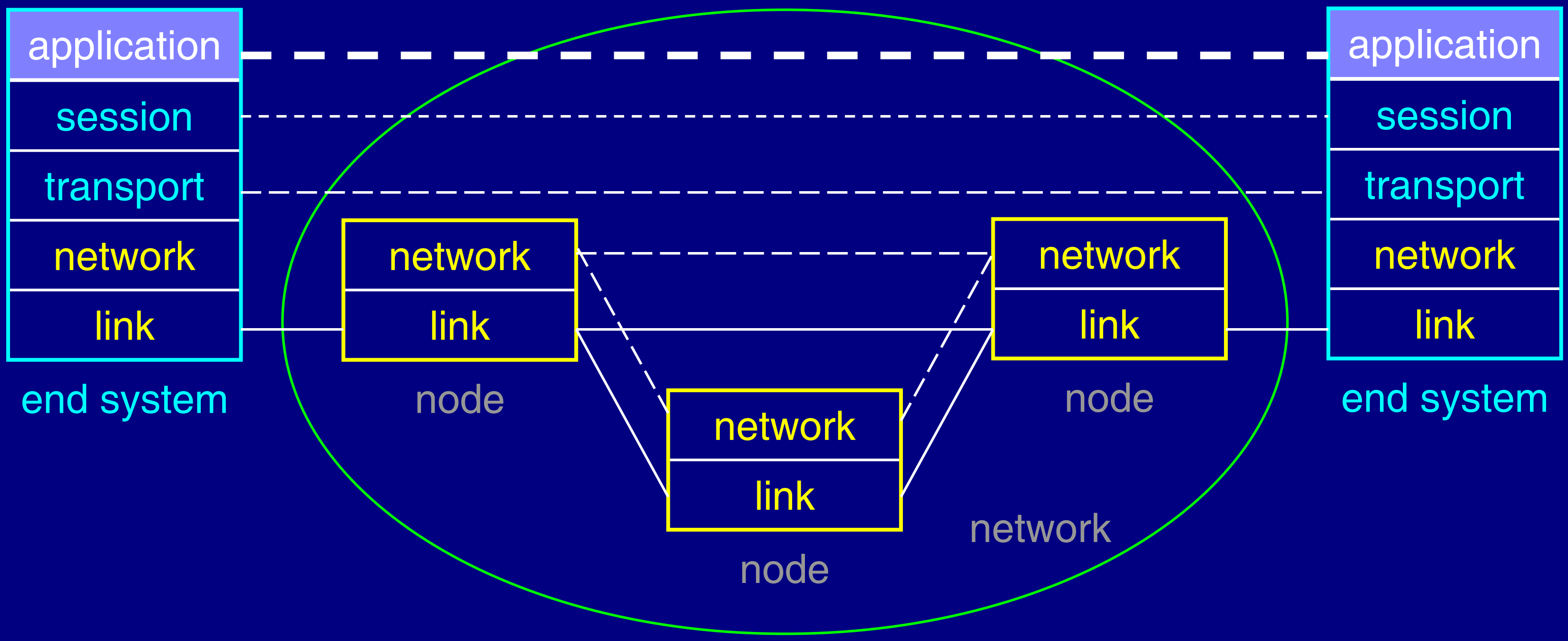
Networked Applications

1. Introduction
2. Fundamentals and design principles
3. Network architecture and topology
4. Network control and signalling
5. Network components
   5.1 links
   5.2 switches and routers
6. End systems
7. End-to-end protocols
8. Networked applications
9. Future directions

Networked Applications



8.1. Application characteristics
8.2. Application categories
8.3. Application adaptation
8.4. Network interaction

Application Primacy

Applications Justify Network Infrastructure

Application Primacy A-I

The sole and entire point of building a high-performance network infrastructure is to support the distributed applications that need it.

Field of Dreams vs. Killer App Dilemma A-I.1

Advances in network technology create the "field of dreams" to which new applications are drawn, but are difficult to motivate without the tangible pull of a new "killer application" – which can't emerge in the absence of an adequate network infrastructure. This chicken-and-egg dilemma governs the supply-and-demand dynamic of high-speed networking.

Networked Applications
Application Characteristics

8.1 Application characteristics
- 8.1.1 Bandwidth
- 8.1.2 Latency
- 8.1.3 Error tolerance
- 8.1.4 Application flow characteristics

8.2 Application categories

8.3 Application adaptation

8.4 Application–network interaction

Application Characteristics
User Expectations

- Users of applications care only about *delay*
  - end-to-end through the network
  - application processing

User Expectations of Interapplication Delay A-I.2

The performance metric of primary importance to users of communicating applications is the total delay in completing the communication, including both the end-to-end delay and any delay introduced by the application itself.

Application Characteristics
User Expectations

- Users of continuous media care about quality
  - video: frame rate and resolution
  - audio: frequency range and distortion
  - note that if *delay* is zero, all playout is local

User Expectations of Continuous Media Quality A-I.2*c*

The performance metric of primary importance to users of continuous media streams is the quality of the stream, such as frame rate and resolution for video; frequency range and distortion for audio.

Application Characteristics
Network Performance Requirements

- Users expectations drive network performance
  - latency: directly from delay requirements
  - bandwidth: part of object transmission delay

High-Speed Applications Need High Bandwidth and Low Latency A-I.3

Bandwidth and latency are the primary performance metrics governing interapplication delay. The latency requirements of an application depend on the expectations of the user. For data-intensive applications, delay sensitivity drives bandwidth requirements.

Application Characteristics
Bandwidth Requirements

- Individual bandwidth
  - single application instance that needs significant bandwidth
  - example: interactive Web browsing needs ~ 1 Gb/s per user
- Aggregate bandwidth
  - all instances of application use significant fraction bandwidth
    - 1/100th to 1/1000th
  - example: interactive Web browsing
  - example: HDTV streams to US households need ≈1.5 Pb/s

Application Characteristics
Bandwidth Requirements

- "High speed" is relative
  - more difficult to achieve up the protocol stack
  - more difficult to achieve toward the edge of the network
  - increases with time as technology improves

Relative Magnitude of High-Speed A-2A

High-speed is a relative term. Application requirements for high speed – alleged or actual – have historically grown to at least the limits of currently attainable link speeds and network capacity.

