INF5061:
Multimedia data communication using network processors

Multimedia Network Processor Examples

30/9 - 2005

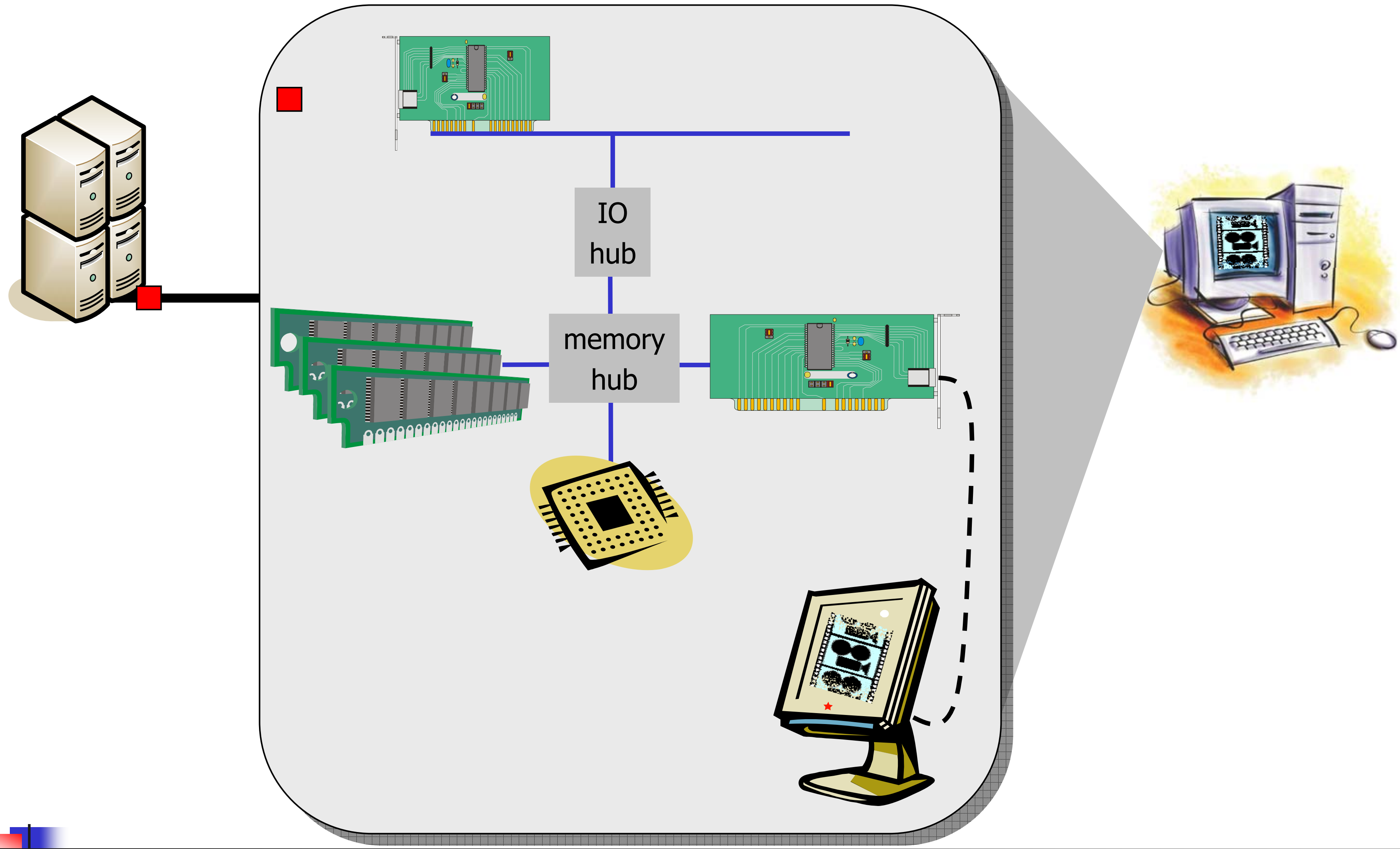
Overview

- Video Client Operations
- Multicast Video-Quality Adjustment
- Booster Boxes

Example:

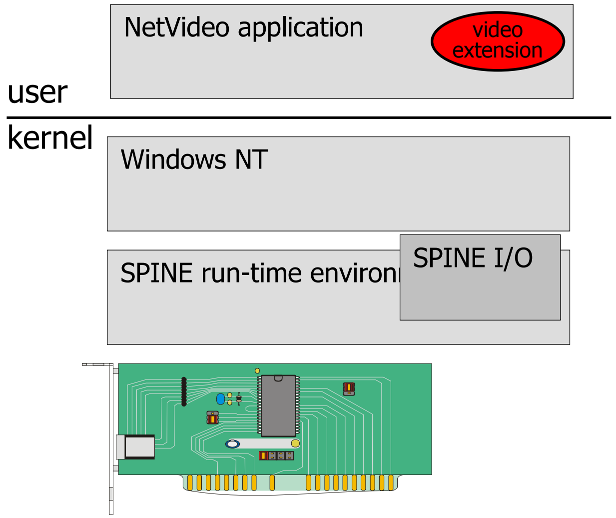
Video Client Operations

Video Client Operations



SPINE: Video Client Operations

- **Fiuczynski et. al. 1998**:
  - use an extensible execution environment to enable applications to compute on the network card
  - SPINE extends SPIN (an extensible operating system) to the network interface
    - define I/O extensions in Modula-3 (type-safe, Pascal-like)
    - these I/O modules may be dynamically loaded onto the NIC, or into the kernel (as in SPIN)

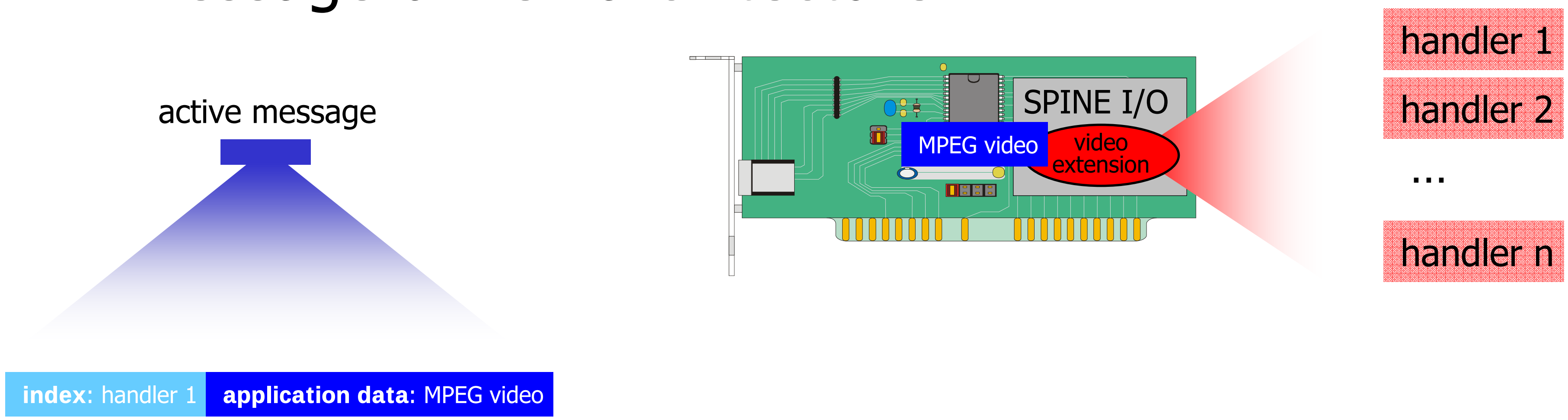


  - perform video client operations on-board a *Myrinet* network card (33 Mhz LANai CPU, 256 KB SRAM)

SPINE: Video Client Operations

Fiuczynski et. al. 1998

- A message-driven architecture

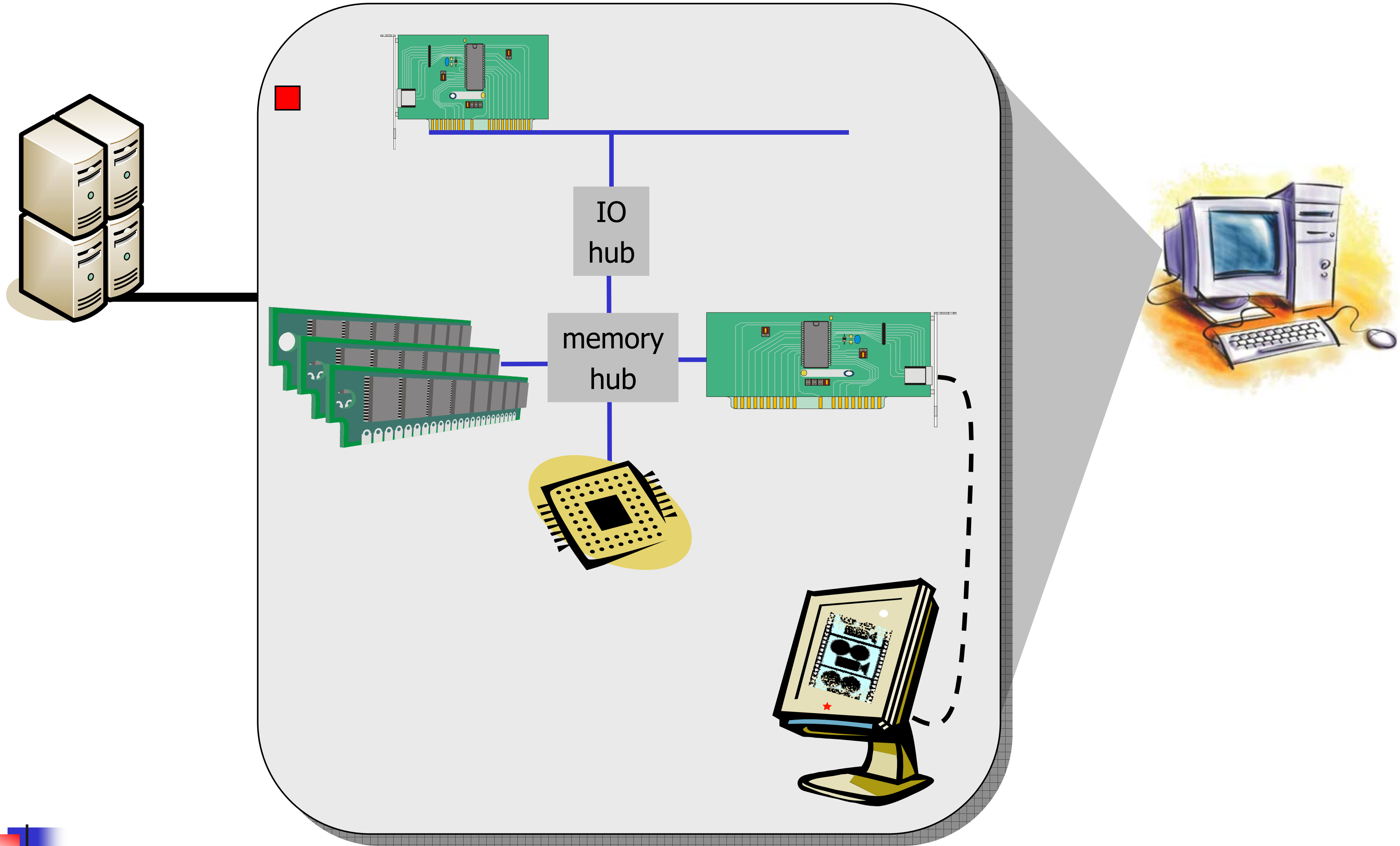


- Application creates the framing window and informs the SPINE extension about the coordinates

