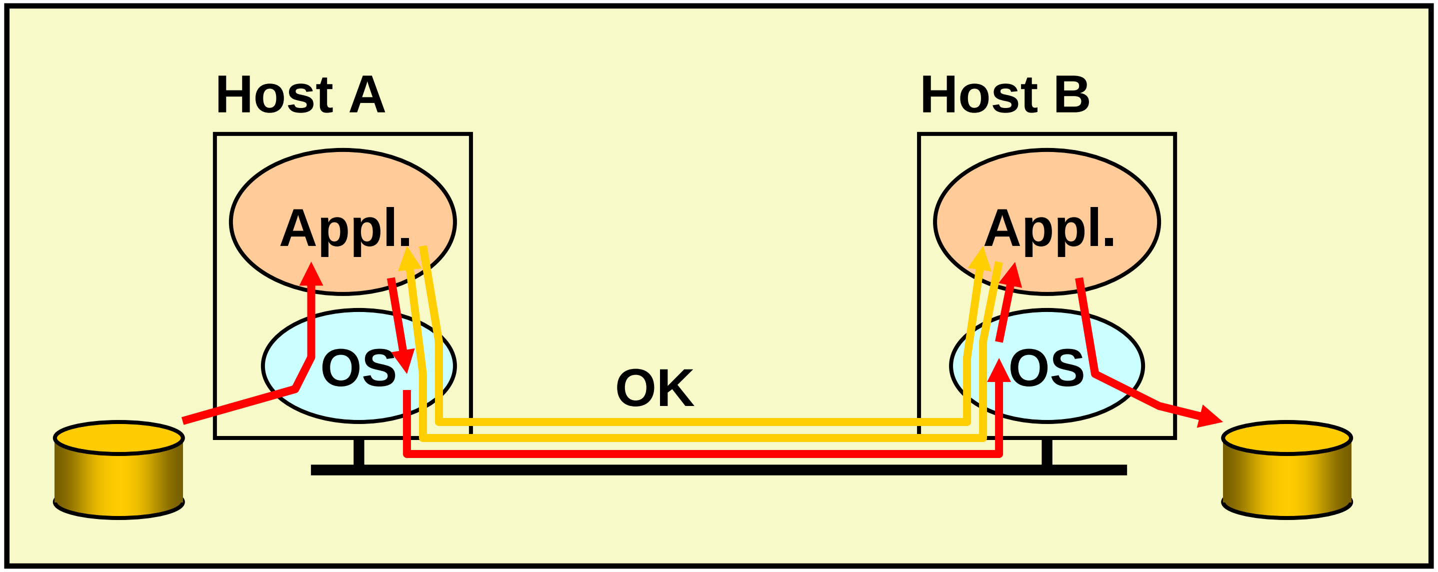
Common View of the Telco Network



Common View of the IP Network



Example: Reliable File Transfer



- Solution 1: make each step reliable, and then concatenate them
- Solution 2: each step unreliable: end-to-end check and retry

Trade-offs

- Application
  - has more information about the data and semantics of required service
- Lower layer
  - has more information about constraints in data transmission (e.g., packet size, error rate)

- These trade-offs are a direct result of layering!

Internet & E2E

- Network layer provides one simple service:
  - best effort datagram (packet) delivery
- Transport layer at network edge (TCP) provides end-end error/flow control
  - Performance enhancement used by many applications (which could provide their own error control)
- All other functionality ...
  - All application layer functionality
  - Many network services: DNS
  - implemented at application level

Internet & E2E: Discussion

Congestion and Flow control: why at transport, rather than link or application layers?

- Claim: common functions should migrate down the stack
  - Everyone shares same implementation: no need to redo it (reduces bugs, less work, etc...)
  - Knowing everyone is doing the same thing, can help
- Congestion control too important to leave up to application/user: true but hard to police
  - TCP is “outside” the network; compliance is “optional”
  - We do this for fairness (but realize that people could cheat)
- Why flow control in TCP, not (just) in app

E2E Argument: Summary

- End-to-end principle emphasizes:
  - *function placement*
  - *correctness, completeness*
  - *overall system costs*
- Philosophy: if application can do it, don't do it at a lower layer -- application best knows what it needs
  - add functionality in lower layers iff (1) used by and improves performances of many applications, (2) does not hurt other applications
- Allows *cost-performance* tradeoff

E2E Argument: Interpretations

- One interpretation:
  - A function can only be completely and correctly implemented with the knowledge and help of the applications *standing at the communication endpoints*
- Another: (more precise...)
  - a system (or subsystem level) should consider only functions that can be *completely and correctly* implemented within it.
- Alternative interpretation: (also correct ...)
  - Think twice before implementing a functionality that you believe that is useful to an application at a lower layer
  - If the application can implement a functionality correctly, implement it a lower layer *only* as a performance enhancement

E2E Argument: Discussion

- End-end argument emphasizes correctness & completeness, not
  - Complexity: is complexity at edges result in a “simpler” architecture?
  - Evolvability, ease of introduction of new functionality: ability to evolve because easier/cheaper to add new edge applications than change routers?
  - Technology penetration: simple network layer makes it “easier” for IP to spread everywhere

Internet Design Philosophy (Clark'88)

0. Connect existing networks
   - initially ARPANET and ARPA packet radio network
1. Survivability
   - ensure communication service even with network and router failures
2. Support multiple types of services
3. Must accommodate a variety of networks
4. Allow distributed management
5. Allow host attachment with a low level of effort
6. Be cost effective
7. Allow resource accountability

Different ordering of priorities could make a different architecture!