Application Characteristics
Bandwidth Scaling

- Bandwidth enhanced
  - usable at low bandwidth, utility increases with bandwidth
  - example: streaming media
- Bandwidth challenged
  - requires significant change to operate at high bandwidth
  - example: distributed computing (IPC and state exchange)
- Bandwidth enabled
  - require high bandwidth to operate
  - example: uncompressed video studio production
  - example: distributed scientific visualisation and CAD/CAE

Application Characteristics
Application Partitioning

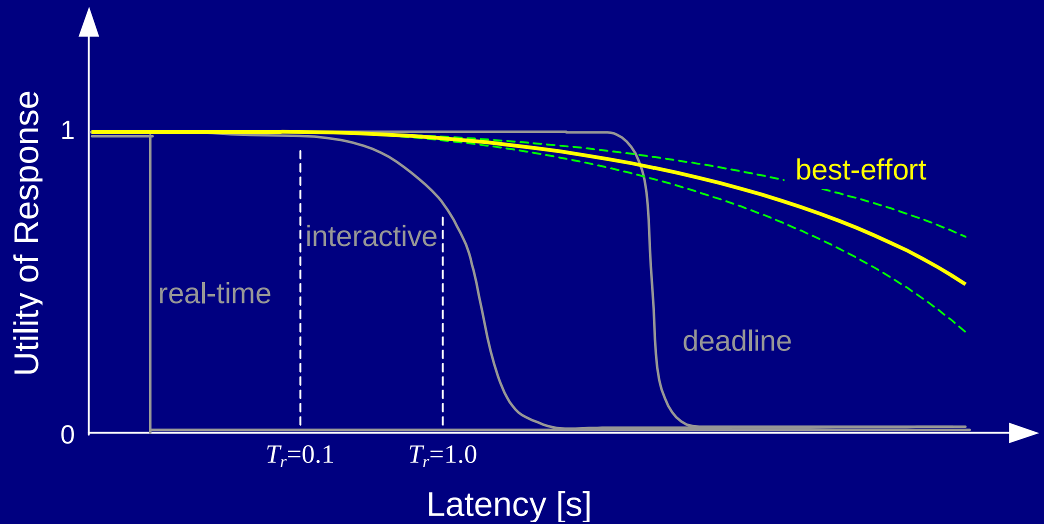
- Application partitioning
  - critical effect on bandwidth requirements
  - e.g. image processing
    - communicate delta and structures (polygons) rather than full bitmaps

Properly Partition Distributed Applications A-IV_3

Carefully determine where to cut applications to distribute across the network. Some "high-speed" applications are simply poorly partitioned low-speed applications.

Application Characteristics
Latency: Best Effort

- Loose *d* bound
- Relative service
  - policy or fair
- Examples
  - email
  - netnews

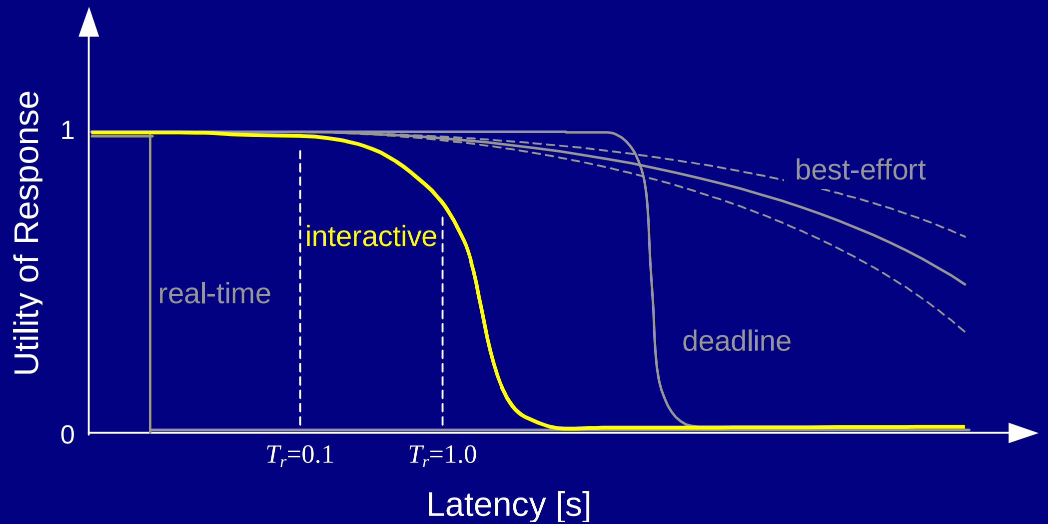


Fair Service Among Best-Effort Applications A-II.2*b*

In the absence of specific policy directives to favor some applications over others, network and end-system resources should be allocated fairly (roughly equally) among competing best-effort applications.

Application Characteristics
Latency: Interactive

- *d* bound
  - subsecond
  - ideally 100ms
- Examples
  - info access
  - Web browsing



Interactive Response Time A-I.2*i*

Interactive applications should provide a reasonable time ranging from an ideal of 100 ms to a maximum target of 1 second. Within this range, consistent response is better than high variance.

Application Characteristics
Latency: Interactive Adaptation

- Structured data
  - presentation of partial data as it arrives (e.g. text)
  - presentation of smaller chunks in interactive response time
    - e.g. hierarchical structuring

Structure Data to Meet Interactive Response A-7A

Bounds *Structure and migrate data for interactive responses such that initially visible data is present at sufficient resolution within interactive response bounds.*

Application Characteristics

Latency: Interactive Adaptation

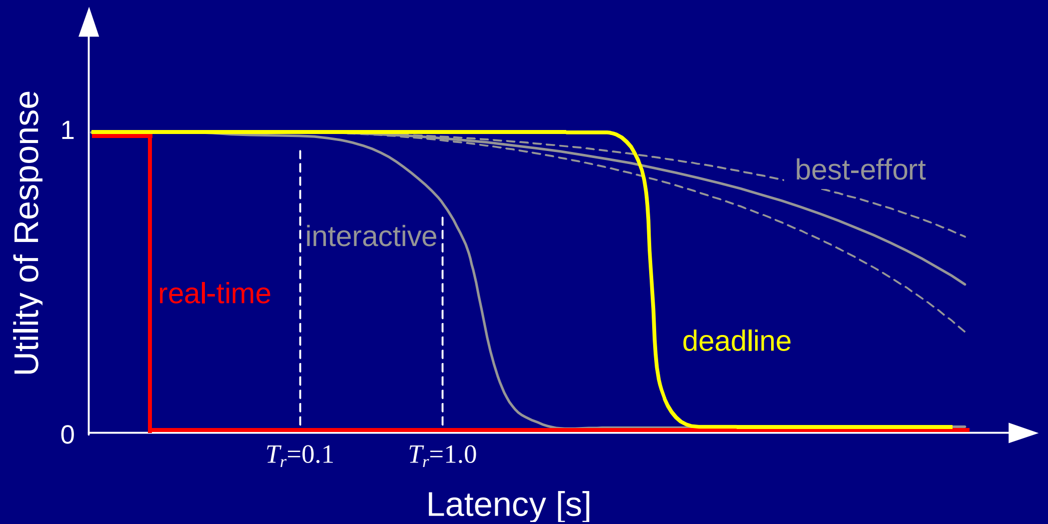
- Successive refinement
  - presentation of progressively more information over time
  - each user can decide when enough is presented
  - supported by
    - structured data
    - layered and progressive coding

Application Adaptation by Refinement A-5D

Quick and partial information is better than slow and complete information, particularly with refinement over time. Successive refinement can be used to adapt a high-speed application to a low-speed network.

