

# Infrastructural proxy-based adaptation for network and client variation

Armando Fox, Steven D. Gribble,  
Yatin Chawathe, Eric A. Brewer,  
Elan Amir

**Presenter: Ranjita Bhagwan**

(tables and figures from

[www.cs.berkeley.edu/~adj/cs294-1.s98/  
notes12b/TACC-Combined.ppt](http://www.cs.berkeley.edu/~adj/cs294-1.s98/notes12b/TACC-Combined.ppt))

# Motivation

- ◆ Lots of variation at the clients

<i>Property</i>	<i>Desktop + Fast LAN</i>	<i>PDA + wide-area wireless</i>
Bandwidth	<b>10-100 Mb/s</b>	<b>4800-19.2 Kb/s</b>
Display	<b>1280x1024x16</b>	<b>320x240x2</b>
CPU	<b>133 MHz Pentium</b>	<b>20 MHz ARM</b>
Memory	<b>10's of MB</b>	<b>2-4 MB</b>

- ◆ Software variations, too.
- ◆ Servers will find it difficult to handle this variation.

# Goals

- ◆ Provide meaningful representation of data for a range of clients
- ◆ Reduce end-to-end latency when the network is slow

# Design Principles

- ◆ Datatype-specific lossy compression mechanisms
  - distillation
  - refinement
- ◆ “On-the-fly” adaptation, as opposed to pre-computed representations
- ◆ Intermediate proxy-based adaptation
  - technical and economic reasons to do this.

# Datatype-specific distillation

- ◆ Application-specific distillation preserves semantic content



6.8x



65x



## 1.2 The Remote Queue Model

We introduce *Remote Queues* (RQ), which provides a general abstraction for low-level systems consisting of three basic elements. First, one or more sending nodes. Second, one or more receiving nodes. Third, a set of *enqueue* operations.

`enqueue(n, q, arg0, ..., argn, sbu, f, ...)`

## 1.2 The Remote Queue Model

We introduce *Remote Queues* (RQ), ....

# Datatype-specific refinement

- ◆ Get specific parts of the object of interest



**Distilled image  
(by 60X)**



**Zoom in to original  
resolution**

# On-demand adaptation

- ◆ Adaptation tuned specifically to each client.
  - Maximum benefit from distillation/refinement.
  - More work to be done by the adaptation module.
- ◆ Precomputation is bad
  - Specific websites for PDAs should just be a temporary solution
  - Extra work for servers, as client variability increases.
- ◆ **Hybrid approach**
  - Precompute different resolutions of text/images/video
  - Switch from one to the other depending on b/w variations
    - ◆ Realplayer streaming
  - Not as fine-grained

# Proxy-based adaptation – technical issues

- ◆ Servers provide only high-quality content
  - ◆ Legacy servers remain unchanged
    - Proxies will have to be continuously upgraded.
  - ◆ Client communicates only with proxy
    - Extra load on the proxy.
- Put all adaptation functionality into the proxy**