⇨ SPINE puts video data in corresponding frame buffer memory according to window placement on screen

SPINE: Video Client Operations

Fiuczynski et. al. 1998



SPINE: Video Client Operations

Fiuczynski et. al. 1998

- Evaluation
  - managed to support several clients in different windows
  - data DMA'ed to frame buffer
    → zero host CPU requirement for video client(s)
  - a 33 Mhz LANai CPU too slow to do large video decoding operations
    → server converted MPEG to raw bitmap before sending
    → only I/O processing and data movement offloading
  - frequent synchronization between host and device-based component is expensive

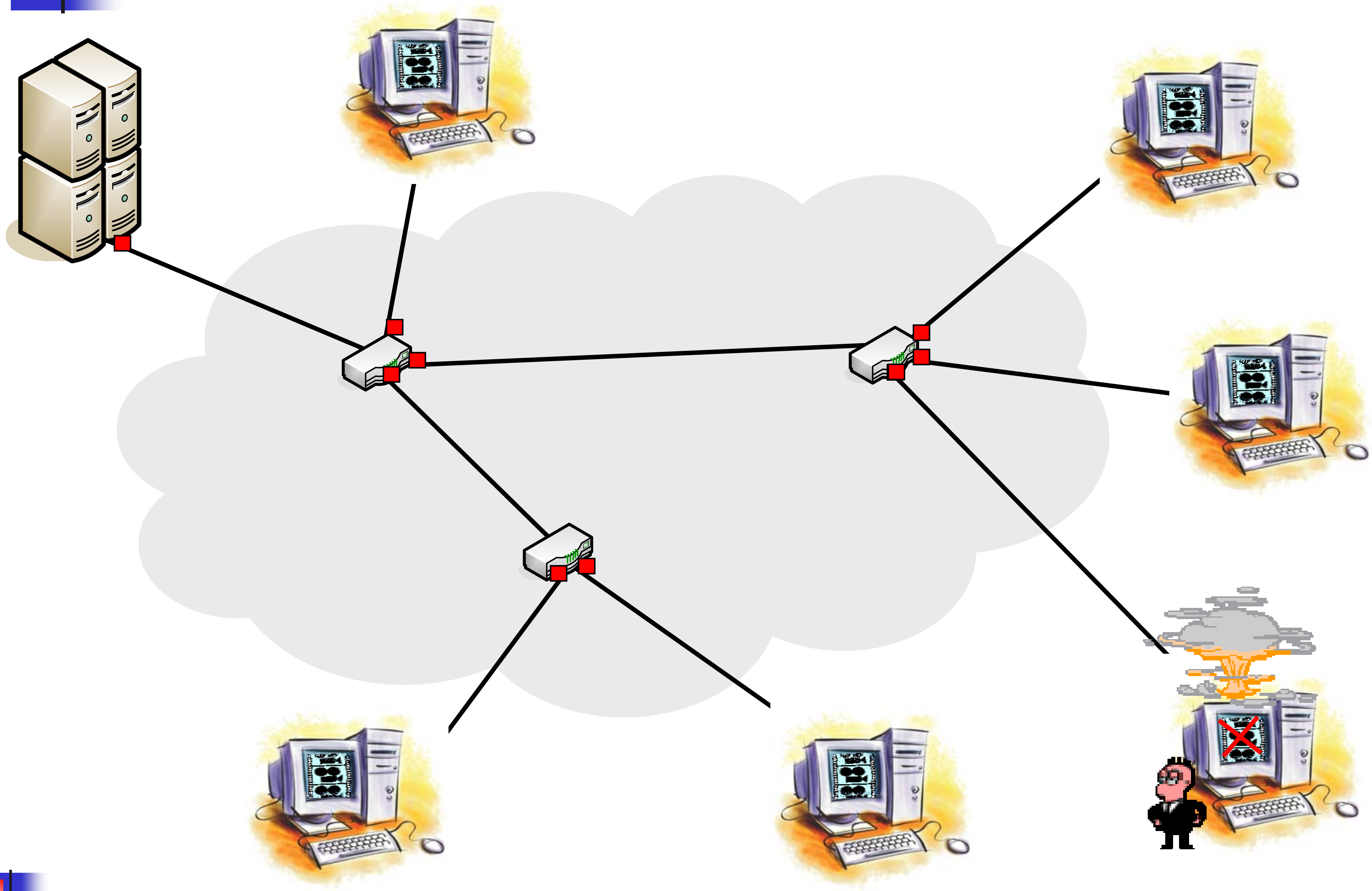
SPINE: Internet Protocol Routing

- A SPINE router extension on the network processor
- Able to fully offload host CPU
- Forwarding latency 6% slower compared to host (but 33MHz embedded CPU vs. 200MHz Pentium Pro)

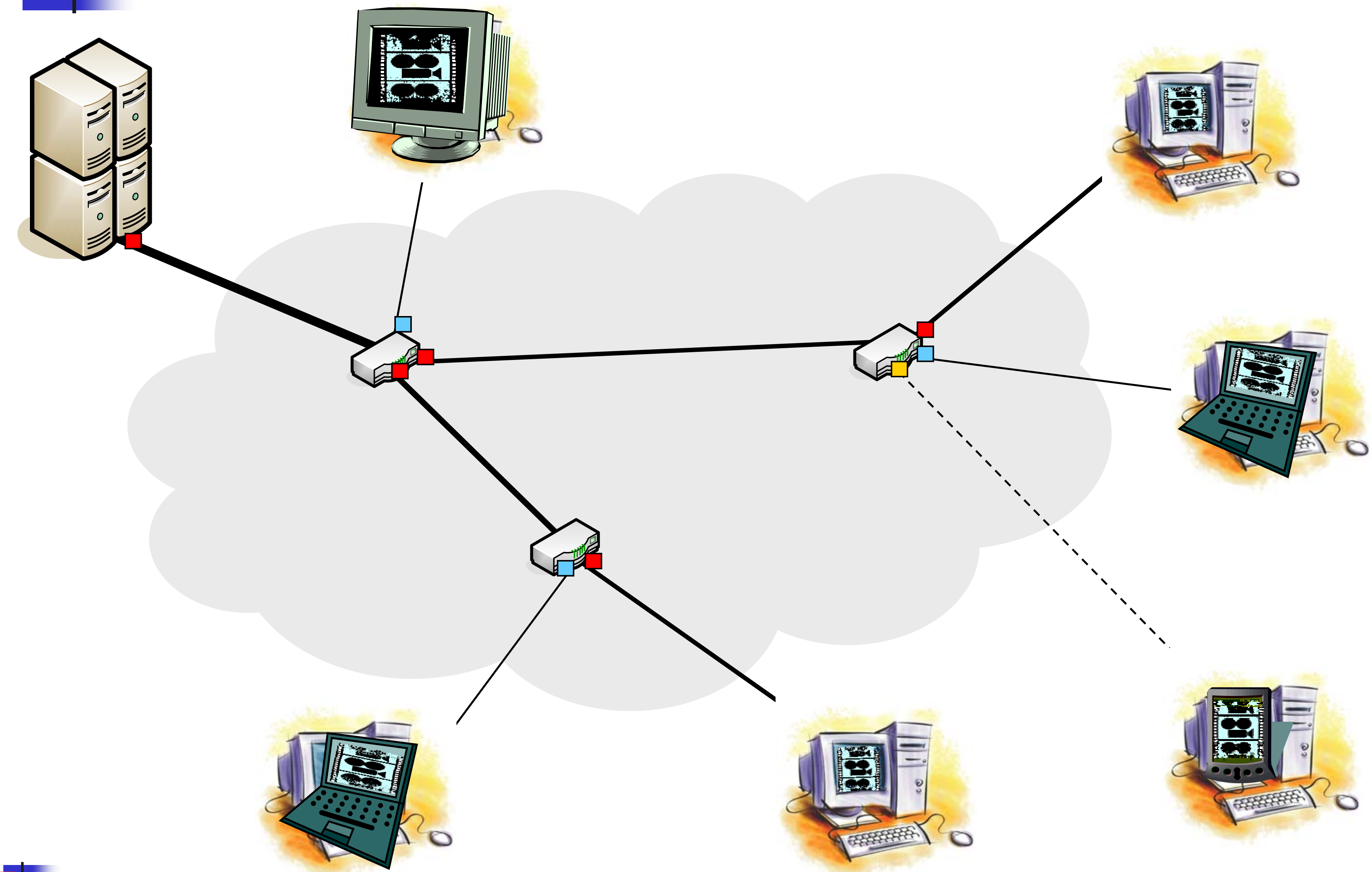
Example:

Multicast Video-Quality Adjustment

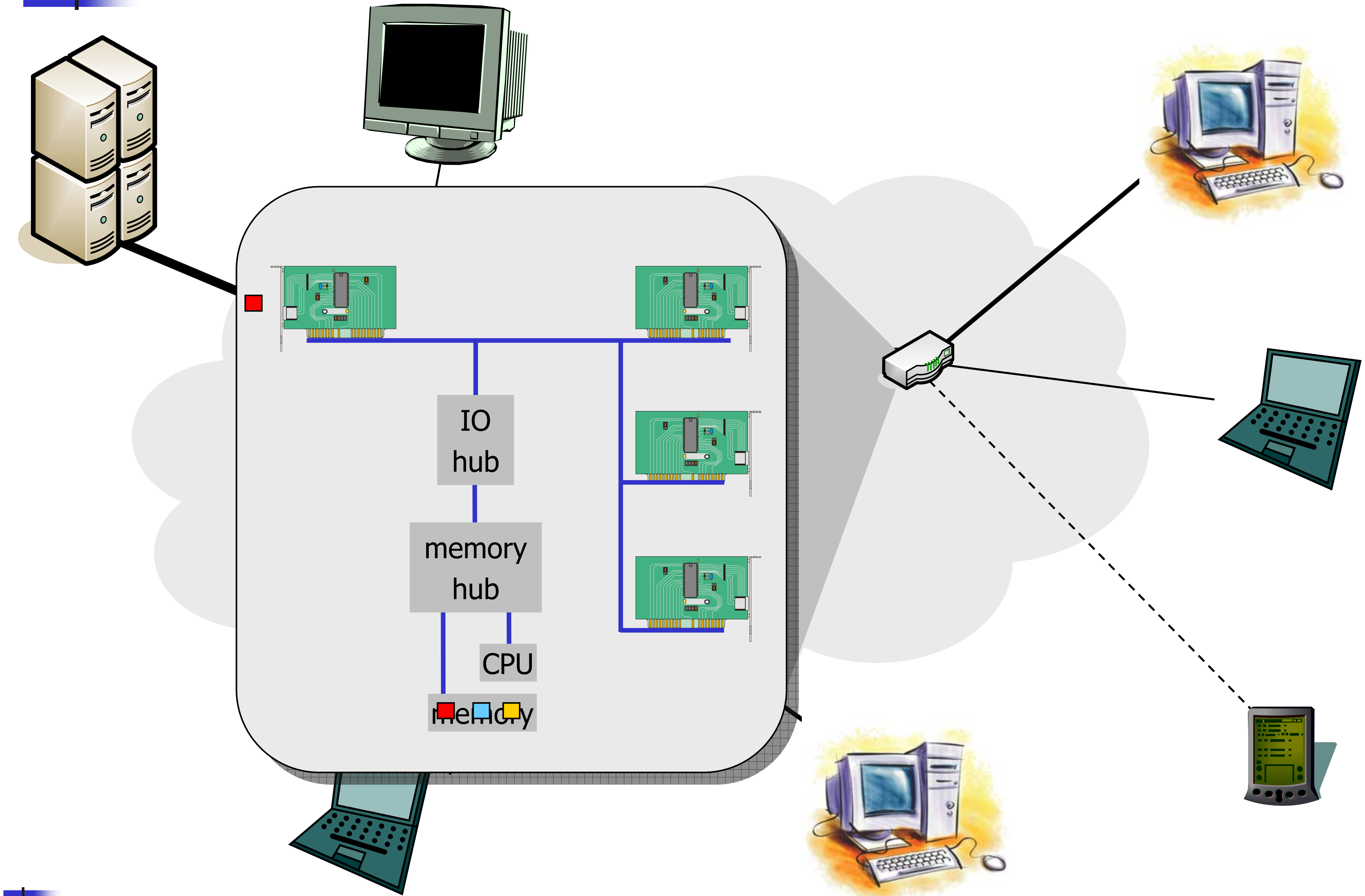
Multicast Video-Quality Adjustment



Multicast Video-Quality Adjustment



Multicast Video-Quality Adjustment



Multicast Video-Quality Adjustment

- Several ways to do video-quality adjustments
  - frame dropping
  - re-quantization
  - scalable video codec
  - …

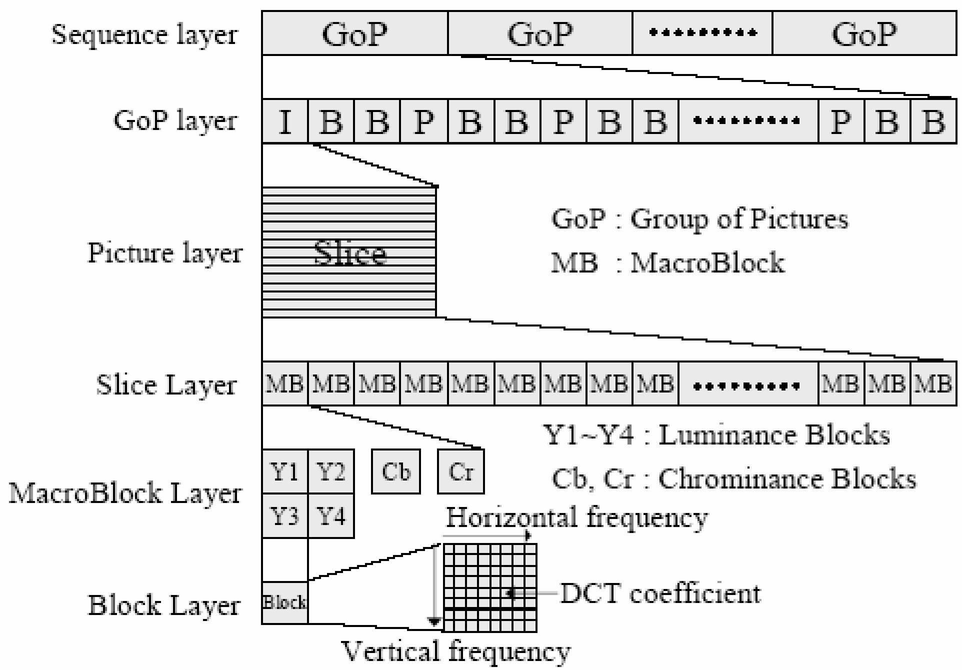
- **Yamada et. al. 2002**:
  use low-pass filter to eliminate high-frequency components of the MPEG-2 video signal and thus reduce data rate
  - determine a low-pass parameter for each GOP
  - use low-pass parameter to calculate how many DCT coefficients to remove from each macro block in a picture
  - by eliminating the specified number of DCT coefficients the video data rate is reduced

  - *implemented the low-pass filter on an IXP1200*

Multicast Video-Quality Adjustment

Yamada et. al. 2002

- Segmentation of MPEG-2 data
  - slice = 16 bit high stripes
  - macroblock = 16 x 16 bit square
    - four 8 x 8 luminance
    - two 8 x 8 chrominance
    - ⇨ DCT transformed with coefficients sorted in ascending order



- Data packetization for video filtering
  - 720 x 576 pixels frames and 30 fps
  - ⇨ 36 “slices” with 45 macroblocks per frame

  - Each slice = one packet
  - 8 Mbps stream → ~7Kb per packet

Multicast Video-Quality Adjustment

Yamada et. al. 2002

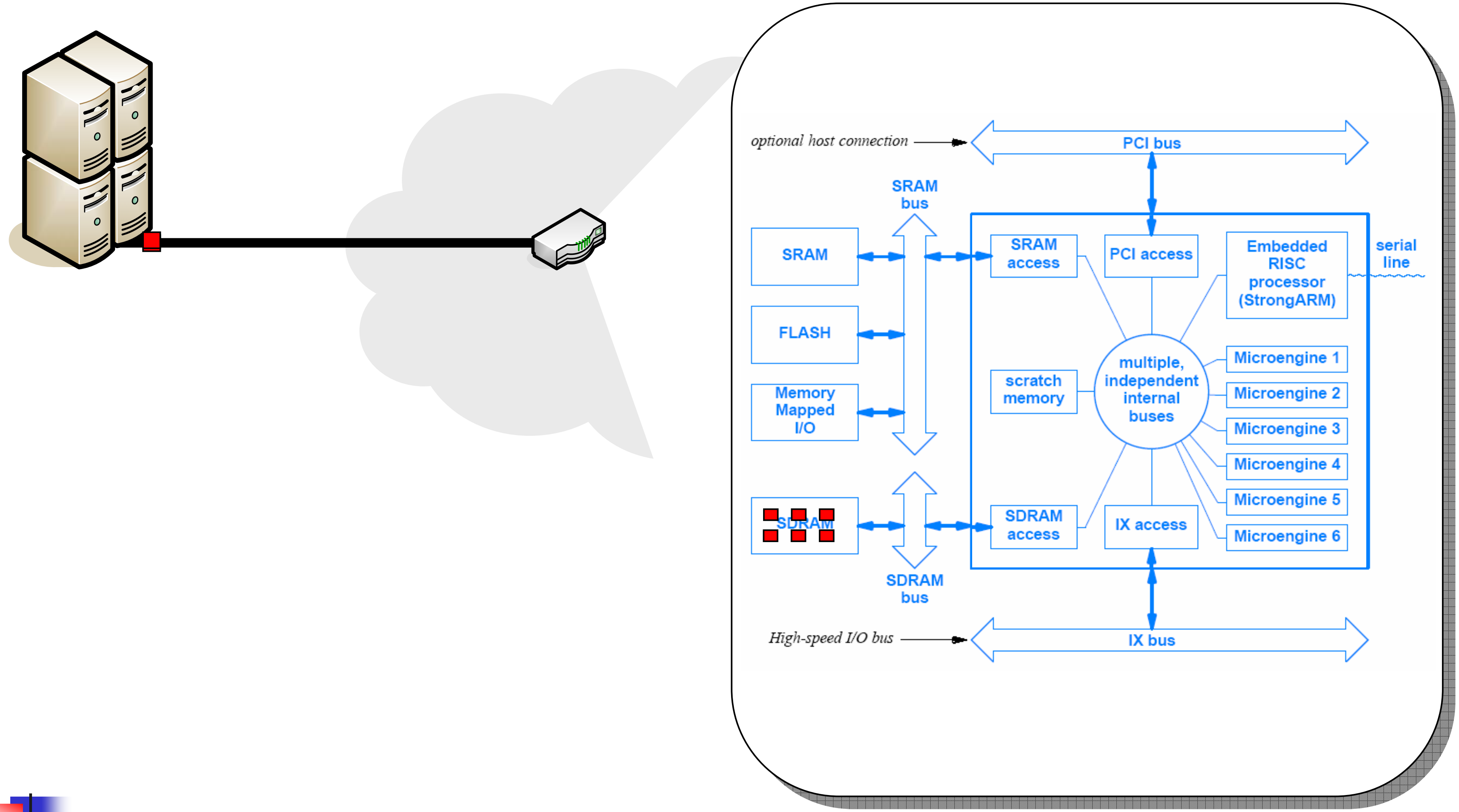
- Low-pass filter on IXP1200
  - parallel execution on 200MHz StrongARM and microengines
  - 24 MB DRAM devoted to StrongARM only
  - 8 MB DRAM and 8 MB SRAM shared

  - test-filtering program on a regular PC determined work-distribution
    - 75% of data from the block layer
    - 56% of the processing overhead is due to DCT

  - ➡ five step algorithm:
    1. StrongArm receives packet → copy to shared memory area
    2. StrongARM process headers and generate macroblocks (in shared memory)
    3. microengines read data and information from shared memory and perform quality adjustments on each block
    4. StrongARM checks if the last macroblock is processed (if not, go to 2)
    5. StrongARM rebuilds packet

