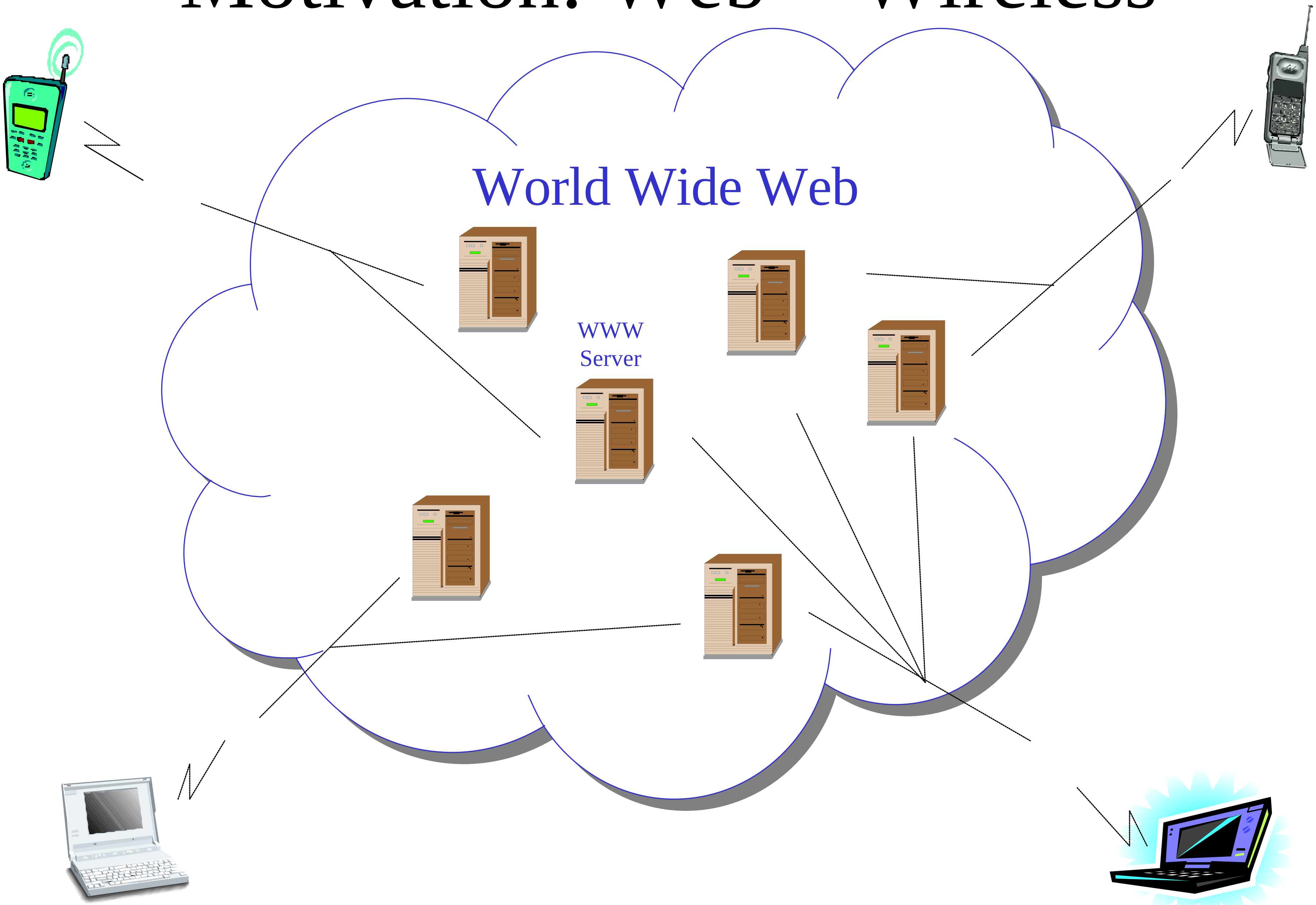
A Flexible Architecture for Customizing Web Streams for Wireless Clients