Application Characteristics
Latency: Real Time and Deadline

- Hard real time
  - *d* bound
    - strict
    - small
- Deadline
  - *d* bound
    - relatively strict
    - large; allows adaptation well in advance of the deadline



- Examples

real time: process control, sensor with no memory, lifeline

deadline: remote backup

Application Flow Characteristics

	Characteristic					
Application flow	Individual bandwidth	Start/transient delay	Steady-state delay	Latency budget	Loss tolerance	Adaptability
Distributed computing	low–high	–	real-time	1μs–10ms	none	low
Process control	low	–	real-time	1μs–10ms	none	low
Haptics	very low	–	real-time	10 ms	low	low
Live interactive voice	low	interactive	real-time	30 ms	very low	limited
Live interactive video	med	interactive	real-time	300 ms	low	moderate
Stored streaming video	mod	interactive	–	1–10 s	low	high
Stored interactive video	mod	interactive	interactive	100 ms	low	moderate
Web browsing	med–high	interactive	–	100 ms – 1 s	none	moderate
Information push	low–med	push	–	1 min – 1 d	moderate	high
Telemetry	low–med	–	varies	varies	none	limited
Remote Backup	high	push	deadline	1 hour	none	high
email	low	push	best effort	1 min – 1 hr	very low	high

Networked Applications
Application Categories

Application Categories

Characteristics

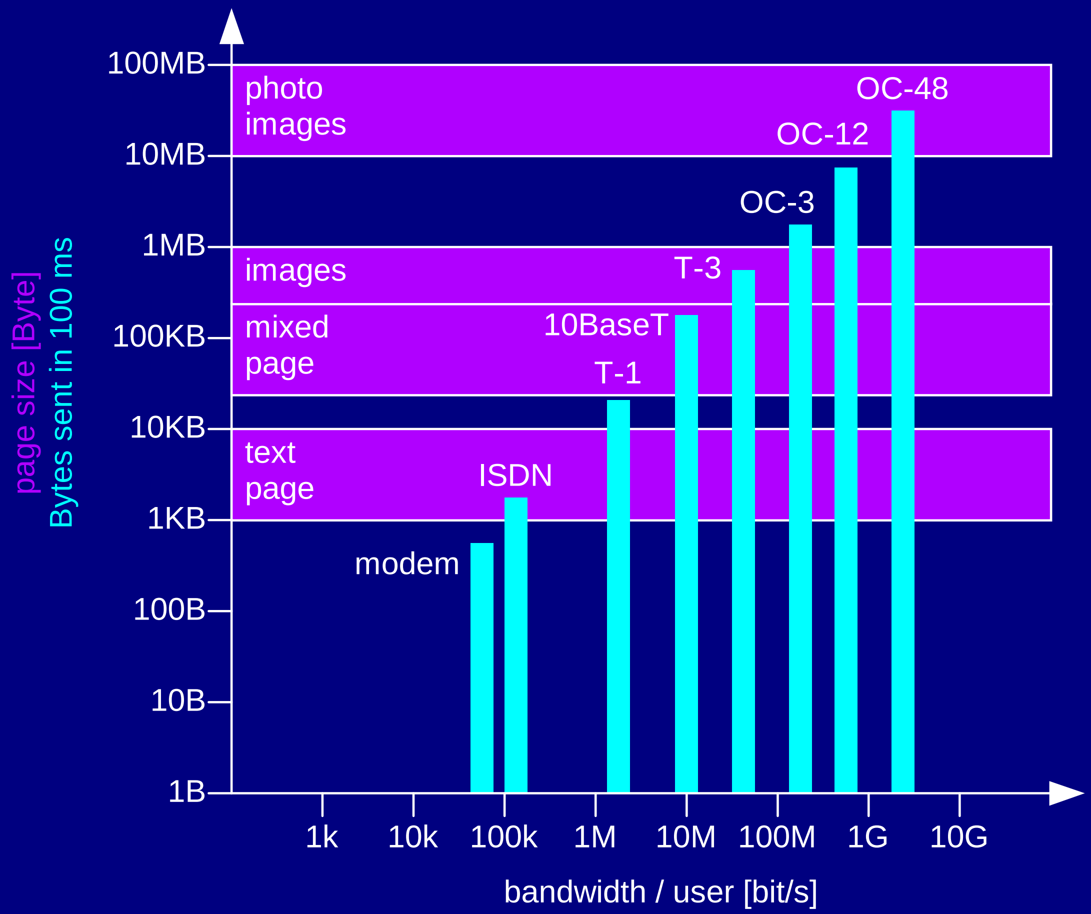
	Category		
Characteristic	Information Access	Telepresence	Distributed Computing and Storage
Application Relationship	client/server	peer-to-peer	varies
Bandwidth symmetry	asymmetric	symmetric	symmetric
Transfer granularity	large	continuous	varies
E2E Synchronisation	none	real time	varies

Application Categories
Information Access

- Client accessing information from a server
- Asymmetric bandwidth
- Response time is important metric
  - 100 ms $\lesssim T_r \lesssim$ 1 s target
- Significant bandwidth requirement
  - individual and aggregate
  - bandwidth challenged for interactive response bounds

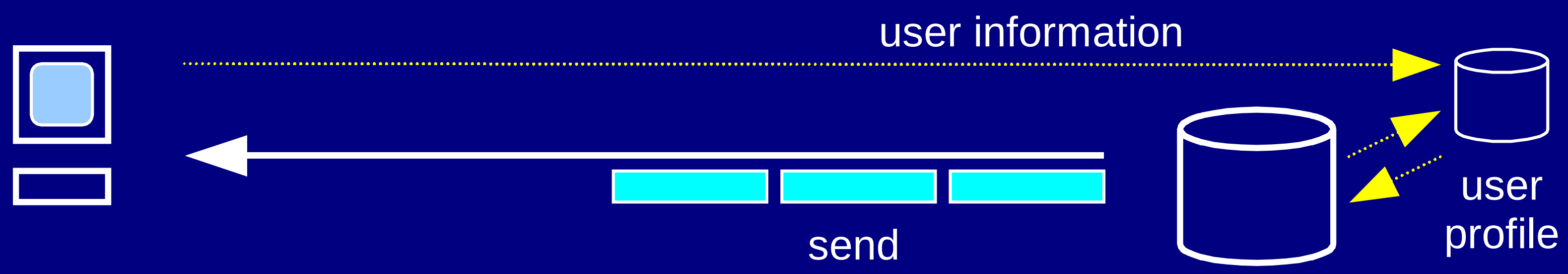
Application Categories

Information Access: Bandwidth



Application Categories

Information Access: Server Push



- Reduce response time by pushing data to user
  - server knows what user wants
  - data already present when user requests
  - reduces peak bandwidth

Client Pull vs. Server Push A-6B

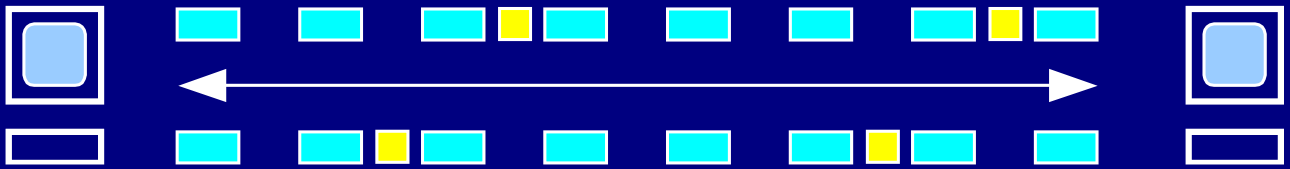
The side that that has the best knowledge of the client is the one that should initiate data transfer. Network load can be reduced by moving some of this knowledge to the server.

Application Categories
Information Access: P2P File Sharing

- Early P2P file sharing
  - P2P only used for content discovery (e.g. Guntella)
  - file download conventional client/server (HTTP)
- Emerging P2P file *swarming*
- Reduce response time by pushing data to user
  - users *exchange* pieces of files (e.g. BitTorrent)
  - modifies traditional information access semantics
    - characteristics of telepresence

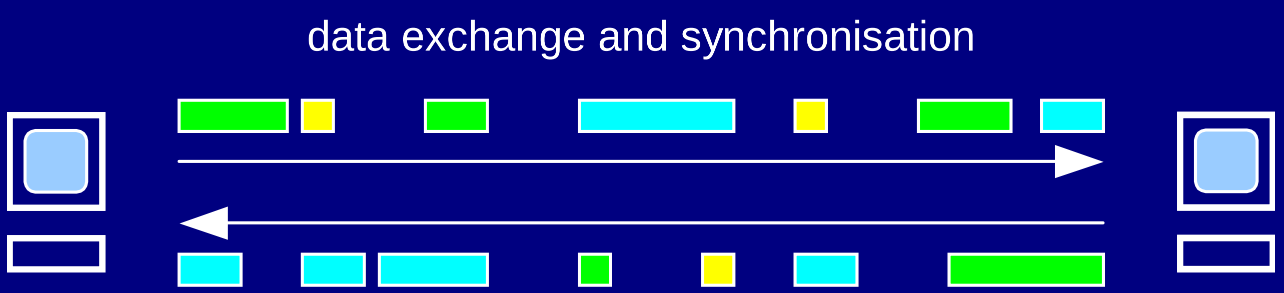
Application Categories
Telepresence

data streams with embedded synchronisation



- Peer-to-peer exchange of virtual presence
  - example: video conferencing
- Relatively symmetric bandwidth
- Bandwidth requirements
  - teleconferencing
    - aggregate bandwidth enhanced (operates at low speed)
  - applications with specific requirements
    - bandwidth enabled
      - example: telemedicine

Application Categories
Distributed Computing



- Distribution of computations beyond a room (>LAN)
- Arbitrary exchange of control, data, state
- Application-dependent partitioning critical

Constraints on Partitioning of Distributed Computations A-III.5
The ability to efficiently partition a distributed computation my be limited by the physical distribution of scarce computing and storage resources.

Application Categories
Composed Applications

- Complex applications consist of multiple components
- Composition of
  - information access
  - telepresence
  - distributed computing
- Example: distance learning
  - information access for class and reference materials
  - telepresence for student/teacher interaction

Networked Applications
Application Adaptation

8.1 Application characteristics

8.2 Application categories

8.3 Application adaptation

- 8.3.1 Latency reduction
- 8.3.2 Bandwidth improvement
- 8.3.3 Scaling and aggregation
- 8.3.4 Application layer framing
- 8.3.5 Mobile and wireless applications

8.4 Application–network interaction

Application Adaptation
Network Feedback

- Applications benefit from network feedback
  - dynamic controls need path characteristics
    - e.g. loss rate to determine dynamic FEC strength
    - e.g. congestion information to determine transmission rate

Application Adaptation depends on Network Feedback A-4F*f*

Application adaptation techniques involve the application reacting to information about the network, and are dependent on knowledge about the communication path. Some adaptations benefit from active cooperation from the network.

Application Adaptation
User Feedback

- Applications benefit from user feedback
  - exert controls
  - determine tradeoffs
    - e.g. resolution vs. frame rate
    - e.g. bandwidth vs. price

User Feedback to Adaptive Applications A-6B*u*

Adaptive applications benefit from user controls: either an explicit feedback loop, or implicit controls based on past behaviour or a user profile.

Application Adaptation
Synergy of Mechanisms

- Multiple adaptations
  - may interfere with one another
    - encoding may eliminate ability to cache
    - multiple encodings may *reduce* overall performance
  - may render application unusable

Application Adaptation Synergy A-IV_1

Many application adaptation techniques are mutually exclusive; some can occur only at the expense of others. They must be carefully implemented and coordinated to avoid interference.

Application Adaptation
Latency Reduction

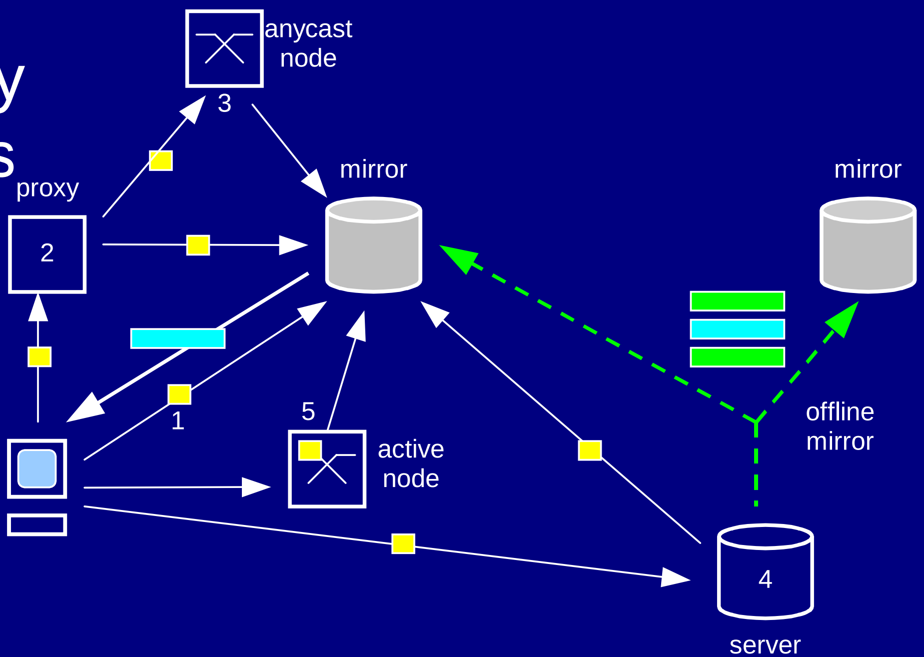
- User perceived delay
  - network latency
  - end system delay
  - application processing delay

Application Latency Principle A-1A/

Interapplication delays are the sum of all the constituent delays that cannot be parallelised. User-perceived delay is the ultimate measure of the performance of a high-speed system.

Application Adaptation
Mirroring

- Mirroring: locate copy of data close to users
  - location techniques:
    1. manual selection
    2. proxy redirection
    3. anycast
    4. server redirection
    5. active mirroring

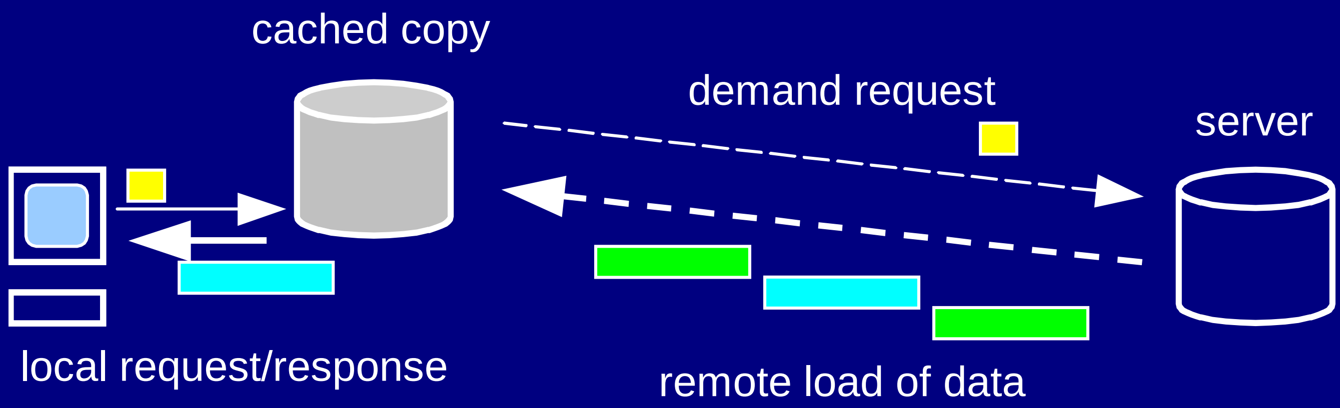


Exploit Spatial Locality to Reduce Latency A-7B*s*

Reduce the latency of access by locating data physically close to the users and applications that need it.

Application Adaptation

Local Caching



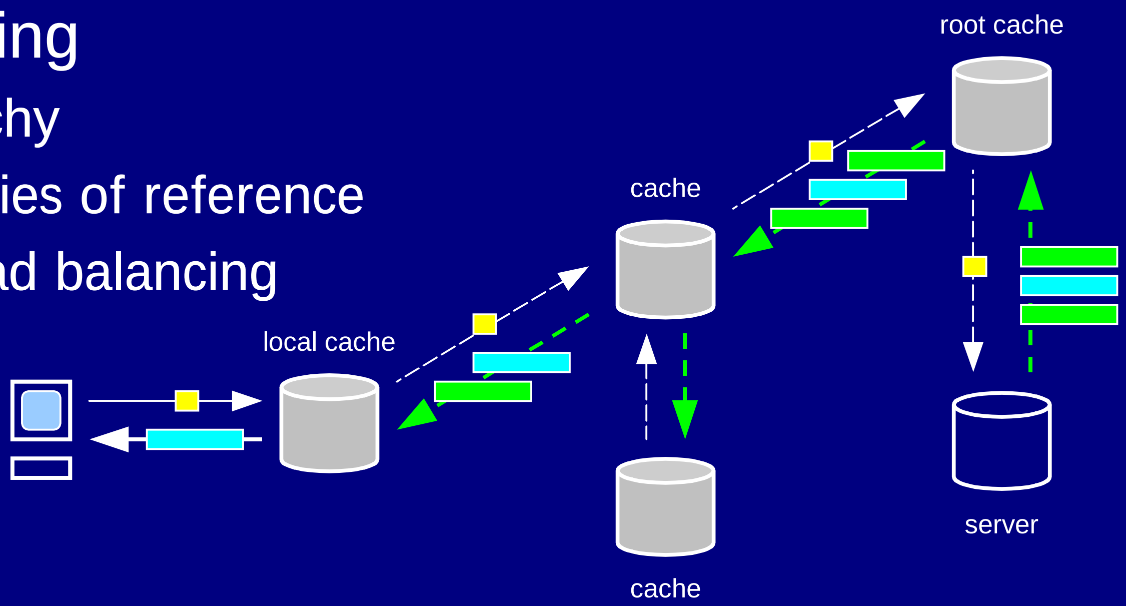
- Local caching: keep copy of data local (or close)
  - assuming it will be used again

Exploit Temporal Locality to Reduce Latency A-7B*t*

Reduce the latency of access by retaining data locally (or nearby) that is likely to be referenced in the future.

Application Adaptation
Network Caching

- Network caching
  - cache hierarchy
  - exploit localities of reference
  - automatic load balancing



Balance Network Resources by Hierarchical Caching A-5*c*

Balance aggregate network bandwidth against storage required by hierarchical caching.

Application Adaptation
Caching Dynamic Content

- Dynamic content cannot be easily cached
  - structure into cacheable units
    - isolate static from dynamic content
  - cache program code with data

Structure Data and Migrate Code for Dynamic Content A-7A*c*

Structure data from which content is dynamically generated into cacheable units. Cache the programs that generate the content along with the data.

Application Adaptation

Anticipation

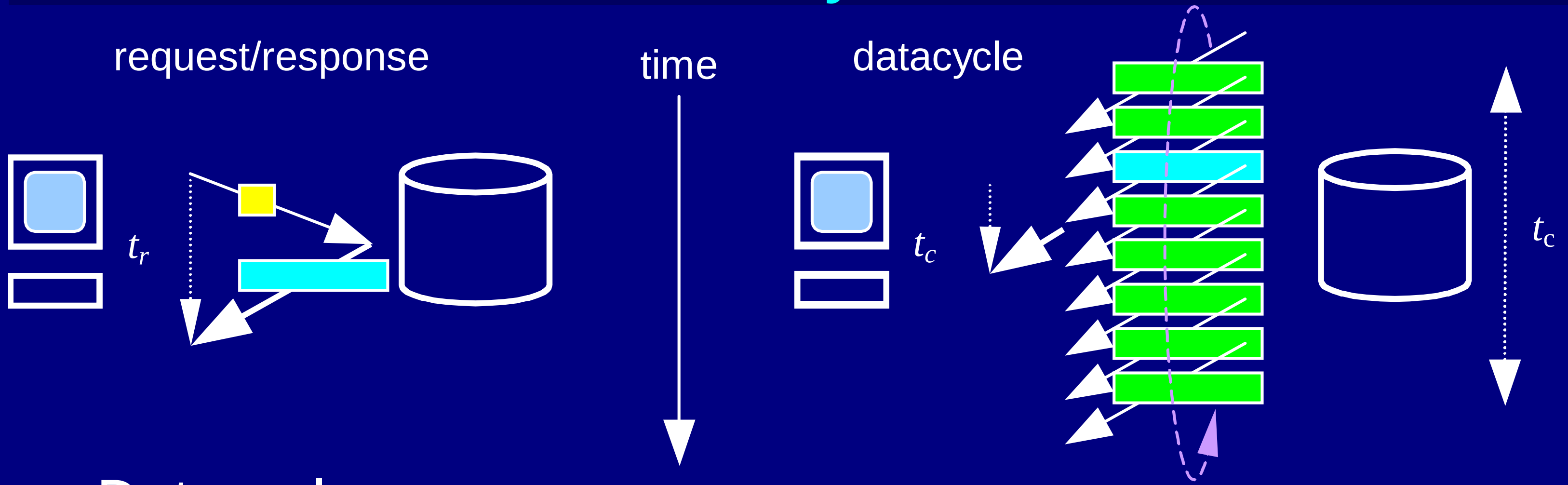
- Uses bandwidth to reduce latency
- Prefetch to anticipate future reference
  - exploit structured data (e.g. prefetch down hyperlink tree)
  - use past behaviour to anticipate future reference
- Application prediction to reduce data exchange
- Presend (push) when server knows what client wants
  - user profiles

Use Bandwidth to Reduce Latency A-2

Bandwidth can be utilised to reduce latency by anticipatory prefetching and presending.

