Design Principles of Internet

David D. Clark's paper "The Design Philosophy of the DARPA Internet Protocols" 1988

Why Internet is the way it is?

David Clark

Position: Senior Research Scientist

Office: 32-G816

Phone:253-6003

E-mail: ddc@csail.mit.edu

Research Directorate(s): AI

URL:

Biography:

Since the mid 70s, Dr. Clark has been leading the development of the Internet; from 1981-1989 he acted as Chief Protocol Architect in this development, and chaired the Internet ActivitiesBoard. Recent activities include extensions to the Internet to support real-time traffic, explicit allocation of service, pricing and related economic issues, and policy issues surrounding local loop employment. New activities focus on the architecture of the Internet in the post-PC era. He is chairman of the Computer Science and Telecommunications Board of the National Research Council.

Early Goals

Make disjoint networks talk to each other effectively

Choices

A. Build a tightly integrated, unified network

B. Interconnect existing network

Why ?

more practical. Networks represents separately administered entities.

Choices

A. Packet Switching
B. Circuit Switching

Why ?

The networks to be integrated are packet switched network. Packet switch is natural choice for the applications at the time (remote login).

More Goals

Robust - work despite failure of networks or gateways

Versatile - support a variety of services and networks

Permit distributed management of resources

Cost effective

Easy to add new hosts

Permit accounting of resources

Top Goal

"Survivability in the Face of Failure"

Communication between two entities should continue after temporary disruption without needing to reestablish connection states.

Or

Mask transient failure

Store connection states in

A. packet switching nodes
B. end nodes

Why ?

Easier to implement than replication. Replication only protects against finite number of node failures.

Fate-Sharing

The only way the states are lost is the failure of end hosts.

Consequences

Stateless packet switchers.
Need to trust end hosts.

Goal No. 2

“Support a variety of services”

Services

Remote login - low delay, reliable

File transfer - delay not important, reliable

Teleconferencing - reliability not important, low delay

Choice

A. Break into multiple protocols.

Protocols

IP - datagram-based, best effort
TCP - reliable service over IP
UDP - unreliable service over IP

Compared to

X.25 - provides reliable services
(that cannot be switched off!)

Goal No. 3

“Support a variety of networks”

Make minimal assumptions

Can transport packets
Best effort delivery
Addressing
Minimum packet size

Not assuming

Reliability
Ordered delivery
Packet prioritization
Broadcast/multicast
Knowledge of network stats

Goals

Robust - work despite failure of networks or gateways
Versatile - support a variety of services and networks
Permit distributed management of resources
Cost effective
Easy to add new hosts
Permit accounting of resources

Another David D. Clark's paper
"End-to-End Arguments in System Design" 1984
(with Saltzer and Reed)

E2E Argument

A tool to guide designers:
which layer to implement a
given functionality?

Example

Reliable file transfer between host A and host B

Steps

1. *A* reads file from disk
2. *A* transmits file as packets
3. Network delivers packets
4. *B* receives packets
5. *B* write data to disk

Possible Errors

1. Disk failure
2. Software bugs
3. Packet loss
4. Processor/Memory errors
5. OS crashes

Choices

A. Make sure every step is reliable

B. End-to-end check and retry (compare checksum, resend if error)

The Argument

To achieve careful file transfer, the transfer application must apply application-specific, end-to-end reliability guarantee.

The Argument

> "The end-to-end check of the file transfer application must still be implemented no matter how reliable the communication system becomes."

Conclusion

No need to provide reliability guarantee at lower level (e.g. network, OS, hardware)

Actually,

Lower level reliability can improve performance.

To implement at low-level?

Additional cost for applications that do not require the feature.

Less information than the “end”, less efficient.

Other Example: Data Encryption

Choices

A. Encrypt at the network-level

B. Encrypt in the application

Why?

Intercept before reaching the network
Need to trust the network
Still need to authenticate

Other Example: RISC

The Argument

Any attempt by the computer designer to anticipate the client's requirements will probably miss the target and the client will end up re-implementing it anyway.

The End Point?

Applications?

Users?

Hosts?

The end-point is a trustworthy entity.

Example

Reliable file transfer between host A and host B

If I don’t trust the file transfer application, I need to check for error myself.

E2E Argument

> The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the endpoints of the communication system. Therefore, providing the questioned function as a feature of the communication system itself is not possible. (Sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement)