

TELE 4363 Review

Part 1: Principles of Protocol Design

Copyright © 2003, Tim Moors

Exam

Cover page is on the web page

Content:

- Material covered in lectures (weighting \propto # lectures)
 - Some lectures (wireless TCP & scheduling) used slides that the lecturer didn't produce. Some of these slides appear on the web page, but weren't covered in lectures, and aren't examinable.
 - Review lectures summarise the main topics, but not necessarily all topics covered in the exam.
- Material covered in assignments

Style: Similar to tutorial questions:

- Descriptive
- Test understanding of concepts

Copyright © 2003, Tim Moors

Outline



- Desirable protocol attributes
 - Connectivity
 - Extensibility
 - Modularity
- End-to-end arguments
- State information
- Scalability
- Implementation

Copyright © 2003, Tim Moors

Desirable protocol attributes

Correctness: It provides the required service (!)

Robustness: The service doesn't degrade excessively when components fail/experience unusual stimuli

Security

Determinism/predictability: ease of understanding

Ease of use

Extensibility: It can be changed in the future to rectify bugs, or add features.

Performance: high throughput, low response time; "tweakability"

Efficiency: High performance at low cost (few resources)

Scalability: Its performance is insensitive to the number of interacting parties.

Copyright © 2003, Tim Moors

Network externalities

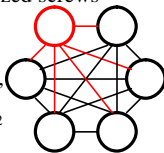
"a change in the benefit ... that an agent derives from a good when the number of other agents consuming the same kind of good changes." [Liebowitz & Margolis 98]

e.g. economies of scale from standard-sized screws

Metcalf's Law:

"The value of a network grows as the square of the number of its users."

$$\text{"value"} = n(n-1) \propto n^2$$



Positive examples: telephones & email; document exchange formats (Microsoft Word, PDF).

Negative example: videophone

Copyright © 2003, Tim Moors

Connectivity through minimalism

If variations of layer L offer different services, then layer $L+1$ maximises connectivity by only assuming the lowest-common denominator service

e.g. different link layers may offer different services (priorities, guaranteed timeliness, max frame length, etc).

- Network layer in end-system maximises generality by **Hetero links** assuming minimal service.
- Intermediate systems may not be able to map exactly between services \Rightarrow internetwork offers lowest common denominator.

Internet protocol maximises connectivity by minimal assumptions about links: Assume datagram (discrete packets that may be delivered out of order), moderate size (576B minimum [RFC1122]), no service guarantees.

c.f. virtual circuits (datagrams have other advantages, e.g. robustness)

Copyright © 2003, Tim Moors

Extensibility

Motivations:

- Interoperability with legacy systems increases connectivity
- Experience in using a protocol may lead to improvements; prefer to extend an existing protocol than develop again from scratch.

TCP header

Extension of protocol structures:

PDU's often contain more codepoints than are needed

- binary \Rightarrow power-of-2 codepoints available, whereas requirement may be smaller
- aligning fields on word boundaries for performance leaves free bits

Deliberate hooks to facilitate extensibility: version numbers in headers, leave some codepoints reserved for future use.

Copyright © 2003, Tim Moors

Modularity

“All good computer scientists worship the god of modularity, since modularity brings many benefits ...”
– D. Clark in the foreword to [Peterson & Davie]

Reduction of complex problem into tractable parts
Reuse of modules, both for different roles in the system, and in different systems

Replacement of modules to allow the system to be upgraded, configured for a particular application, or repaired after component failure

Removal of modules, allowing the system to be stripped down to reduced functionality, to which further functionality can be added to create different systems.

Copyright © 2003, Tim Moors

Assumptions about modularity

Overhead is insignificant, i.e. communication between modules within a system is trivial compared to communication between different systems.

Often true: Low error rates & high bandwidth, homogeneity
Counter example: Outboard processors [RFC 817]

Implementations can be treated as black boxes: only external interfaces matter. **Modularity of implementation needn't correspond to modularity of specification.** Examples:

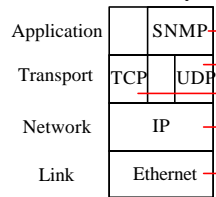
- Each layer holding data in its own buffer space requires copying between buffer spaces, which is inefficient. Implementations tend to use buffer cut-through.
- Network interface card supposedly implements link layer (with driver software isolating it from higher layers), but may also implement aspects of higher layers (e.g. transport layer checksum computation).

Copyright © 2003, Tim Moors

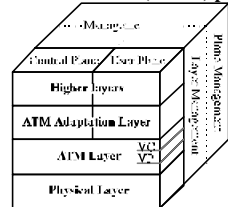
Parallel layers – planes

Some functions (e.g. management) pervade all layers
May separate processing of control and payload

SNMP interacts with layer agents



Broadband-ISDN separates control and data (“user”) planes



Copyright © 2003, Tim Moors

Proliferation of layers

Causes:

- Convergence
- Sophisticated stacks as dumb pipes
- Tunnelling

Performance concerns may act as a counterbalance:

- Efficiency is less important than ease-of-development today, as technology rapidly advances (processing: Moore's Law, fibre/wireless bandwidth).
- Cutting out the middleman with IP-over-optical:
 - early 1990s: IP-over-ATM
 - late 1990s: IP over SONET
 - early 2000: IP over WDM

Copyright © 2003, Tim Moors

Convergence layers

Enhance the service of a layer beyond what it intrinsically provides by covering it with another layer

- Add new services, e.g. TLS above TCP
- Emulate old services with new technology, e.g. LAN Emulation over ATM

Protocols may be deliberately designed with sublayers to facilitate composition, e.g. NCP \rightarrow IP+TCP/UDP, ATM:

- ATM Adaptation Layers: SAR + convergence;
- Link: Physical Medium Dependent + Transmission Convergence

Copyright © 2003, Tim Moors

13

Layers upon layers

Any bit pipe (irrespective of sophistication) can be used as a link

⇒ Low layers can be stacked above “upper layers”, e.g.:

- dial-up ISP: phone service might implement voice over IP; modem uses this as an analog link
- ATM never made it to the desktop, but was used for router links

Copyright © 2003, Tim Moors

14

Tunnelling

Allow communication through networks that don't directly support the protocol.

Virtuous applications: IP in IP encapsulation.

- New service types: IPv6 [www.6bone.net], multicast (Mbone)
- Virtual Private Networks (VPNs): Use Internet as packet carrier between gateways to protected domains.

Questionable applications: Bypass firewalls

Figure from <http://www.cisco.com/univercd/jlls/5/85/52685.gif>
Copyright © 2003, Tim Moors

15

Outline

Copyright © 2003, Tim Moors

16

The (principal) argument

Certain functions can only be implemented “completely and correctly” at the transfer end-point.

Implementations elsewhere **may** be justified as performance enhancements

Questions:

- Which functions?
- What is complete/correct implementation?
- What are “end-points”?

Copyright © 2003, Tim Moors

17

Error control performance: e2e vs local

Copyright © 2003, Tim Moors

18

Outline

Copyright © 2003, Tim Moors

Definition

Definition: “memory in the system used to influence future behaviour” [Keshav, p. 108]

Some examples:

Network layer: State may indicate route \Rightarrow preserve sequence resource reservations

Transport layer: State indicates what’s been transferred \Rightarrow reliable transfer (e.g. completeness)

Application layer: How to present information (e.g. Big vs Little Endian number representation; CR or CRLF line breaks)
Content of a “shopping basket” for e-commerce

Classifications:

- Critical vs performance-enhancing
- How long does it last?/How is it released?

Copyright © 2003, Tim Moors

Critical state info

Critical (necessary for correct implementation of functionality), e.g.:

End-systems:

- Error control: What has been successfully received?
- Flow control: How much more info can be sent without ack?

Network:

- Providing service guarantees: What resources have been assigned to admitted flows? What service level/load did we agree to for this flow? How much traffic has this flow contributed?
- Billing information

Copyright © 2003, Tim Moors

Performance-enhancing state info

Performance-enhancing, e.g.:

- Web cache: If find page in client’s web cache, then don’t have to retrieve it from server \Rightarrow reduce delay and network load.
 - If page isn’t in cache, can always get it from the server.
 - Losing information in cache affects performance, but not overall functionality.
- Complicated network functions (e.g. QOS routing) performed once during connection setup for later quick reference during transfer.