Application Adaptation

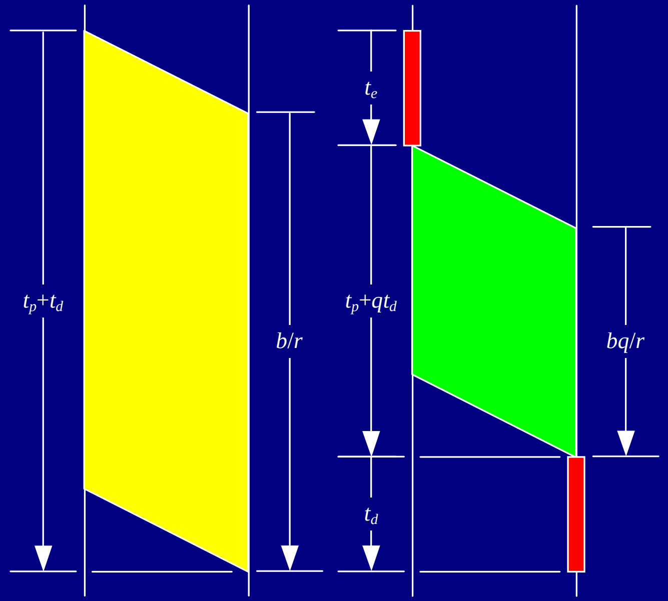
Datacycle



- Datacycle
  - when cycle time less than request/response loop
  - clients wait until desired data cycles past

Application Adaptation
Compression

- Compression to reduce delay
  - total delay includes
    - transmission delay
    - compression/decompression
  - benefit tradeoff between
    - path bandwidth
    - processing rate and cycles



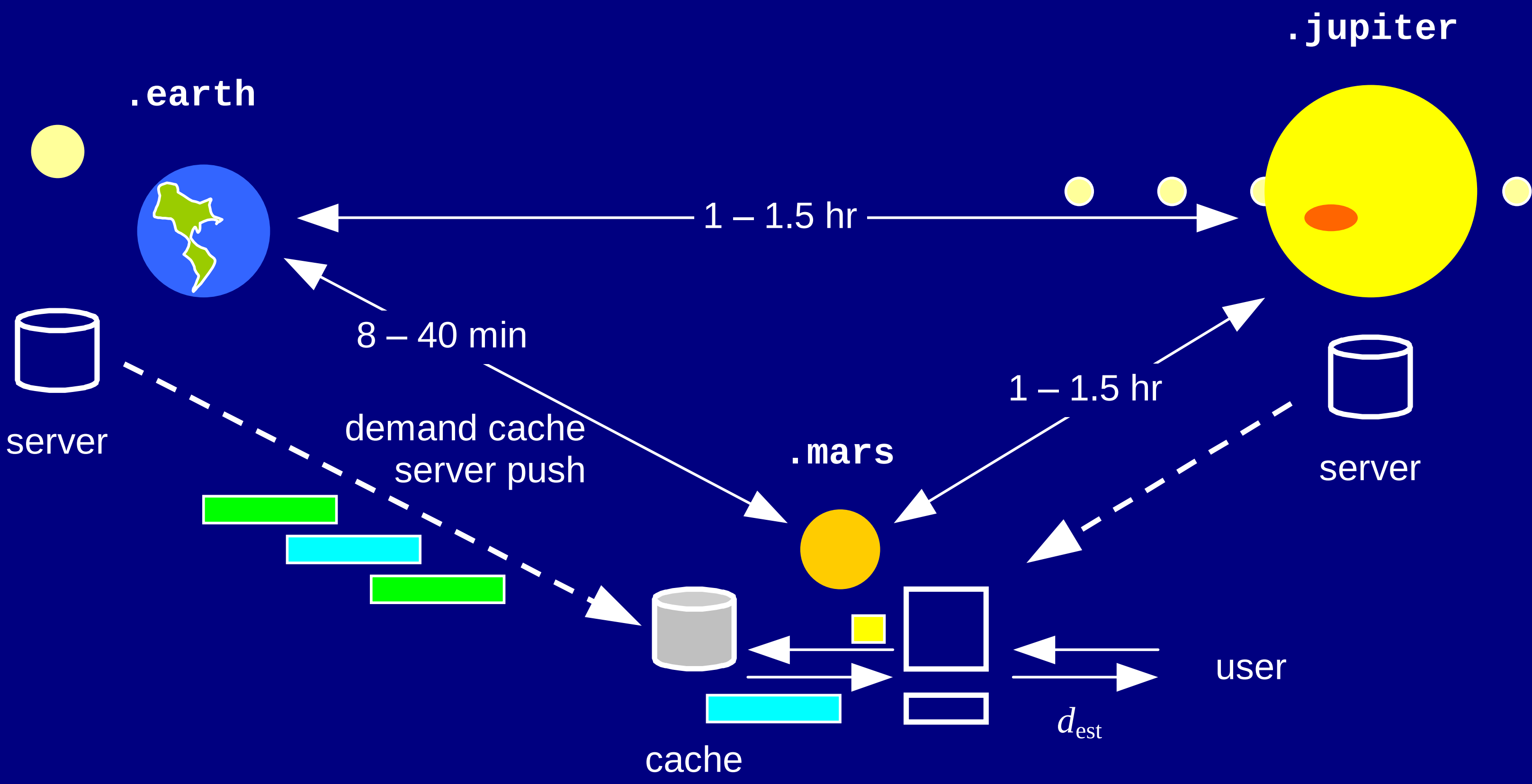
Compression of Data

A2B

Compression of data reduces bandwidth at the expense of the processing complexity and delay of the compression/decompression algorithms.

Latency Reduction

Example 8.1 Interplanetary Web Browsing



Latency Reduction

Example 8.1 Latency-Aware Web Browsing

- Links colored according to access characteristics
  - *green* close or fresh cached $\hat{t}_r < t_i$ $\hat{a} < a_i$
  - *yellow* cached but not fresh $\hat{t}_r < t_i$ $\hat{a} > a_i$
  - *red* long latency to receive $\hat{t}_r > t_i$ $\hat{a} > a_i$
  - *blue* no reasonable estimate possible
- Status bar at bottom of browser window
  - connectivity information
  - per URL information on mouse-over
    - freshness information a
    - estimates on t_r

[Sterbenz 2002]