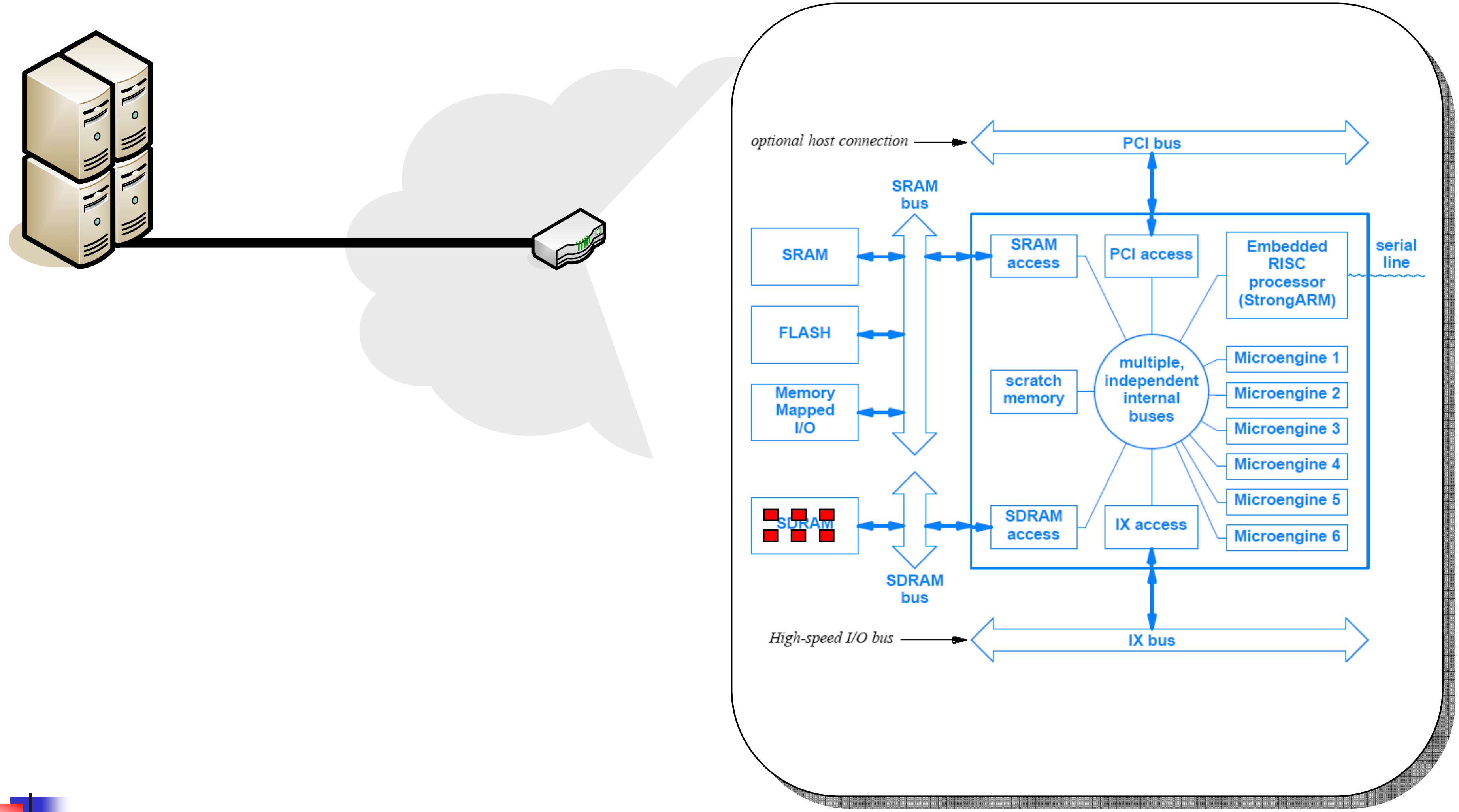
Multicast Video-Quality Adjustment

Yamada et. al. 2002



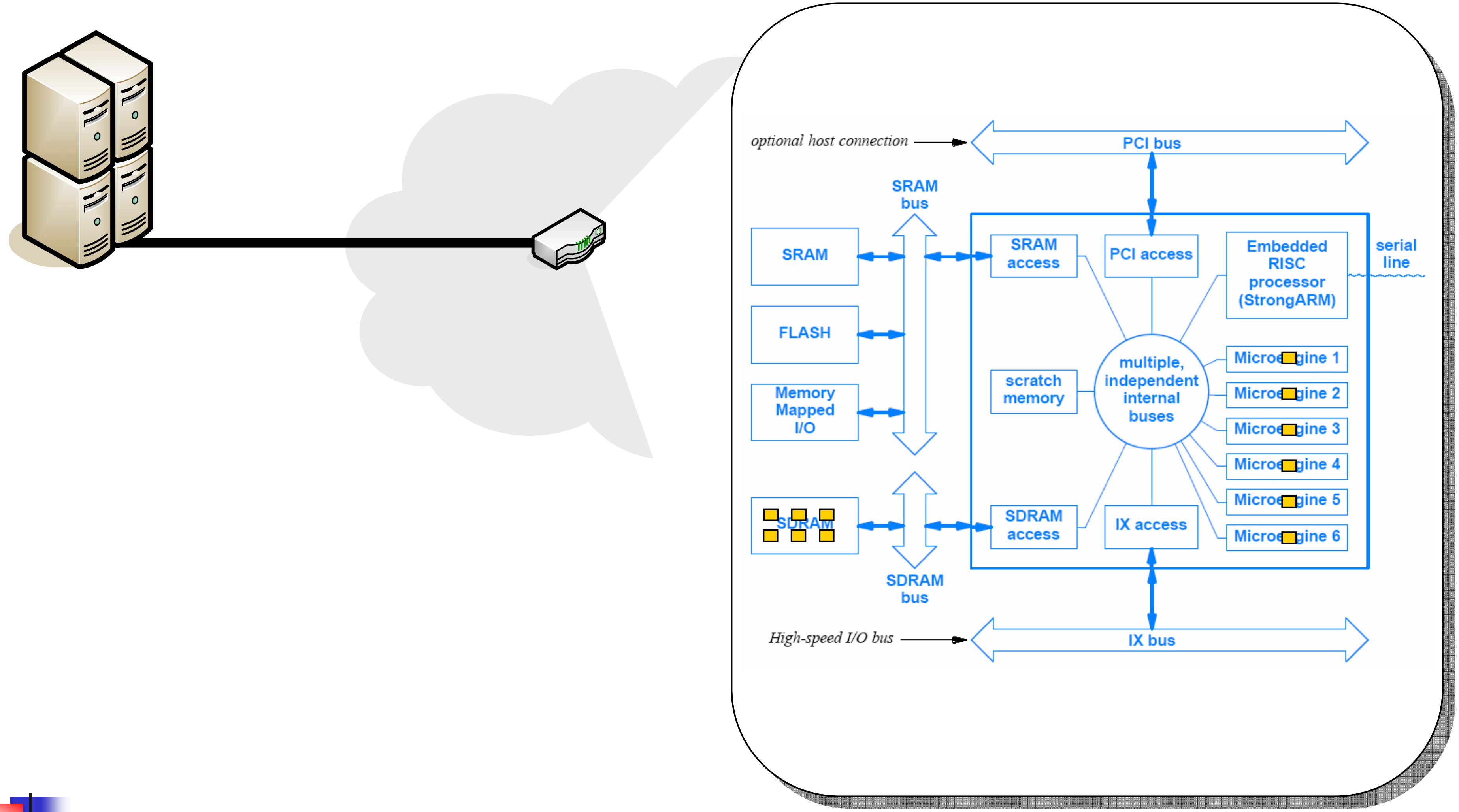
Multicast Video-Quality Adjustment

Yamada et. al. 2002



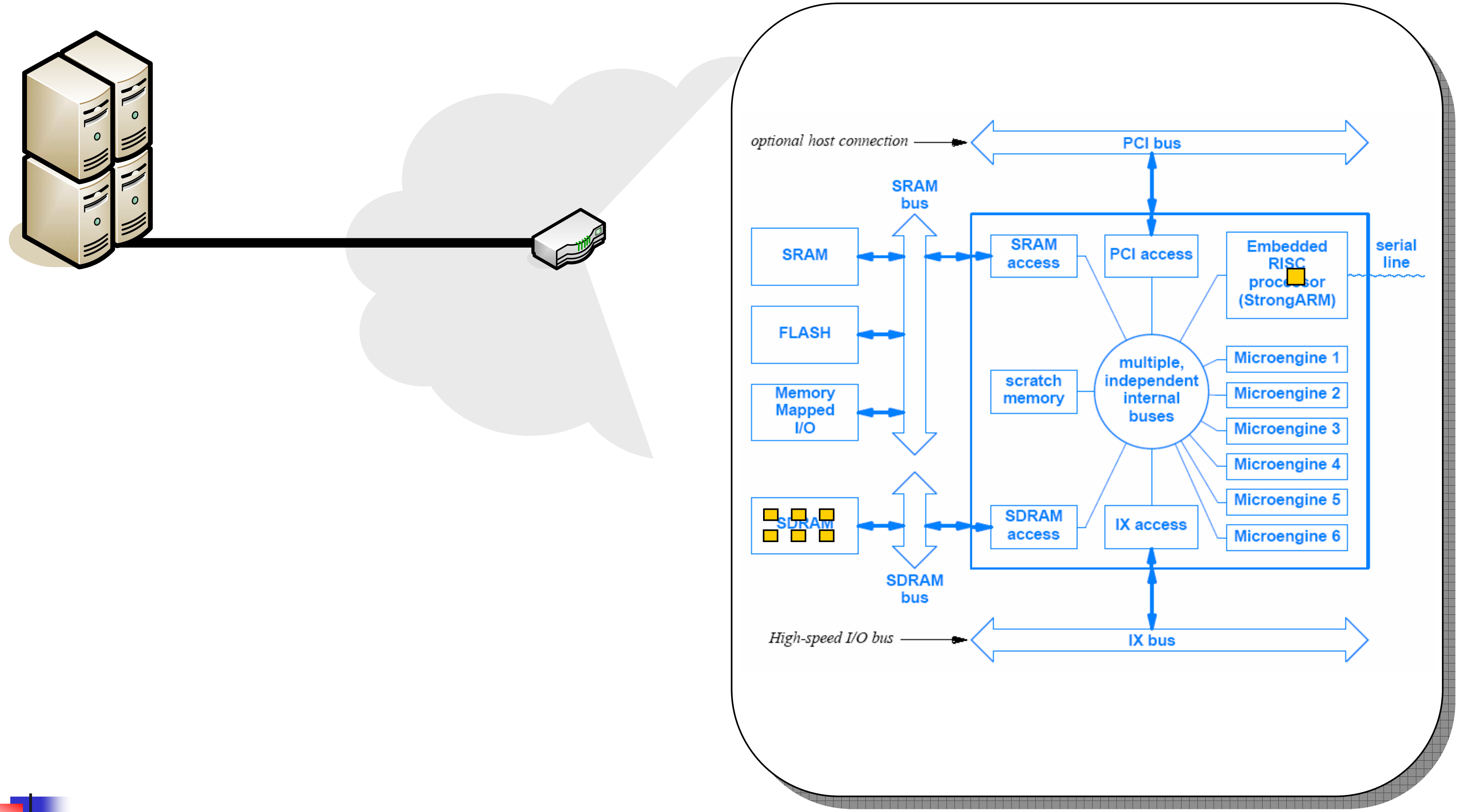
Multicast Video-Quality Adjustment

Yamada et. al. 2002



Multicast Video-Quality Adjustment

Yamada et. al. 2002



Multicast Video-Quality Adjustment

Yamada et. al. 2002

- Evaluation – three scenarios tested

  - StrongARM only → 550 kbps
  - StrongARM + 1 microengine → 350 kbps
  - StrongARm + all microengines → 1350 kbps