0. connect existing networks

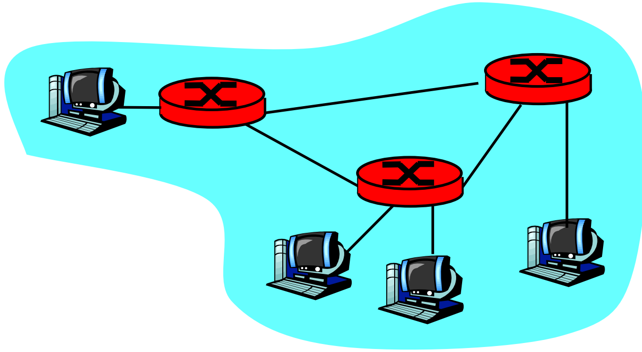
1974: multiple unconnected networks

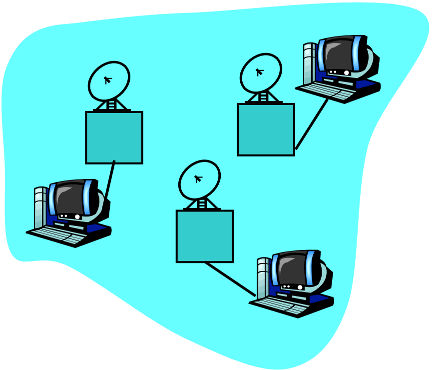
- ARPAnet
- data-over-cable networks
- packet satellite network (Aloha)
- packet radio network

.. differing in:

- addressing conventions
- packet formats
- error recovery
- routing

Cerf & Kahn: Open network architecture





- "…interconnection must preserve intact the internal operation of each network."
- " ..the interface between networks must play a central role in the development of any network interconnection strategy. We give a special name to this interface that performs these functions and call it a GATEWAY."
- ".. prefer that the interface be as simple and reliable as possible, and deal primarily with passing data between networks that use different packet-switching strategies
- "…address formats is a problem between networks because the local network addresses of TCP's may vary substantially in format and size. A uniform internetwork TCP address space, understood by each GATEWAY and TCP, is essential to routing and delivery of internetwork packets."

