Specification of a Network Adaptation Layer for the Grid

GGF7 presentation

Michael Welzl
michael.welzl@uibk.ac.at
University of Innsbruck

Outline

- Problem statement
- Proposed solution
  - what
  - why
  - how
- Just 10 mins... no time to conclude ;-)

Note: we simplify!

- Assumption 1:
  all is done @ end nodes
  - No dedicated network
  - No management / control of network resources

- Assumption 2:
  realistic usage of *existing* Internet technology
  - No QoS mechanisms like DiffServ or IntServ
  - No QoS routing, no Active Networks

- Assumption 3:
  a single end2end connection!
  - No overlay structure
  - No distributed schemes ... no P2P, caching, etc.

a tough one, right? ;-)

Current use of the Internet

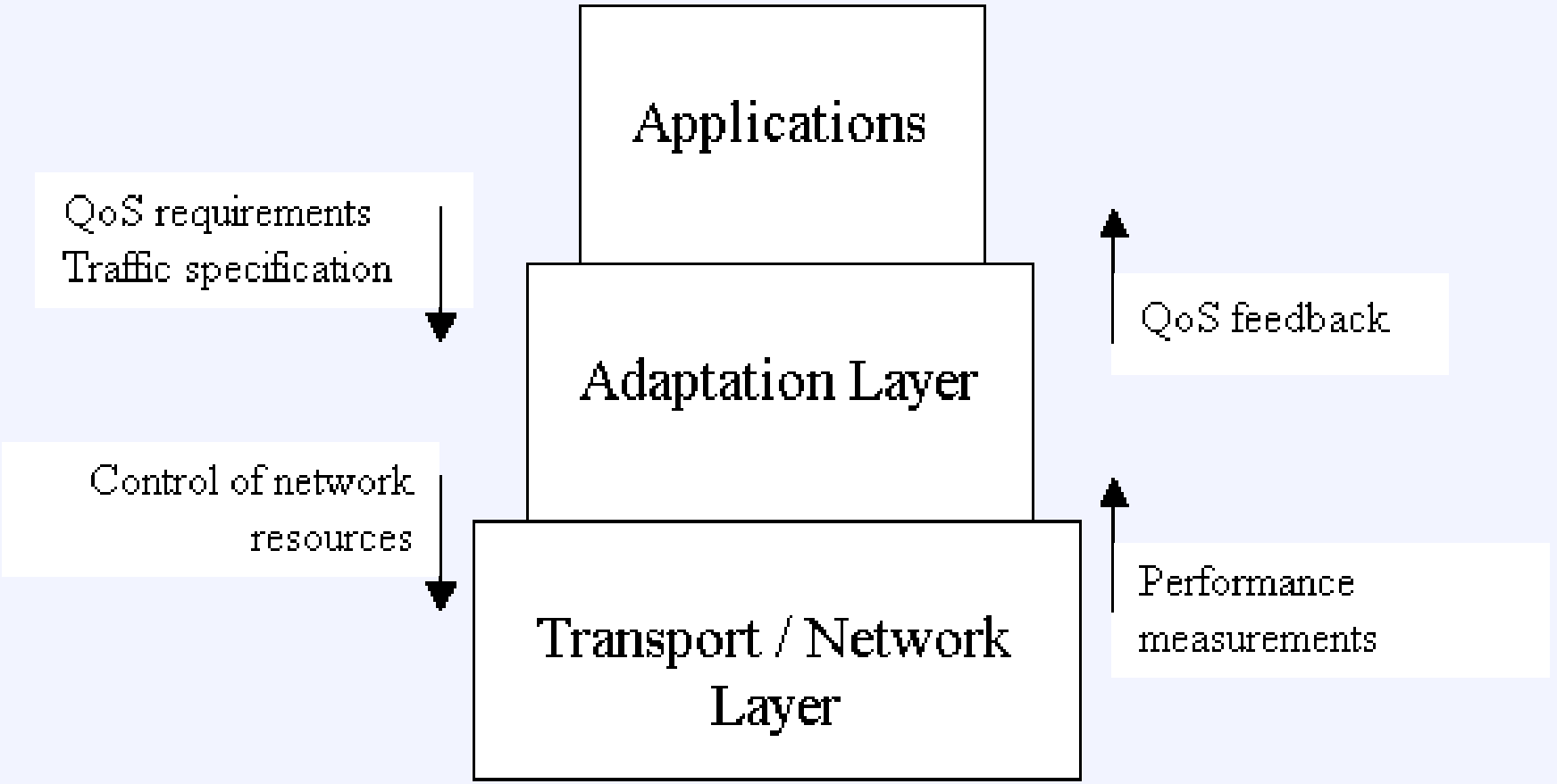
- TCP
  - byte stream from source to destination
  - reliable, connection oriented service
  - all kinds of complex features
    - window based flow and congestion control
      - RTT estimation, self-clocking, parameters: max. / init. window size,...
      - slow start / congestion avoidance
    - flavors: Tahoe, Reno, NewReno, SACK, with and w/o ECN, ..
- UDP
  - connectionless service
  - ports and a checksum ... that's it :)
    - simpler, but useless for reliable transport (DIY)
    - What about congestion control?

Some (realistic) things we can do...

- Alter packet size
- Tune TCP parameters
  - decide which TCP flavor to use, fiddle with window size, ..
- Implement rate control on top of UDP
- Use new technologies, like...
  - UDP Lite: transmission of erroneous payload
  - SCTP: transport level multihoming, reliable out-of-order transmission
  - DCCP: congestion control for datagrams (connectionless)
- Measure and do something... maybe adapt payload...

Proposed solution

What: an "Adaptation Layer"



Why we need it

- Application relieved of burden
  - more sophisticated transmission mechanisms possible
  - tailored network usage instead of "one size fits all" (just UDP / TCP)
- Network provides service - app specifies QoS requirements
  - Adaptation layer makes the most out of available resources
- Adaptation layer provides QoS feedback
  - Information logically closer to application
- Full transparency to application
  - gradual deployment of new transport mechanisms

How it could work: application interface

- from application
  - QoS spec
    - apply weights to QoS parameters
    - goal: tune trade-offs (packet sizes, ..)
    - Examples:
      - reduced delay is more important than high throughput
      - I don't care about a smooth rate (I use large buffers)
  - Traffic spec
    - Example: long lasting stream, "greedy"
- to application
  - "video frame complete" instead of "throughput = ... loss = ... ", ..

How it could work: internals

- Control of network resources
  - Tune packet size
    - maximize throughput + minimize delay according to QoS spec
  - Choose congestion control + tune parameters
    - based on QoS-centric evaluation of mechanisms: RAP, TFRC, TEAR, LDA+, GAIMD, Binomial CC., ..
    - Negotiation: DCCP
  - Further functions: buffer, bundle streams, ..
    - Example: long-term stream, sporadic interruptions + delay not important ⇒ buffer, don't restart CC
- Performance measurements
  - use existing tools (NM-WG) + passively monitor TCP

Bringing it to life

- Now
  - architectural design: interfaces, QoS spec format, ..
  - could be done in IETF for other apps, but:
    - Grid apps have special QoS requirements / traffic properties
      => tailored architecture
- Future
  - a lot of work required (QoS-evaluation of CC., DCCP, ...)
  - extension to use "real" QoS mechanisms, distributed measurements, coordination protocol, ...
  - could grow along!
- RG? WG? Document(s?) in existing RG / WG?