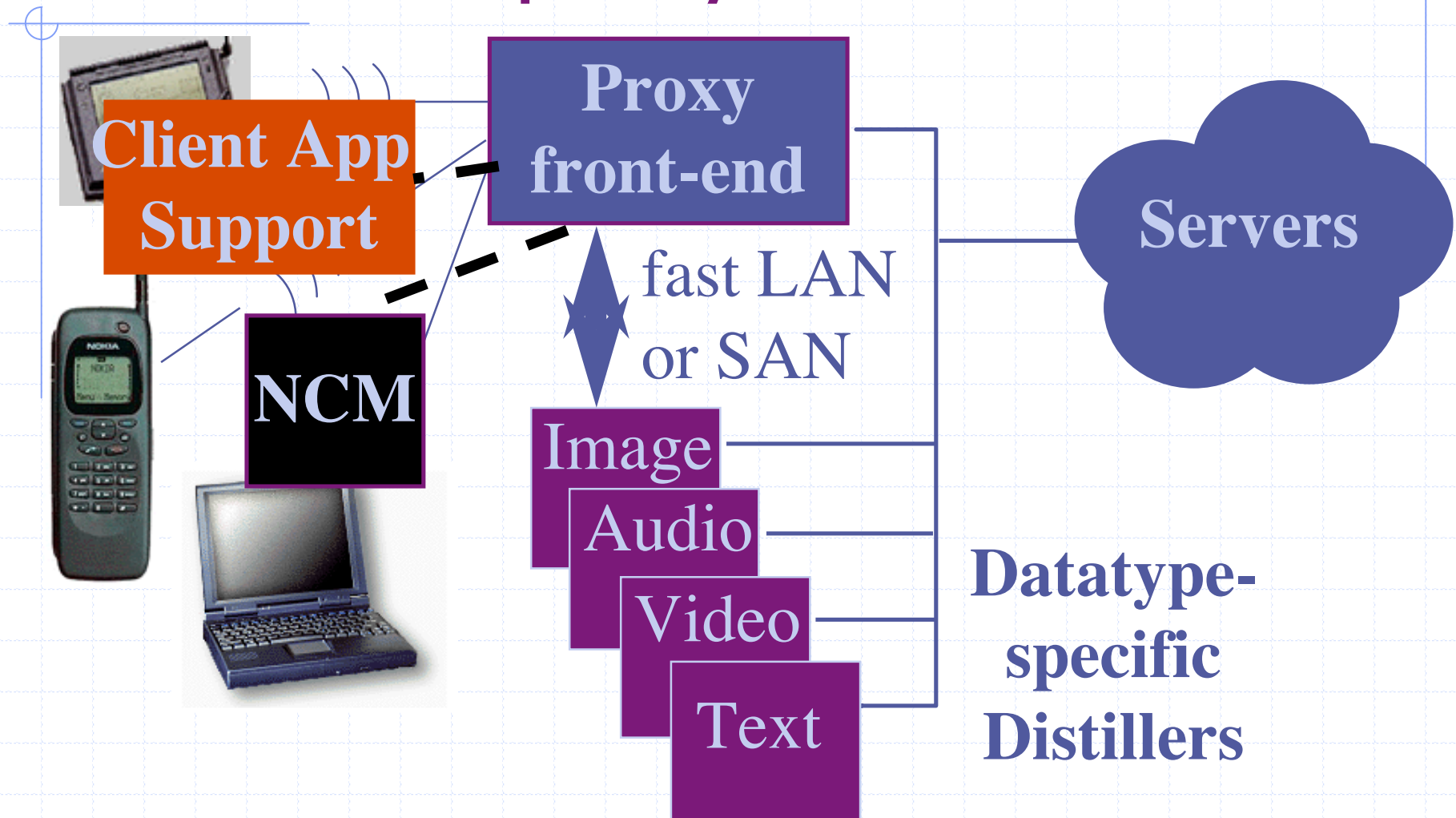
# Proxy-based adaptation – technical issues

- ◆ Incremental deployment to support various clients' software+hardware
- ◆ Economy of scale: Case for Network computers
  - not yet happened