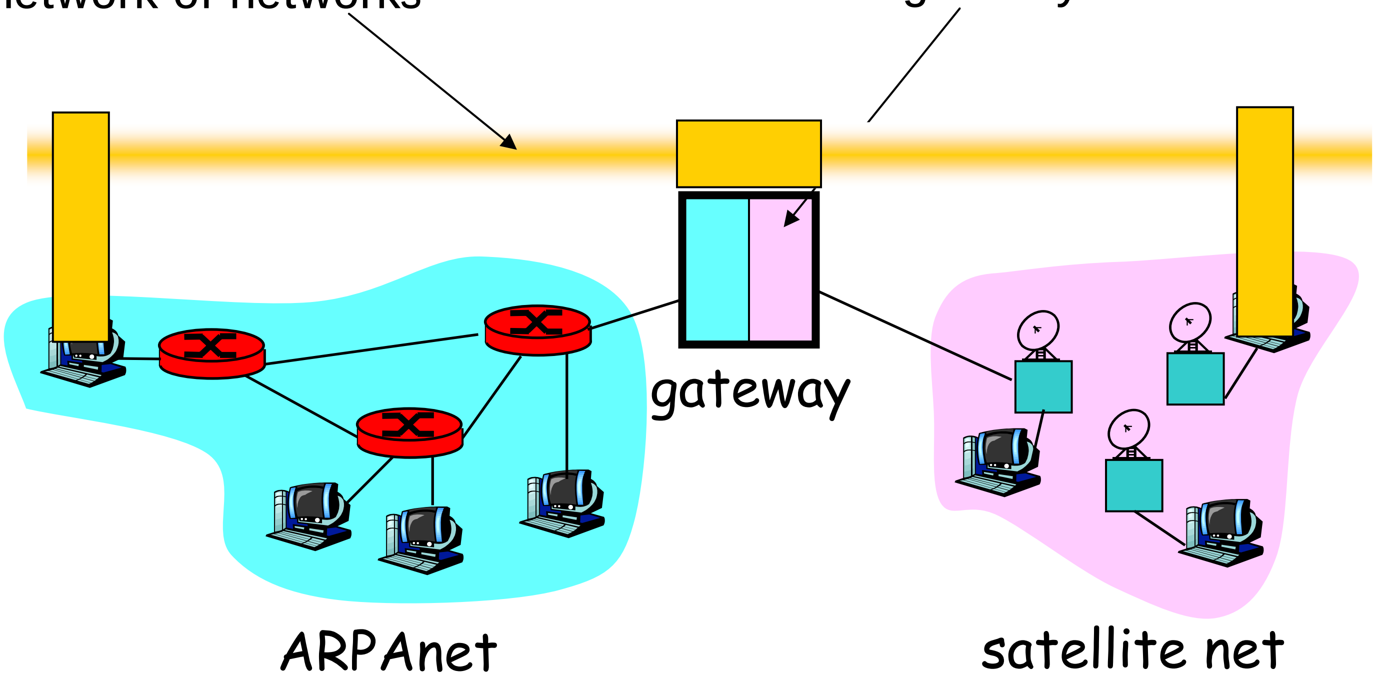
Cerf & Kahn: Open network architecture

Internetwork layer:

- addressing: internetwork appears as a single, uniform entity, despite underlying local network heterogeneity
- network of networks

Gateway:

- "embed internetwork packets in local packet format or extract them"
- route (at internetwork level) to next gateway



1. Survivability

- Continue to operate even in the presence of network failures (e.g., link and router failures)
  - as long as network is not partitioned, two endpoints should be able to communicate
  - any other failure (excepting network partition) should be transparent to endpoints
- Decision: maintain e2e transport state only at end-points
  - eliminate the problem of handling state inconsistency and performing state restoration when router fails
- Internet: stateless network architecture
  - No notion of a session/call at network layer
  - Grade: **A-**, because convergence times are relatively slow
  - BGP can take minutes to coverge
  - IS-IS OSPF take ~ 10 seconds

2. Types of Services

- Add UDP to TCP to better support other apps
  - e.g., “real-time” applications
- Arguably main reason for separating TCP, IP
- Datagram abstraction: lower common denominator on which other services can be built
  - Service differentiation was considered (remember ToS?), but this has never happened on the large scale (Why?)
- **A-**: proven to allows lots of applications to be invented and flourish (except MM, but maybe that’s not a transport service issue)

3. Variety of Networks

- Very successful (why?)
- The mantra: IP over everything *[read that again!]*
  - Then: ARPANET, X.25, DARPA satellite network..
  - Now: ATM, SONET, WDM...

Grade **A**: can’t name a link layer technology that IP doesn’t run over (carrier pigeon RFC)

Other Goals

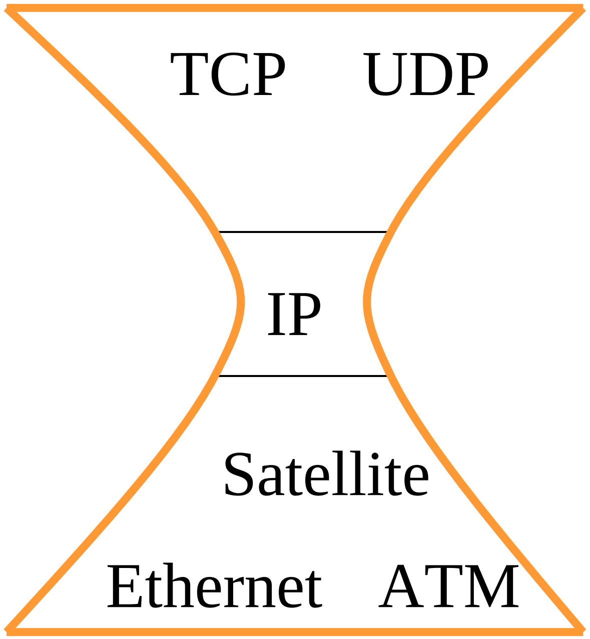
- Allow distributed management
  - Administrative autonomy: IP interconnects networks
    - each network can be managed by a different organization
    - different organizations need to interact only at the boundaries
    - ... but this model complicates routing
  - Grade **A-** for implementation, **B** for concept
- Cost effective
  - sources of inefficiency
    - header overhead
    - retransmissions
    - routing
  - ...but "optimal" performance never been top priority

Other Goals (Cont)

- Low cost of attaching a new host
  - Not a strong point → higher than other architecture because the intelligence is in hosts
  - Bad implementations or malicious users can produce considerably harm (remember fate-sharing?)
  - Grade **C**: but things are improving with DHCP, auto-cofigurations. Looks like a higher grade in future
- Accountability
  - Grade **F**

Summary: Internet Architecture

- Packet-switched datagram network
- IP is the glue (network layer overlay)
- IP hourglass architecture
  - all hosts and routers run IP
- Stateless architecture
  - no per flow state inside network



IP hourglass

Summary: Minimalist Approach

- Dumb network
  - IP provide minimal functionalities to support connectivity
  - Addressing, forwarding, routing
- Smart end system
  - Transport layer or application performs more sophisticated functionalities
  - Flow control, error control, congestion control
- Advantages
  - Accommodate heterogeneous technologies (Ethernet, modem, satellite, wireless)
  - Support diverse applications (telnet, ftp, Web, X windows)
  - Decentralized network administration