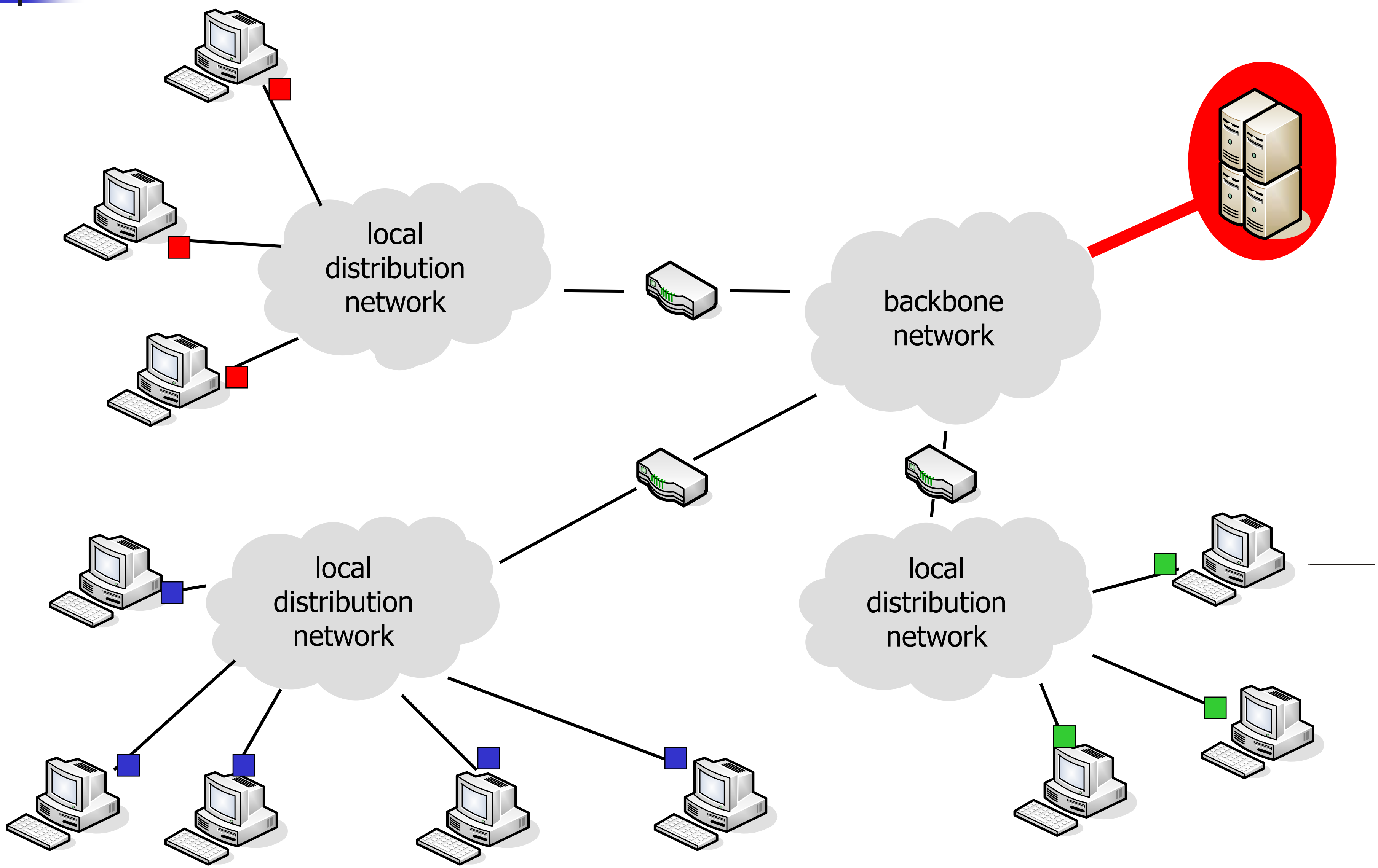
  - achieved **real-time** transcoding not enough for practical purposes, but distribution of workload is nice

Example:

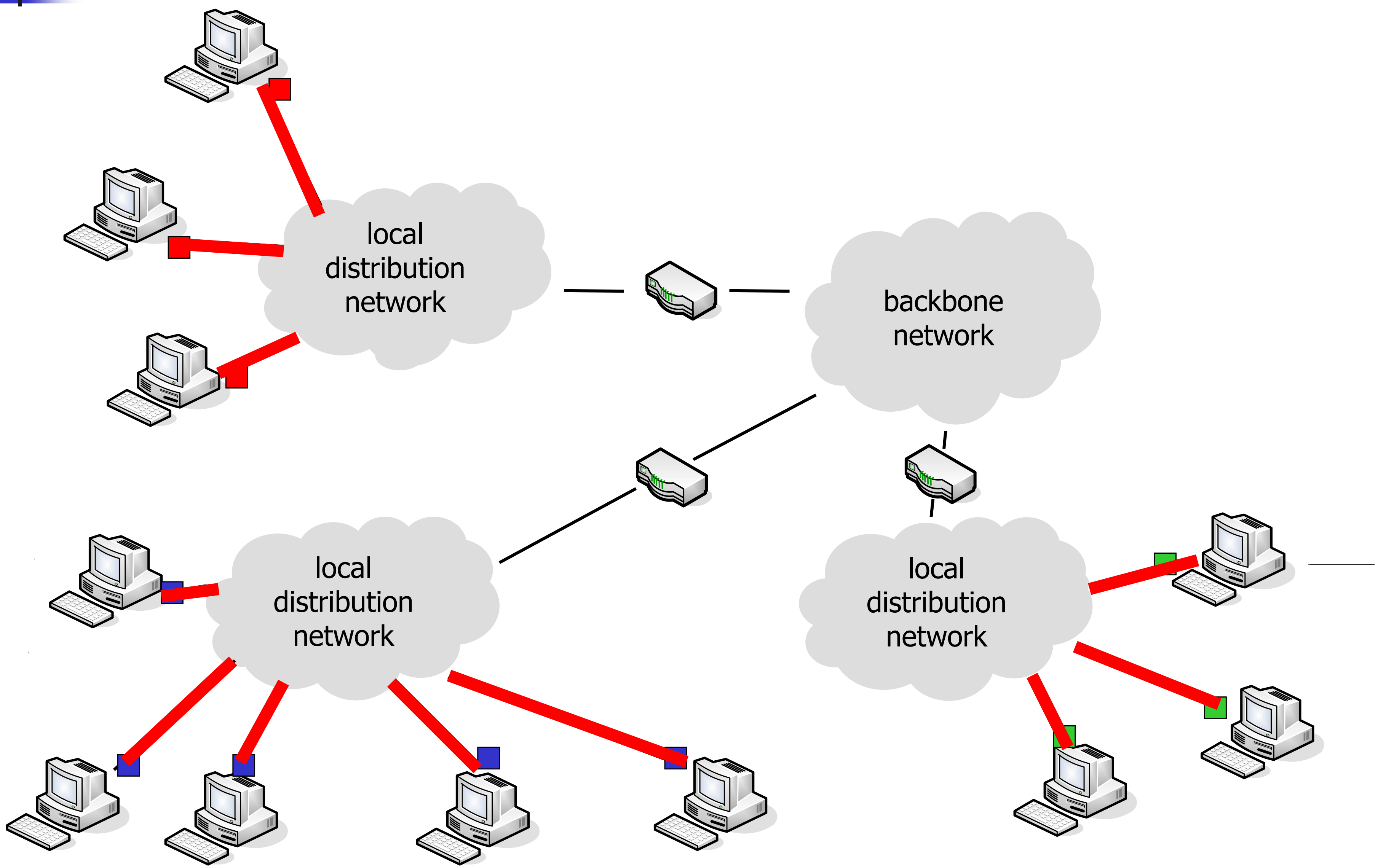
Booster Boxes

slide content and structure mainly from the NetGames 2002 presentation by Bauer, Rooney and Scotton

Client-Server



Peer-to-peer


local
distribution
network
backbone
network
local
distribution
network
local
distribution
network

IETF's Middleboxes

- Middlebox
  - network intermediate device that implements middlebox services
  - a middlebox function requires application specific intelligence
- Examples
  - policy based packet filtering (a.k.a. firewall)
  - network address translation (NAT)
  - intrusion detection
  - load balancing
  - policy based tunneling
  - IPsec security
  - …
- RFC3303 and RFC3304
  - From traditional middleboxes
    - Embed application intelligence within the device
  - To middleboxes supporting the MIDCOM protocol
    - Externalize application intelligence into MIDCOM agents

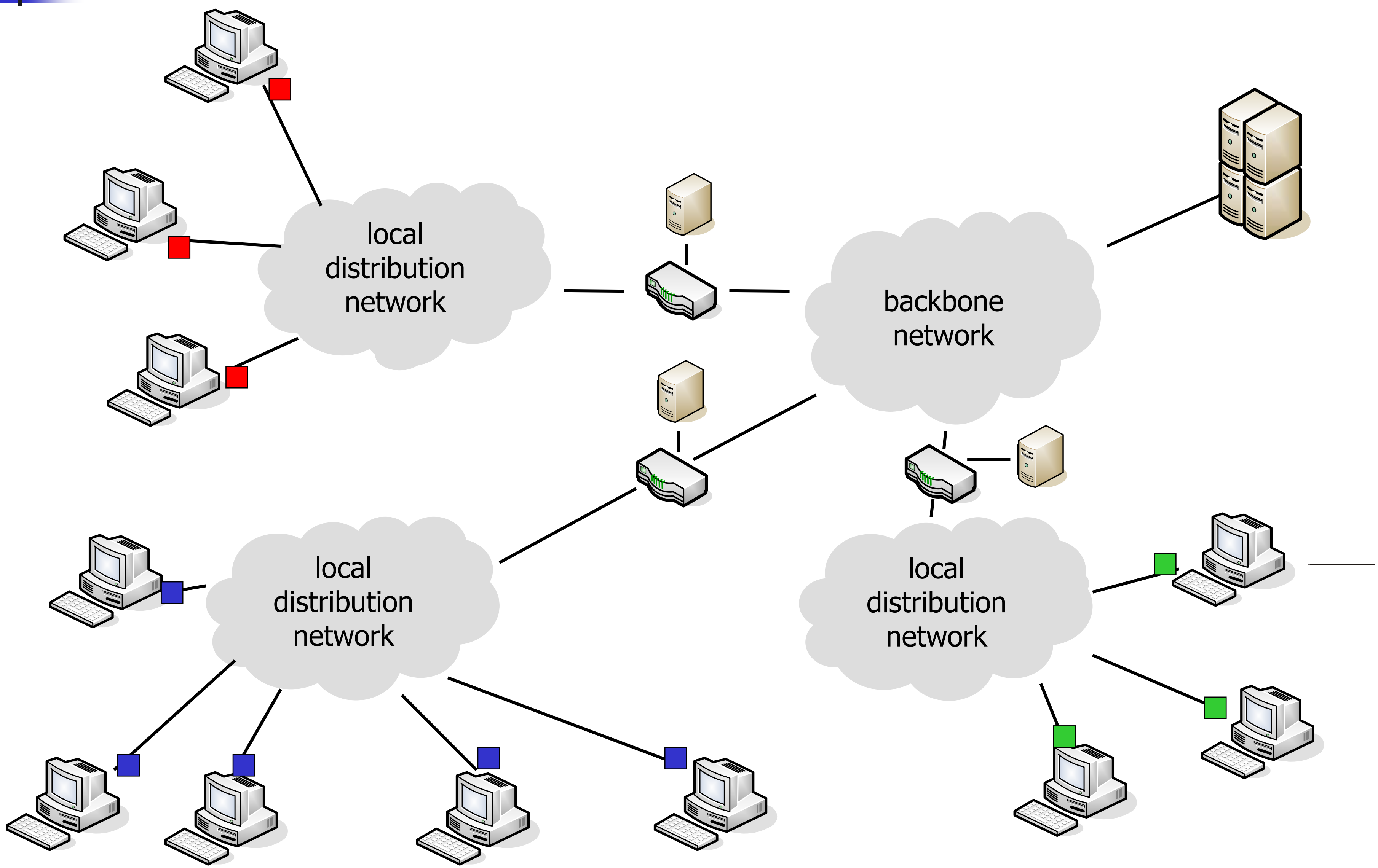
Booster Boxes

- Booster Boxes ≈ Middleboxes
  - attached directly to ISPs' access routers
  - less generic than, e.g., firewalls or NAT