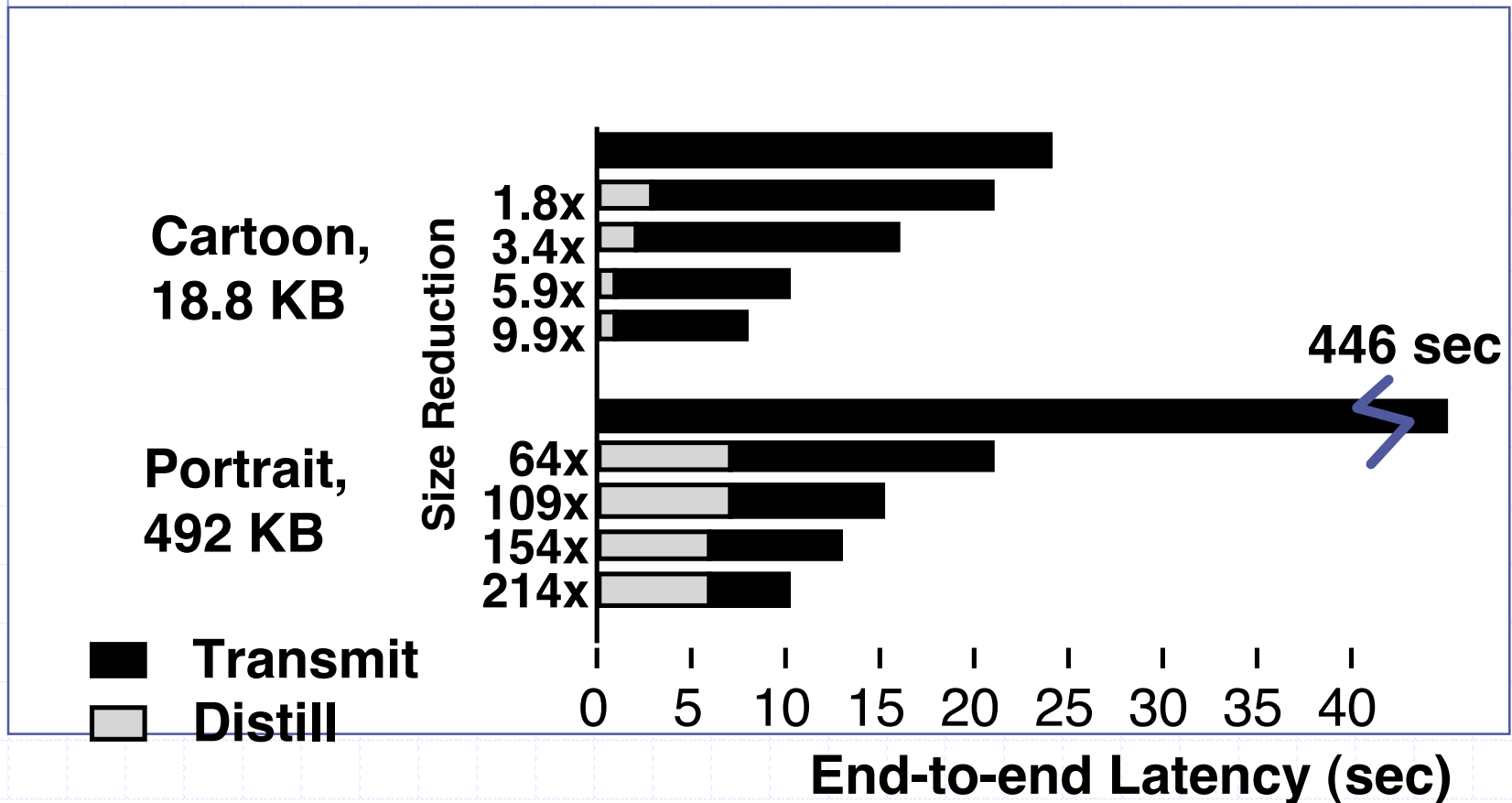
# Proxy-based adaptation – Economic issues

- First and foremost, it is in the content provider's interest to provide tailored content. ISP is secondary.
  - ◆ Akamai: 244,000 hits
  - ◆ Intel Quickweb: 353 hits
  - ◆ Proxinet: Couldn't find
- For wireless devices, ISPs could be more interested.
- Content provider may give some guarantees that the proxy needs to honour
  - ◆ Need interaction between CP and home ISP of client – Lots of work!
  - ◆ Proprietary formats – proxy would need to support all of them.
    - Realplayer, Quicktime, Windows MediaPlayer
- Who should pay for this? The content provider, or the client/consumer?