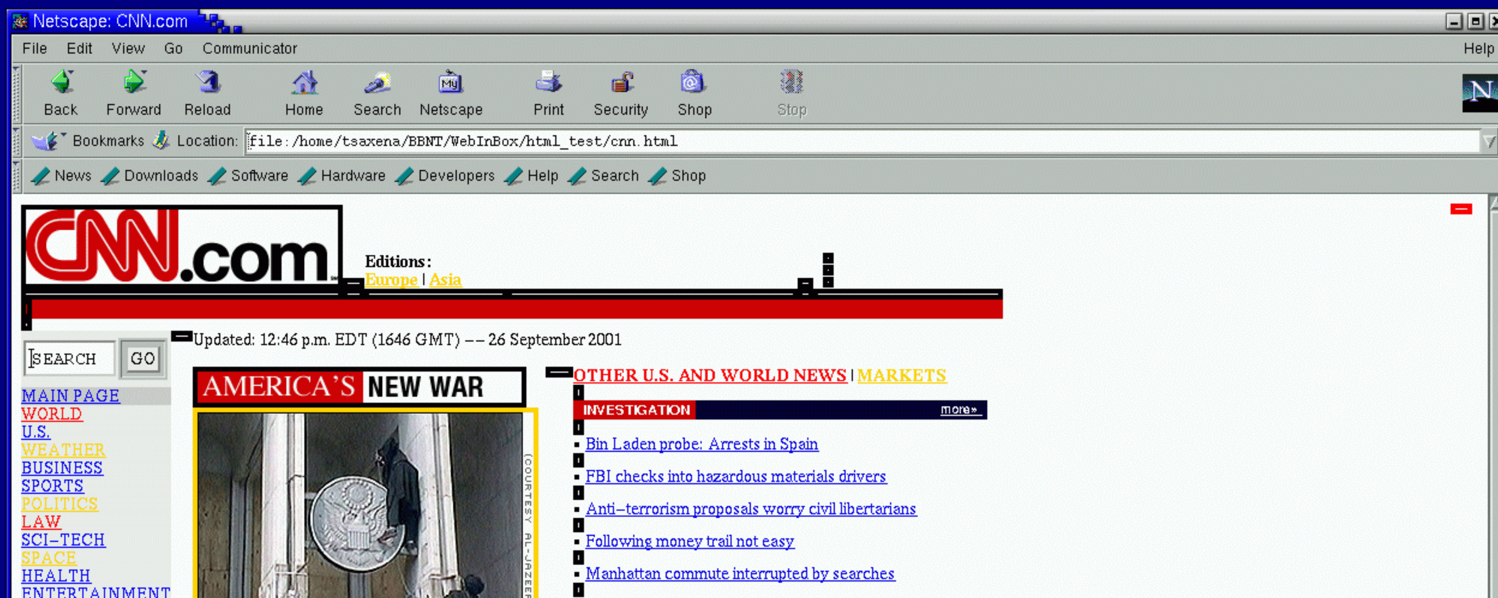
Latency Reduction

Example$_{8.1}$ Latency-Aware Web Browsing

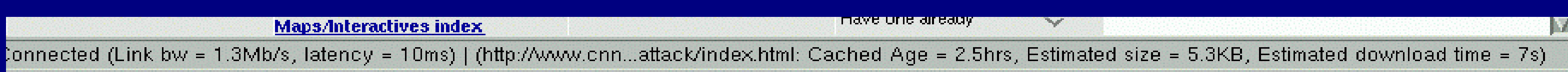
- Measured to *lighthouse* across known weak link
  - bandwidth
  - latency
- Sampled periodically to adjust response time est.
- Examples:
  - wireless LAN lighthouse at base station
  - ship or plane lighthouse at fixed access nodes

Latency Reduction

Example 8.1 Latency-Aware Web Browsing



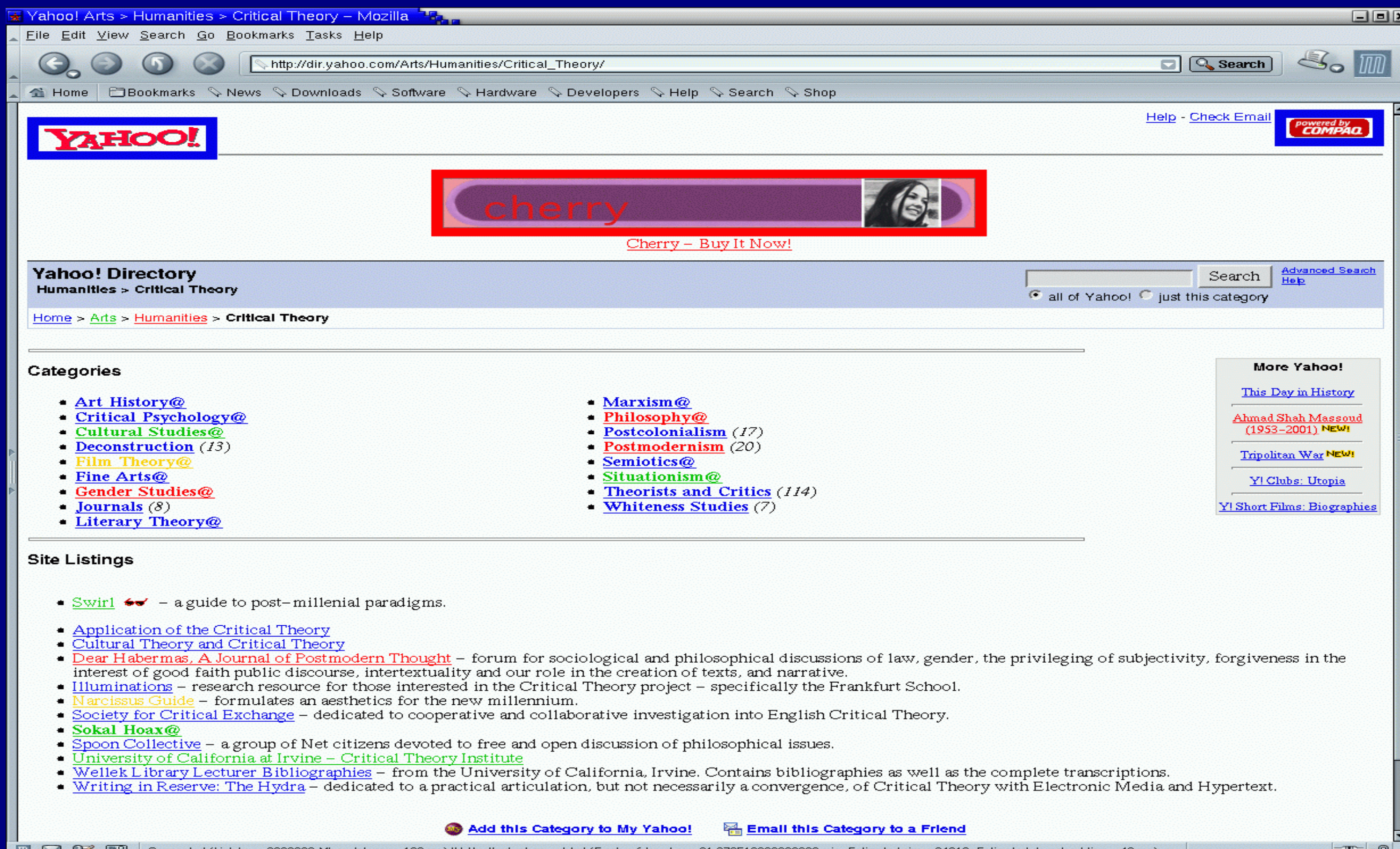
- Link color gives high level {*fast*, *old*, *slow*, *unknown*}



- Status bar indicates global and *per* link details

Latency Reduction

Example$_{8.1}$ Latency-Aware Web Browsing



Latency Reduction

Example$_{8.1}$ Latency-Aware Web Browsing

- Click type determines fetch action
  - left click: “normal” behavior
    - get cached if available
    - profile based action
  - right click gives options
    - fetch definitive
      refresh window when definitive copy arrives
    - nonblocking fetch
      definitive copy in new window when available
- View menu selection
  - allows display of unmodified page