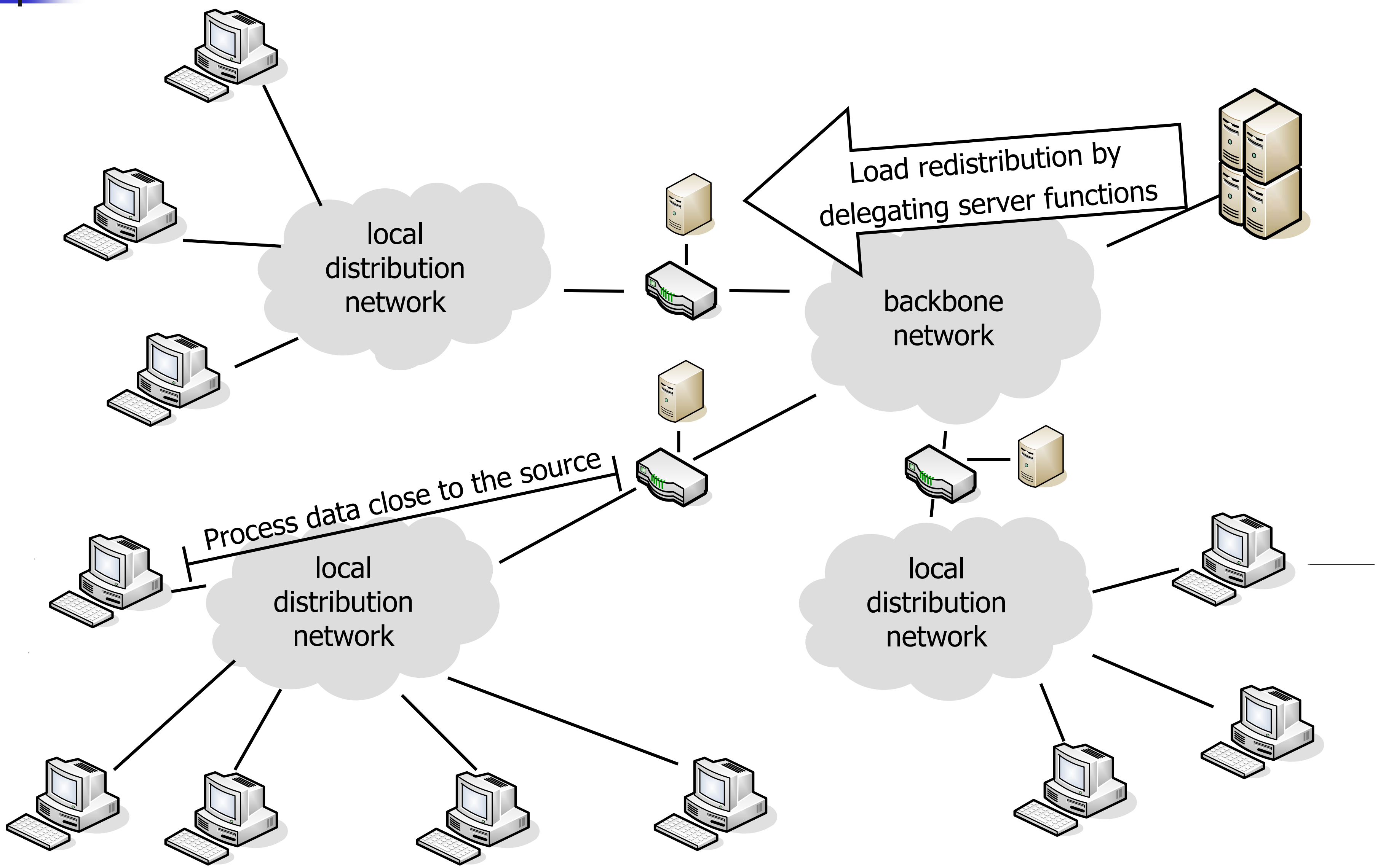
- Assist distributed event-driven applications
  - improve scalability of client-server and P2P applications

- Application-specific code: "**Boosters**"

Booster boxes



Booster boxes

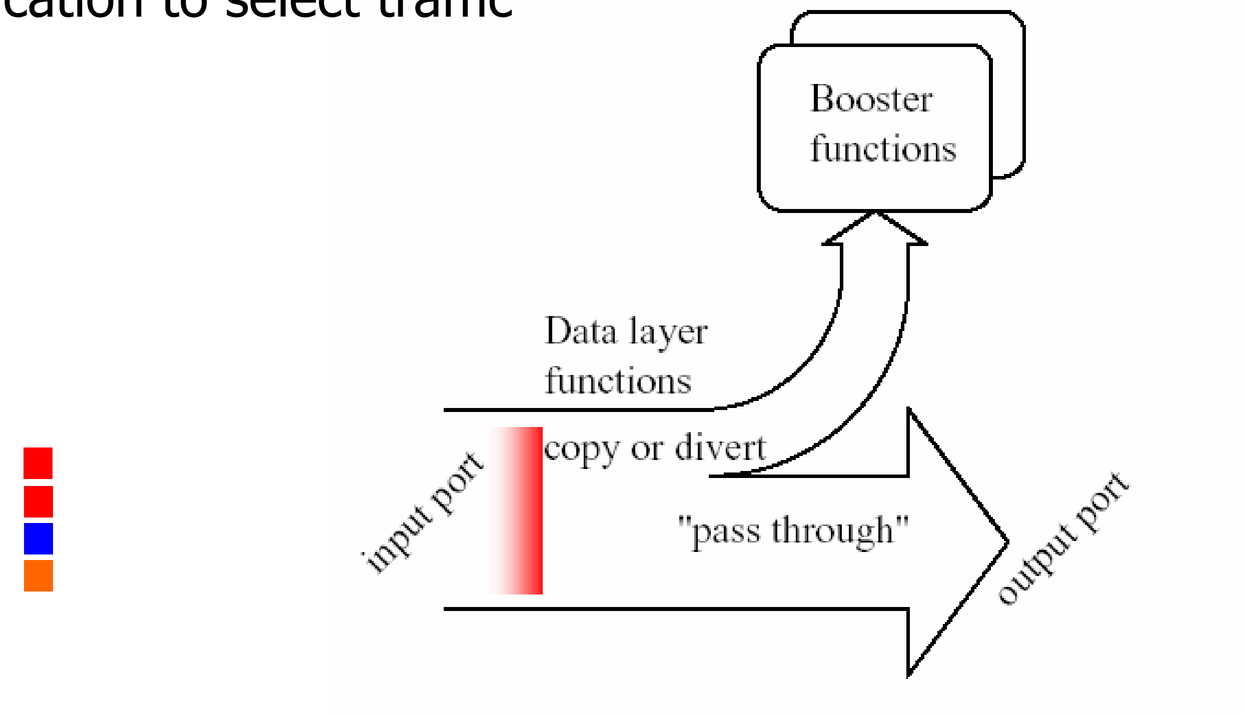


Booster Box

- Application-specific code
  - Caching on behalf of a server
    - Non-real time information is cached
    - Booster boxes answer on behalf of servers
  - Aggregation of events
    - Information from two or more clients within a time window is aggregated into one packet
  - Intelligent filtering
    - Outdated or redundant information is dropped
  - Application-level routing
    - Packets are forward based on
      - Packet content
      - Application state
      - Destination address

Architecture

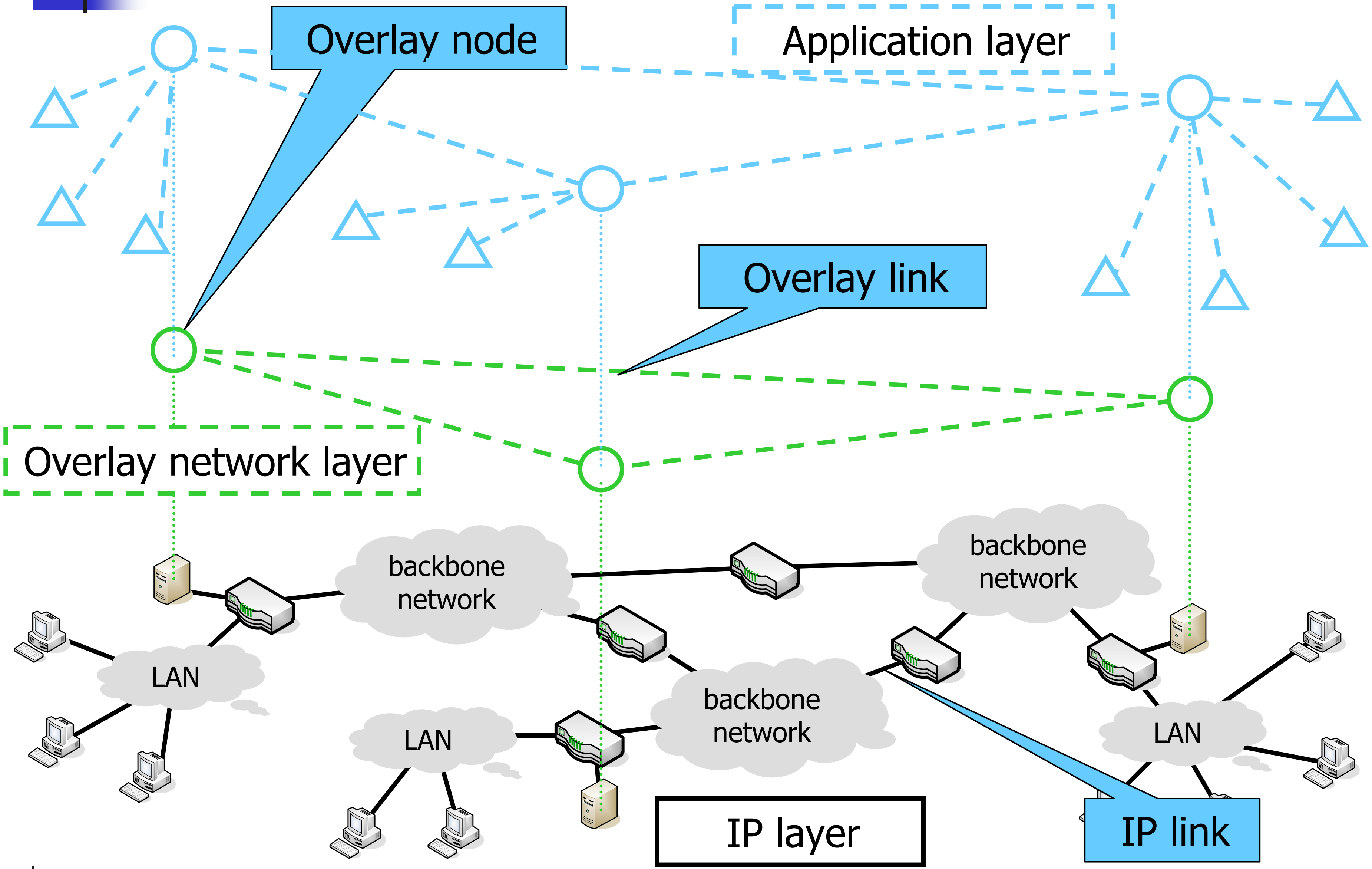
- Data Layer
  - behaves like a layer-2 switch for the bulk of the traffic
  - copies or diverts selected traffic
  - IBM's booster boxes use the packet capture library ("pcap") filter specification to select traffic