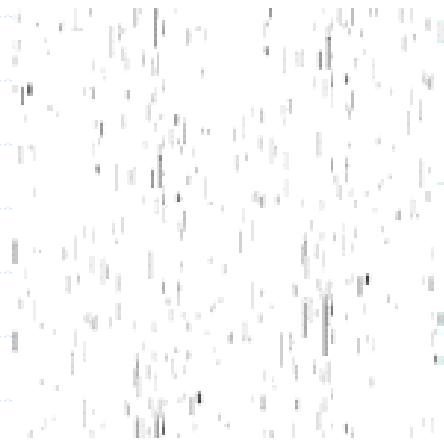
# Distillation proxy architecture



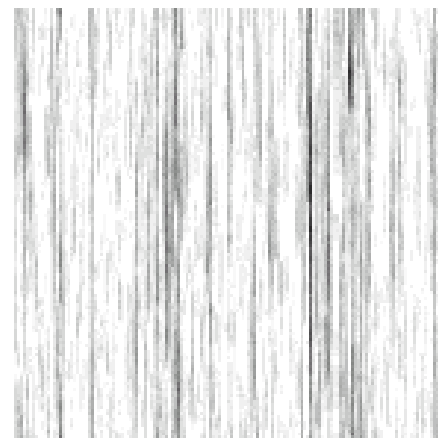
# End-to-end latency for images



# Scalability



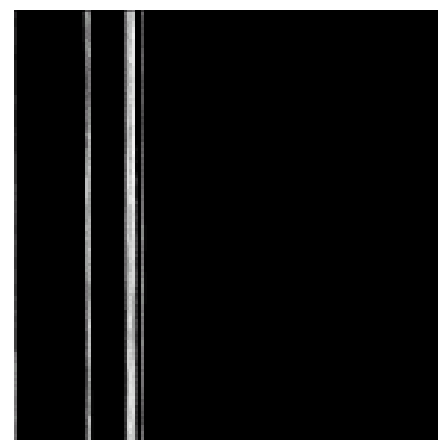
(a) 1 user



(b) 16 users



(c) 20 users



(d) 24 users

# TACC Servers

## ◆ TACC Service

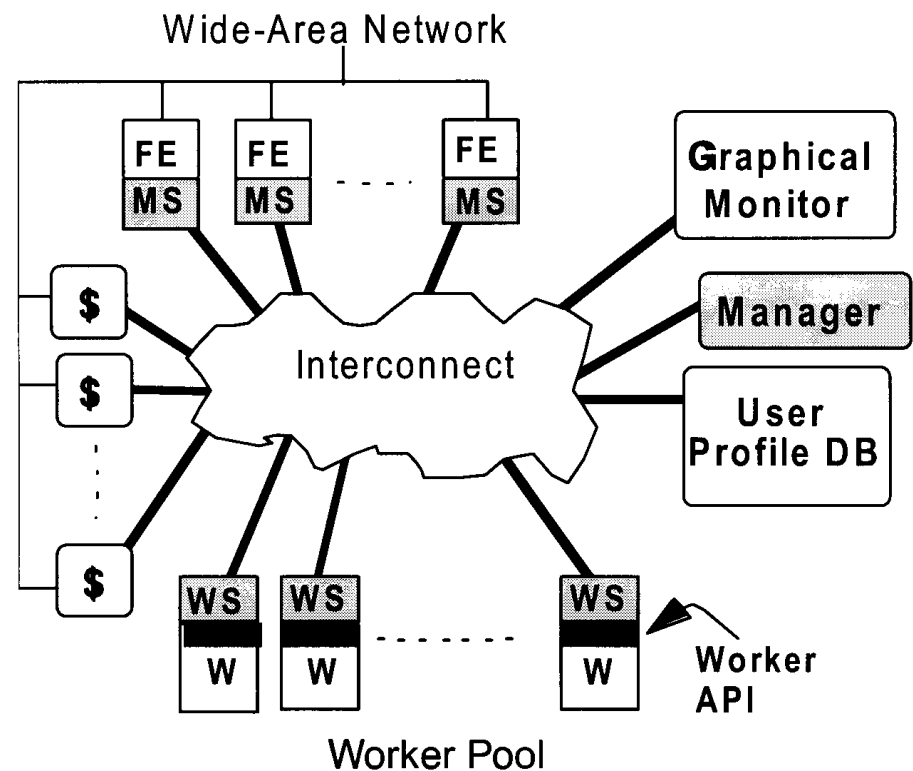
- *Transformation, Aggregation, and Caching* of existing content
- *Customization* tunes behaviour to each user