Copyright © 2003, Tim Moors

Example: State management with TCP

- Introduction to TCP
- TCP state information
- Normal TCP connection lifecycle
- Possible abnormalities
 - Timeout on connection establishment

[Stevens 1, Ch. 23, 22, 18.3]

Copyright © 2003, Tim Moors

Cookies[†]



Another way to manage state information

Process:

1. Client requests information from server
2. Server responds, including state information
3. Client stores state information, associates it with server
4. Client includes state information with subsequent requests to server

Bottom line: Server doesn’t have to store state information.

[†] aka “handle”, “transaction ID”, or “token”

Copyright © 2003, Tim Moors

Outline



Copyright © 2003, Tim Moors

Scalability

Defined: Protocols should perform well & be economical irrespective of how many nodes (N) use them.

- Scaling up: Important if a protocol becomes popular
- Efficiency: Protocol overhead should be proportional to $\log(N)$, rather than to N .
 - Load balancing: Distribute resource demands to avoid performance bottlenecks.
 - Compartmentalisation: Confine errors to limited parts of the whole system.

[Perlman, § 18.3]

- Scaling down: Important for consumer use
- Protocol should be economical to operate with only a few nodes, e.g. Ethernet.

Centralised techniques

Advantages:

- Ease of management, control (e.g. firewalls: concentrate all traffic entering an organisation to one focal point for enforcing policy)
- No communication overhead; consistent world view (c.f. distributed systems where it takes time to develop consensus)
- Simplicity

Disadvantages:

- Component failure can cause system failure
- Load on component increases with system size \Rightarrow scalability issues
Central component may require performance that is beyond what is achievable; distributing the system can replace this with many components with lower performance

Use hierarchies to address disadvantages of centralised techniques.

Examples of synchronisation

Some phenomena are periodic, and participants happen to maintain phase, e.g. file backups & software updates at midnight.

Other phenomena trigger multiple follow-on phenomena, causing waves of consequences.
e.g. a link failure may cause adjacent routers to advertise new routes to neighbours, and advertisements propagate throughout the network.

More examples of synchronisation:

- Ethernet: synchronisation of stations attempting retransmission after collision
- Computers throughout a building rebooting simultaneously after power recovery (protocols, e.g. DHCP, often wait a random period of time before starting to avoid such synchronisation)
- Bluetooth devices responding to an inquiry (e.g. what printers are in the area?) attempt to respond simultaneously & send power burst in ISM band.

Synchronisation causes unnecessary temporary overload & has potential to recur indefinitely. Avoid by adding random delays.

Outline

Hardware vs software partitioning

- Hardware: optics & electronics
 - Software: Programs to control hardware
- Grey areas in between, e.g. is a NIC that includes a processor "hardware" or "hardware+software"?

Occasionally, specialised processors are used for intensive higher-layer processing, e.g. compression, encryption TCP/IP, or parts thereof, are *sometimes* implemented in NIC hardware to reduce CPU processing burden for high-speed links. e.g. Gigabit Ethernet NICs may calculate checksum so that CPU doesn't need to, TCP/IP Offload Engines (e.g. Adaptec ANA-7711) implement all (or almost all) of TCP/IP

Network layer and above *usually* implemented as software on Central Processing Unit. Network processors in routers.

Link layer *often* implemented in hardware (Network Interface Card) + software (driver to interface to NIC to computer)

Physical layer *must* have some hardware components. May also have software (e.g. channel quality monitoring & Software Defined Radios)

Application
Presentation
Session
Transport
Network
Data link
Physical

Buffer management

- In theory, layers are independent: When a layer processes a packet, it stores that packet in its own buffer.
- In practice, multiple layers are implemented on one processor & copying information between separate buffers is wasteful.
- Most implementations use "buffer cut-through": packet remains static in memory, and layers exchange the packet by passing pointers towards it.

Principles of formatting data units

- Formatting reflects processing order
- Formatting to conserve processing
- Optimise for the common case
- Simplify receiver processing