Architecture

- Booster layer
  - Booster
    - Application-specific code
    - Executed either on the host CPU or the network processor
  - Library
    - Boosters can call the data-layer operation
  - Generates a QoS-aware Overlay Network (Booster Overlay Network - BON)

Overlay networks



Architecture

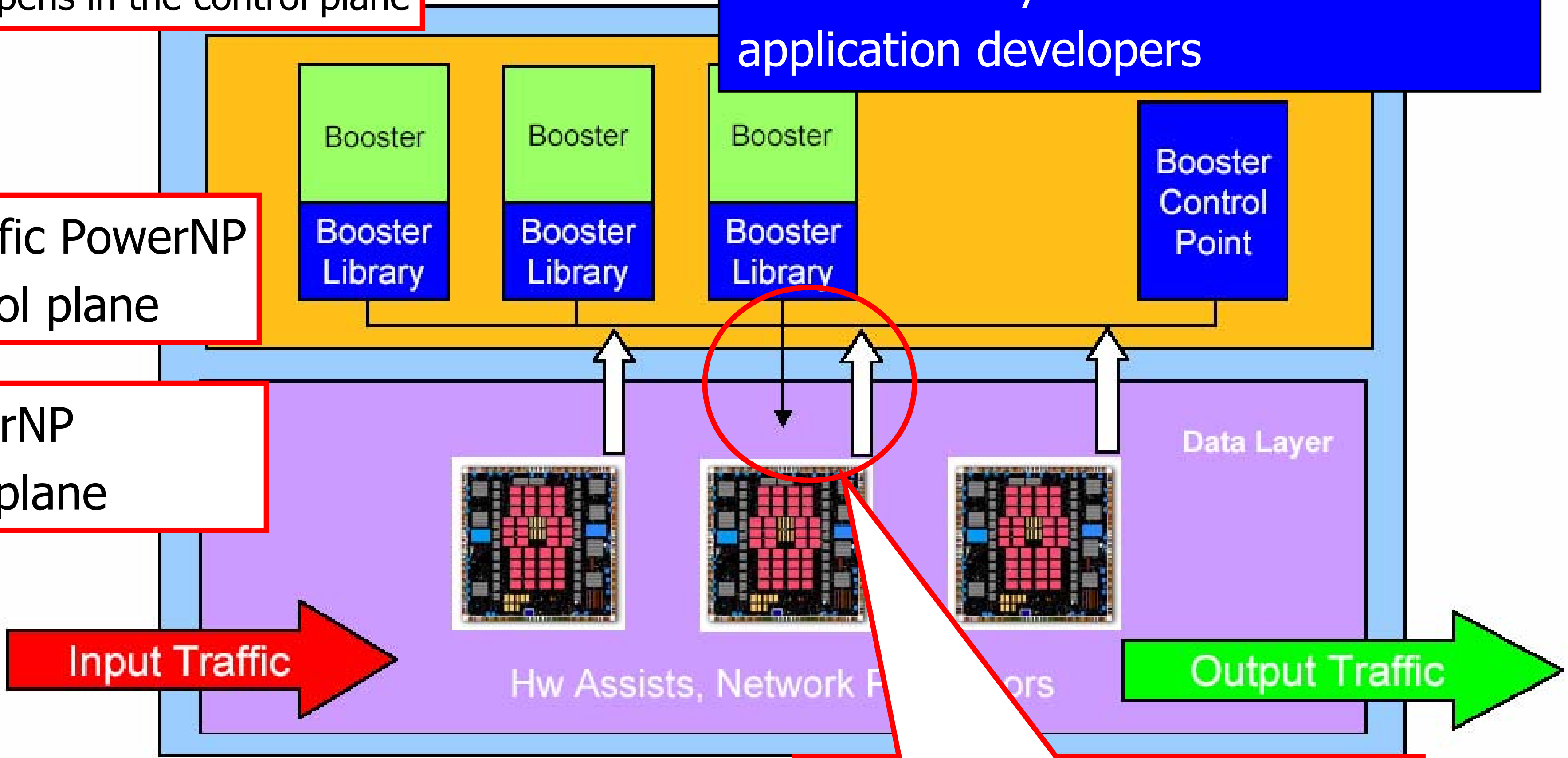
Booster: application developers' code, dynamically installed

Booster library: API's available to application developers

Even IP options processing happens in the control plane

Specific PowerNP control plane

PowerNP data plane



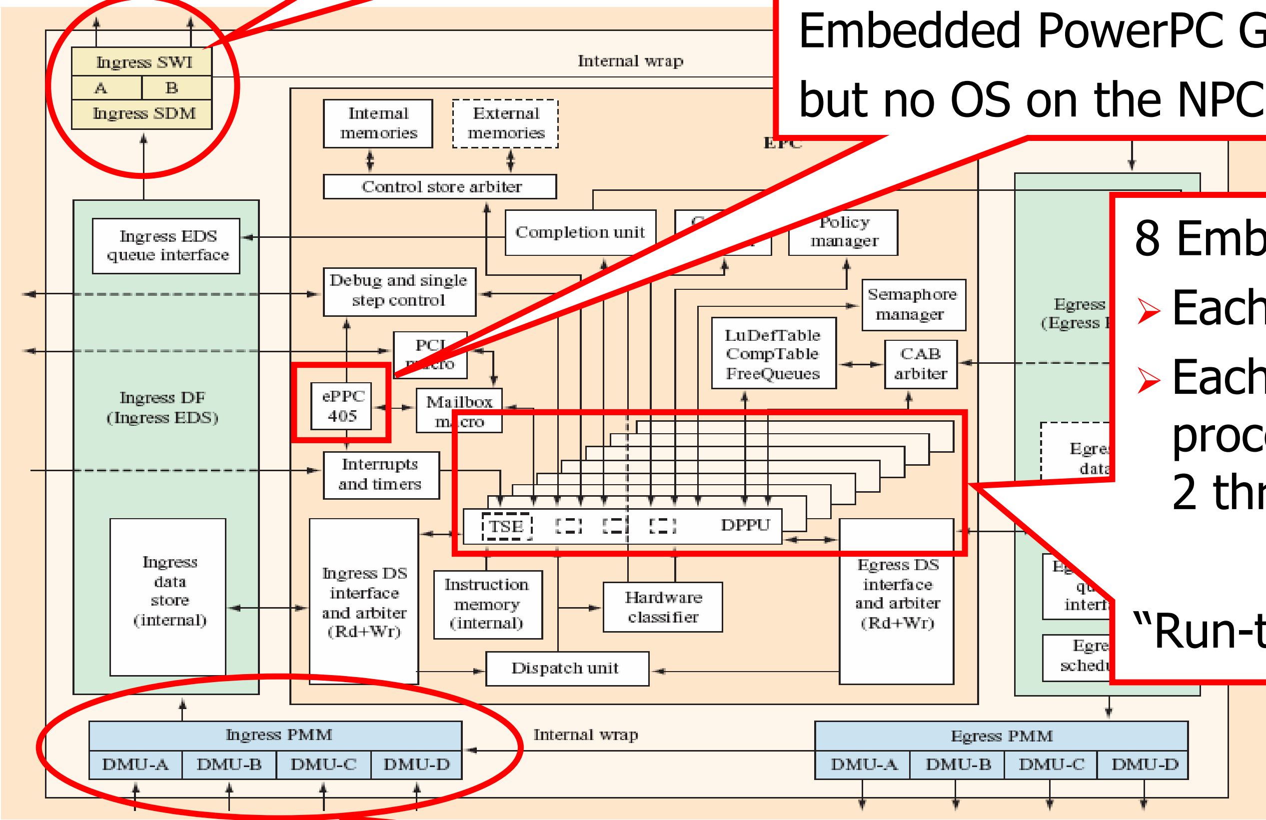
Asynchronous communication via messages

PowerNP functional block diagram

[figure from Allen et al.]

To other NPs or host computer

Embedded PowerPC GPU
but no OS on the NPC



8 Embedded processors

- Each with 4 kbytes memory
- Each with 2 core language processors, each in turn with 2 threads

"Run-to-completion"

Link layer framing: e.g. Ethernet ports

Intel IXP vs. IBM NP

- Difference between IBM NPs and IXP

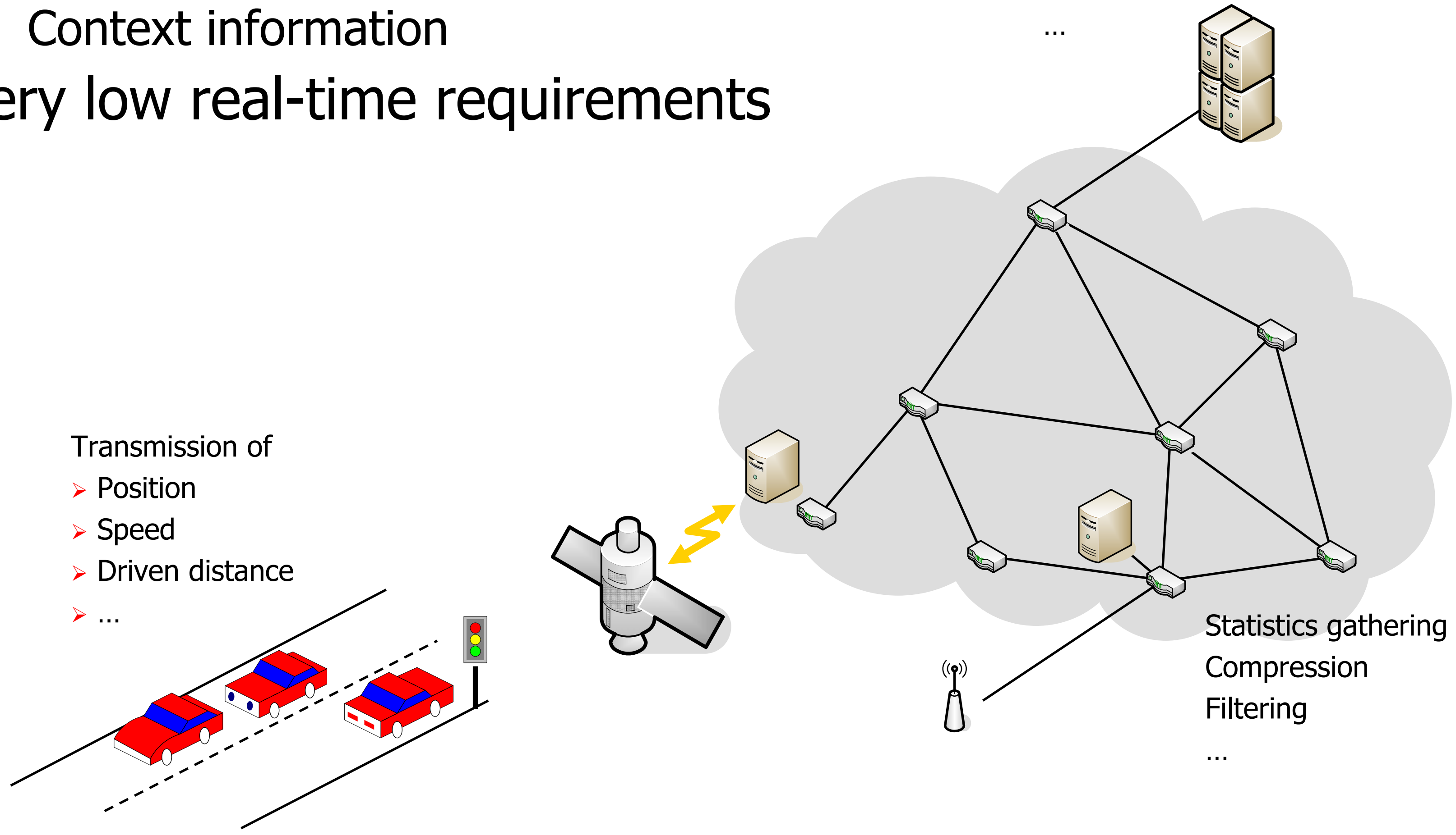
  - IXP advantage
    - General purpose processor on the card
    - Operating system on the card