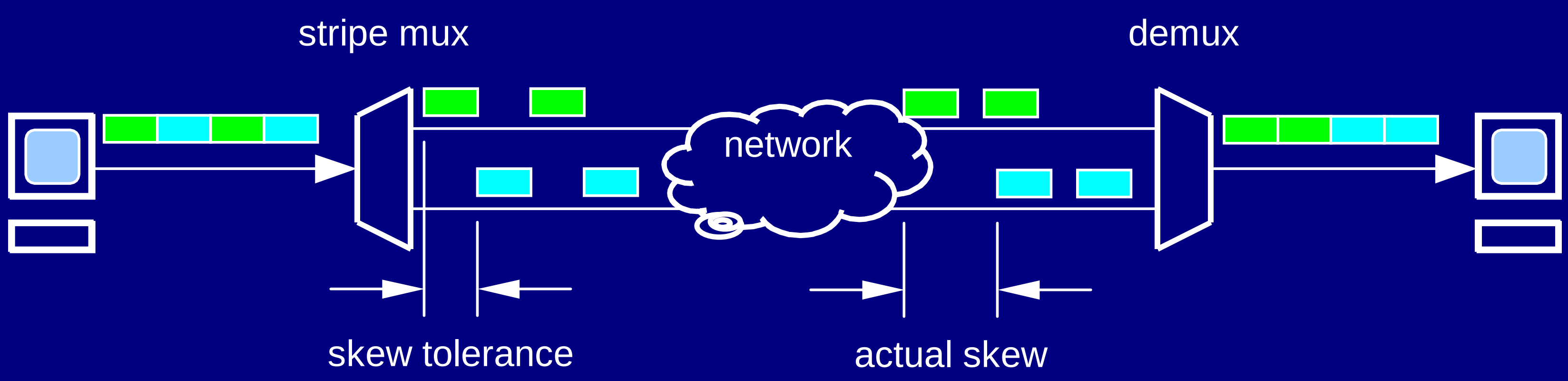
Application Adaptation
Bandwidth Improvement

- Bandwidth constrains transmission delay
- Techniques to improve apparent bandwidth
  - mirroring and caching: reduce individual server load
  - anticipation: smooth traffic and reduce peak bandwidth
  - multicast: reduce aggregate bandwidth
  - rate adaptation and layered coding
  - ADU size
    - larger reduces total per ADU processing (e.g. headers)

Application Bandwidth Principle A-1A*b*

The bandwidth perceived by the user is the minimum of all bandwidths on a path, including through the application itself.

Application Adaptation
Bandwidth Improvement



- Techniques to improve apparent bandwidth
  - striping: exploit parallel paths through the network

Application Adaptation
Scaling and Aggregation

- Applications must scale as needed
  - large numbers of users
  - large numbers of paths
- Scale by aggregation

Aggregation of Application Operations A-5B

Reduce the cost of all per-connection or per-application operations by aggregation. Support ways to aggregate requests and responses between the application and network, and for the application to control the frequency of response and size of aggregation.

Application Adaptation

Scaling and Aggregation

- Aggregation techniques complicate implementation
  - amortisation of connection state can occur at transport layer
  - trade off application vs. transport layer complexity

Partitioning of Application vs. Protocol Functionality A-1C

Keep the network impact of application design choices in mind.
Not all functions are best deployed in an application.

Application Adaptation
Application Layer Framing

- Application layer framing
  - match application characteristics to network
  - decrease overhead of protocol
    - encapsulation
    - control transfer
    - decapsulation

Application – Network Layer Synergy A-4C

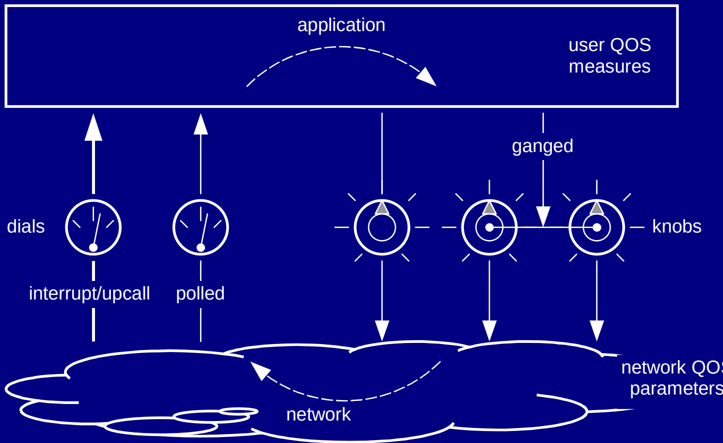
Increase performance by matching application characteristics to the network. Use ALF to decrease the overhead of protocol encapsulation, control transfer, and decapsulation.

Networked Applications

Application–Network Interaction

Application–Network Interaction
Knobs and Dials

- Knobs influence network
  - may be ganged
- Dials feed information back
  - interrupt
  - polled
- Interfaces should scale



API Scalability Principle A-4I

All interfaces between an application and network should scale with respect to bandwidth, latency, and number (nodes, end systems, and users). Parameters should have a scale factor (order of magnitude exponent), a precision factor (multiplier), and a variance.

Application–Network Interaction
Knobs and Dials

- Knobs should correspond to *actual* network actions
- Dials should correspond to *actual* properties...

...otherwise apps can't properly interact with network

Knobs Should Correspond to Network Actions A-4C*k*

A knob should not be built into the network that does not correspond to real controllable action.

Dials Should Correspond to Observable Properties A-4C*d*

A dial should not be provided that does not correspond to a real observable parameter.

Application–Network Interaction
Dials

- Synchronous (polled) dials
  - application predictable events
- Asynchronous (interrupt-driven) dials
  - unpredictable events
  - highly dynamic network conditions
    - mobility
    - time-varying wireless channels

Support Synchronous and Asynchronous Dials A-4H

Network instrumentation should support both asynchronous (interrupt-driven) and synchronous (polled) feedback mechanisms.

Application–Network Interaction
Dials

- Dials may be imprecise
  - imprecise instantaneous values
  - rapidly time-varying values
- Provide a variance to imprecise values
  - allows application to properly interpret dial

Provide Variance Values to Imprecise Values A-5Av

The quality of feedback depends on the stability and precision of the information provided. If information is approximated or aggregated and the application does not know the degree of imprecision a priori, provide an explicit variance to the value.

Application–Network Interaction
Transparency and Dependence

- Abstraction can be useful
- Hiding can be harmful
- Translucency is better than transparency
  - *functional* transparency
    with necessary performance parameters exposed

Functional Transparency and Abstraction Should not Result in Hiding A-4F*h*

Layering is designed around abstraction, providing a simpler representation of a complicated interface. Abstraction can obscure a necessary property or parameter, resulting in hiding. Hiding is not a desirable property of layering.

Application–Network Interaction
Transparency and Dependence

- Functional transparency benefits applications
  - location independence
- Applications may benefit from latency awareness
  - location translucency

Location Independence Should Not Hide Latency A-4F/

Interfaces should be provided to access data that are not dependent on physical location, but latency and other QOS measures should be available to applications that can benefit from this knowledge.

Application–Network Interaction

Transparency and Dependence

- Old applications can’t use new knobs and dials
- Provide default behaviour for legacy applications

Default to Support Legacy Applications A-III.7

Networks should provide a meaningful degenerate behaviour to support legacy applications, or applications whose capabilities are outside the scope of the system.