## ◆ Scalable TACC architecture meets the challenges of infrastructural services:

- Scalability and high availability
- Low incremental operating cost

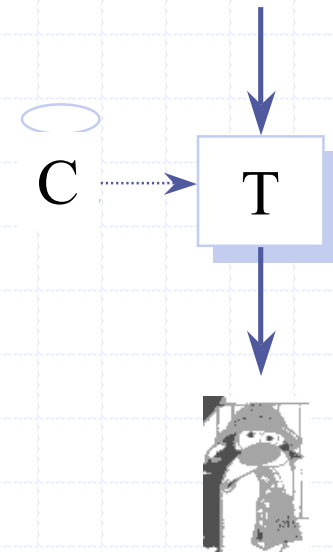
# TACC server architecture

- ◆ Frontends shepherd incoming requests, match them up with user database
- ◆ Workers (caches and service-specific modules) do the TAC part
- ◆ Customization DB stores user profiles
- ◆ Manager autostarts workers when needed
- ◆ Monitor supports error notification and visualization

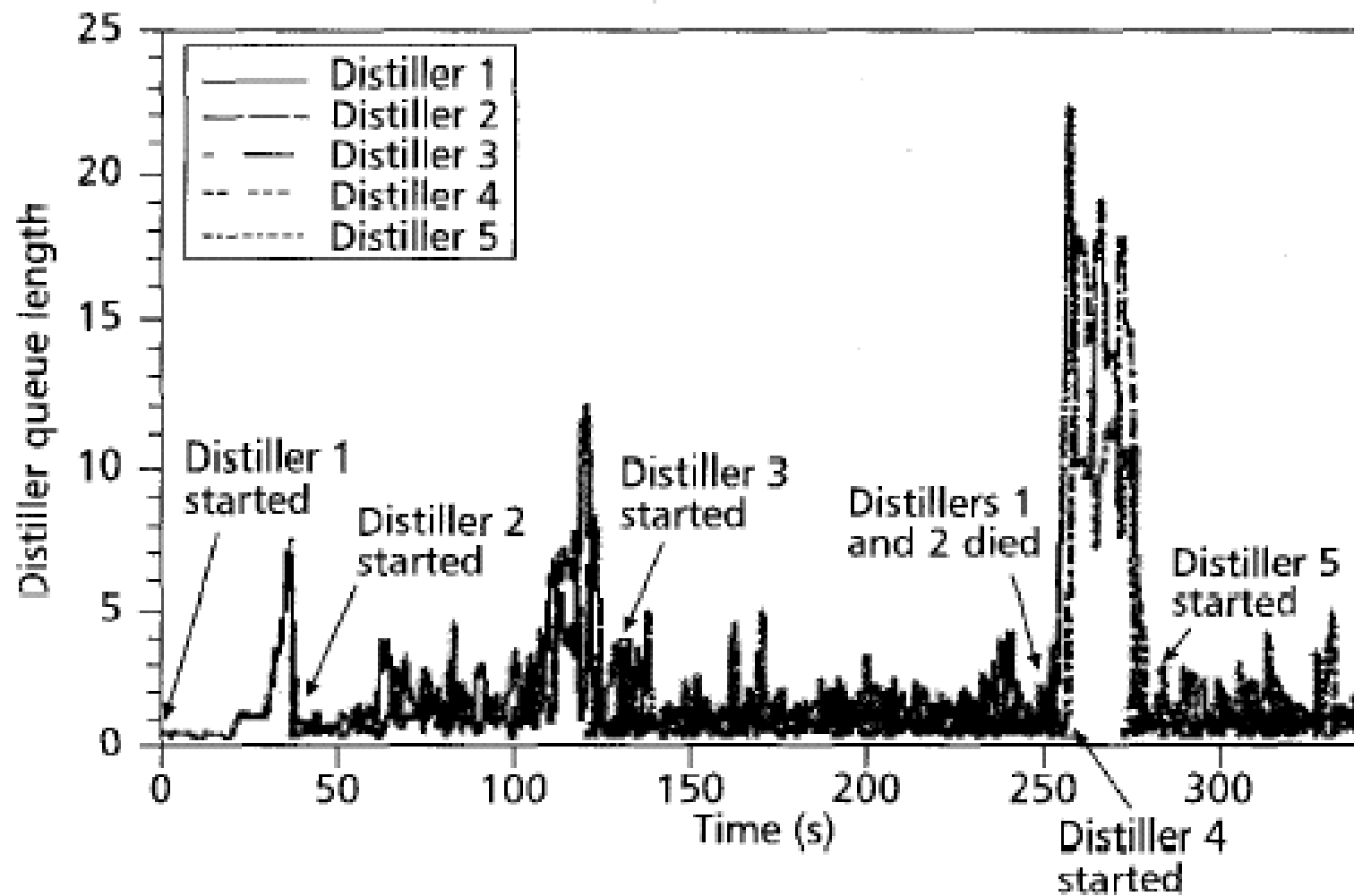


# TranSend: Cluster based TACC server

- ◆ It performs on the fly web image compression
- ◆ Each TranSend worker handles compression or markup for a specific MIME type
  - objects of unsupported types are passed through to the user unaltered (e.g., ps-rich text)