  - IXP disadvantage
    - Higher memory consumption for pipelining
    - Larger overhead for communication with host machine

Data Aggregation Example: Floating Car Data

Main booster task:

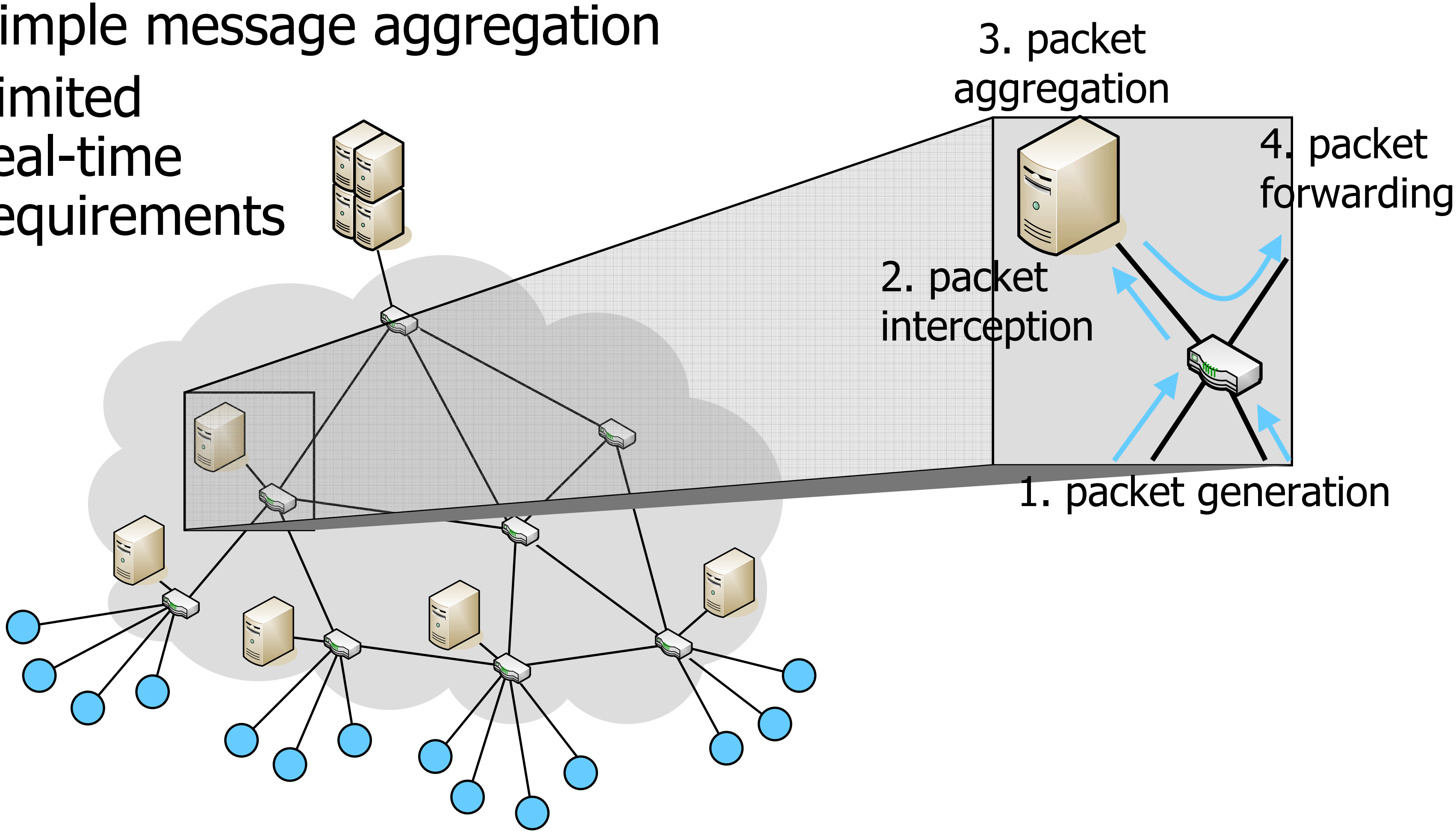
- Complex message aggregation
  - Statistical computations
  - Context information
- Very low real-time requirements



Interactive TV Game Show

Main booster task:

- Simple message aggregation
- Limited real-time requirements

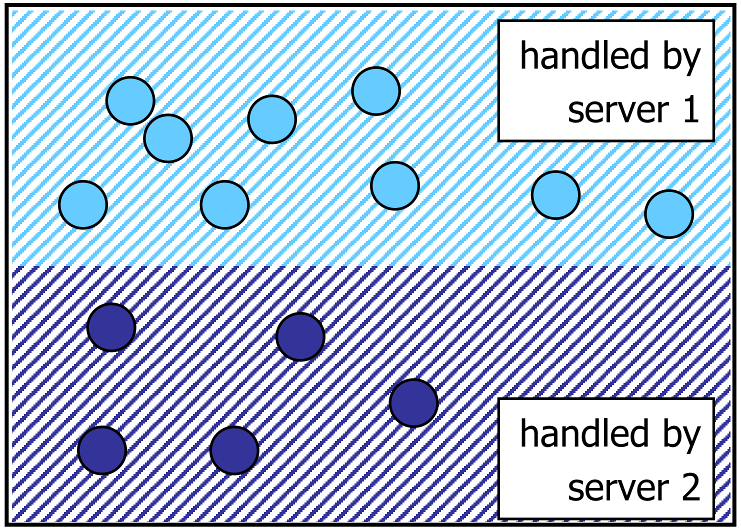


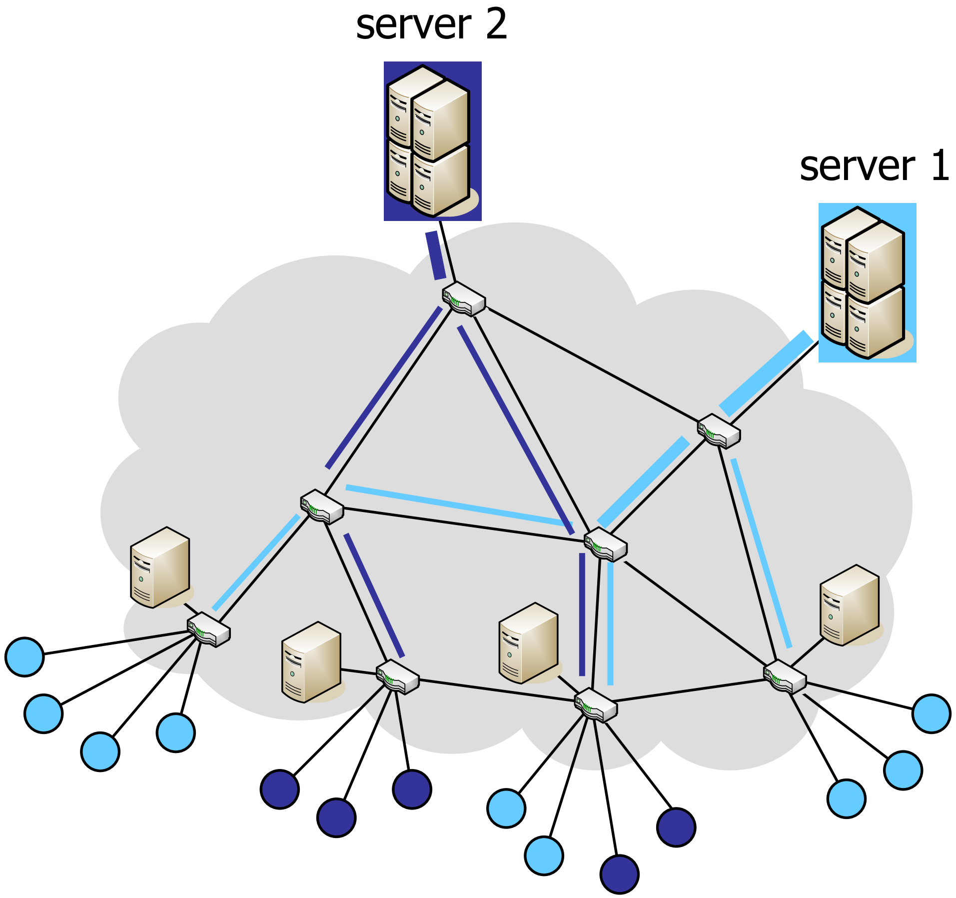
Game with large virtual space

Main booster task:

- Dynamic server selection
  - based on current in-game location
  - Require application-specific processing

Virtual space





- High real-time requirements

Summary

- Scalability
  - by application-specific knowledge
  - by network awareness

- Main mechanisms
  - Caching on behalf of a server
  - Aggregation of events
  - Attenuation
  - Intelligent filtering
  - Application-level routing

- Application of mechanism depends on
  - Workload
  - Real-time requirements

Some References

1. Tatsuya Yamada, Naoki Wakamiya, Masayuki Murata, and Hideo Miyahara: "Implementation and Evaluation of Video-Quality Adjustment for heterogeneous Video Multicast", 8th Asia-Pacific Conference on Communications, Bandung, September 2002, pp. 454-457

2. Marc E. Fiuczynski, Richard P. Martin, Tsutomu Owa, Brian N. Bershad: "On Using Intelligent Network Interface Cards to support Multimedia Applications", The 8th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV), Cambridge, UK, 1998
3. Marc E. Fiuczynski, Richard P. Martin, Brian N. Bershad, David E. Culler: "SPINE: An Operating System for Intelligent Network Adapters", Technical Report UW-CSE-98-08-01, August 1998

4. Daniel Bauer, Sean Rooney, Paolo Scotton, "Network Infrastructure for Massively Distributed Games", NetGames, Braunschweig, Germany, April 2002
5. J.R. Allen, Jr., et al., "IBM PowerNP network processor: hardware, software, and applications", IBM Journal of Research and Development, 47(2/3), pp. 177-193, March/May 2003