- ◆ TranSend can speed up web browsing by a factor of "3-7"



# Self-tuning and load balancing

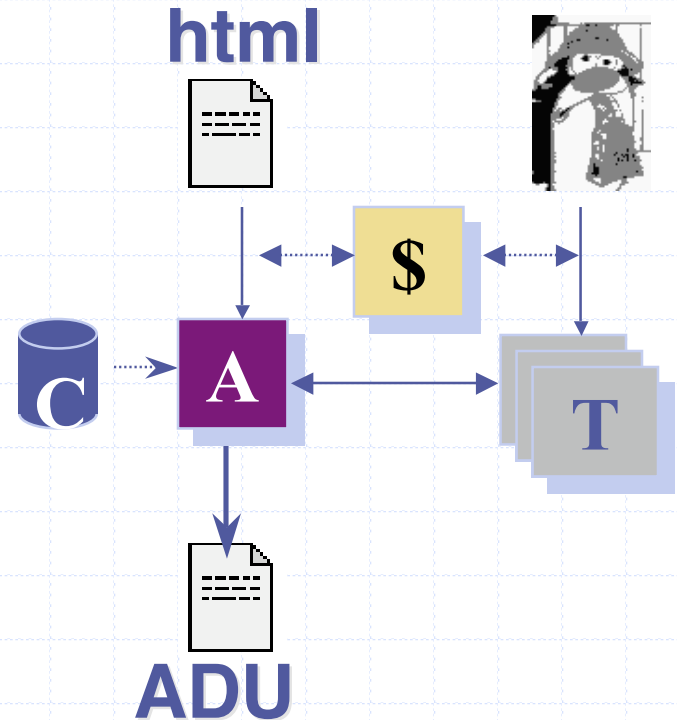


# Scalability

Requests/s	# FEs	# workers	Element that saturated
0-24	1	1	Workers
25-47	1	2	Workers
48-72	1	3	Workers
73-87	1	4	FE Ethernet
88-91	2	4	Workers
92-112	2	5	Workers
113-135	2	6	Workers + FE Ethernet
136-159	3	7	Workers

# Other applications on TACC

- ◆ Top Gun Wingman
  - Browser for the Palm
- ◆ Top Gun Mediaboard
  - Distributed whiteboard application.



# Summary

- ◆ Three design principles proposed
  - Datatype-specific distillation
  - on-demand adaptation
  - infrastructure-based adaptation
- ◆ Scalable and self-tuning architecture described
- ◆ Applications have been built
- ◆ Companies have built products based on this
- ◆ Thorough treatment of text/image adaptation, not yet of audio/video adaptation