Jesse Steinberg and Joseph Pasquale

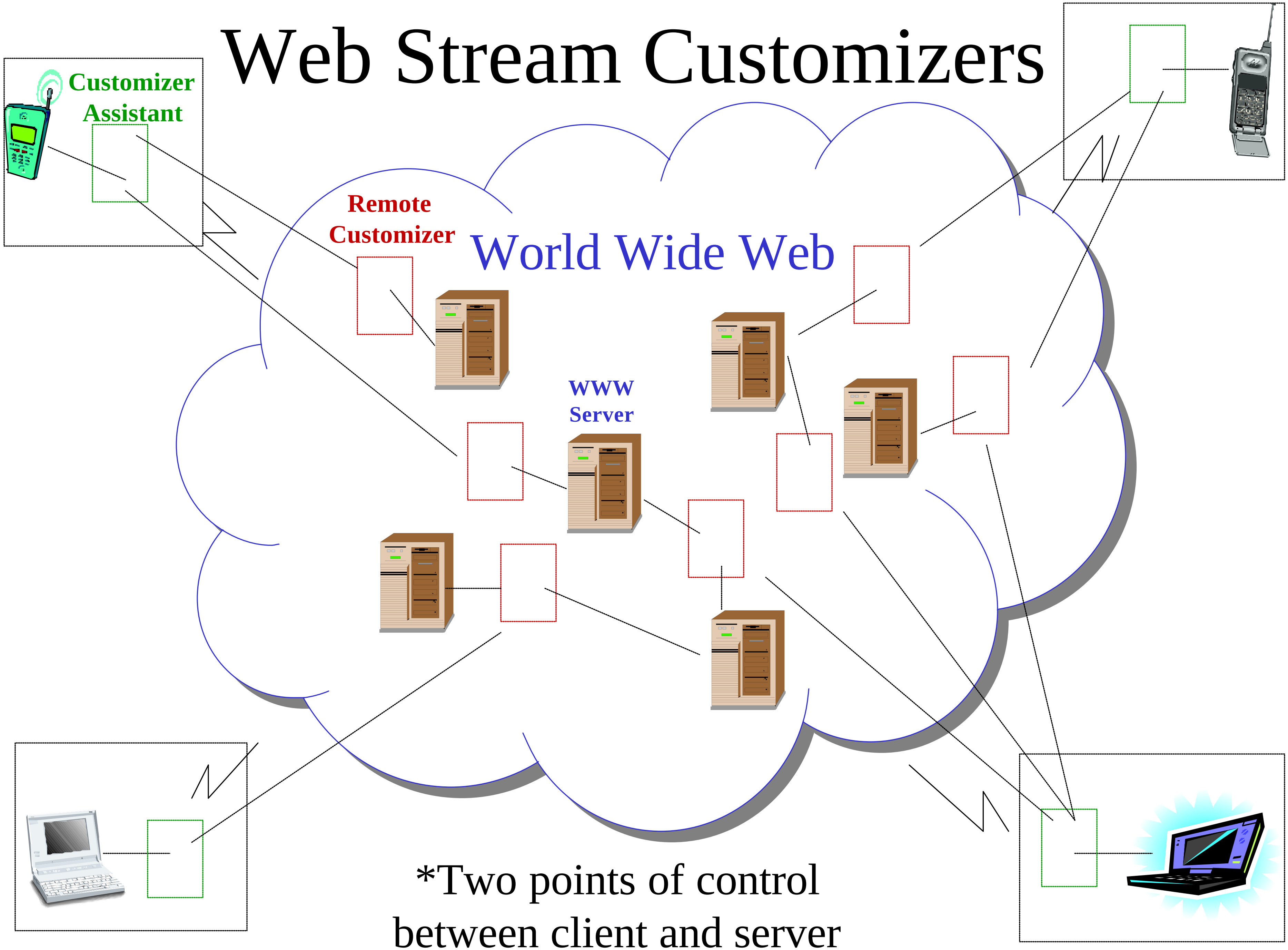
University of California, San Diego

Department of Computer Science and Engineering

Motivation: Web + Wireless

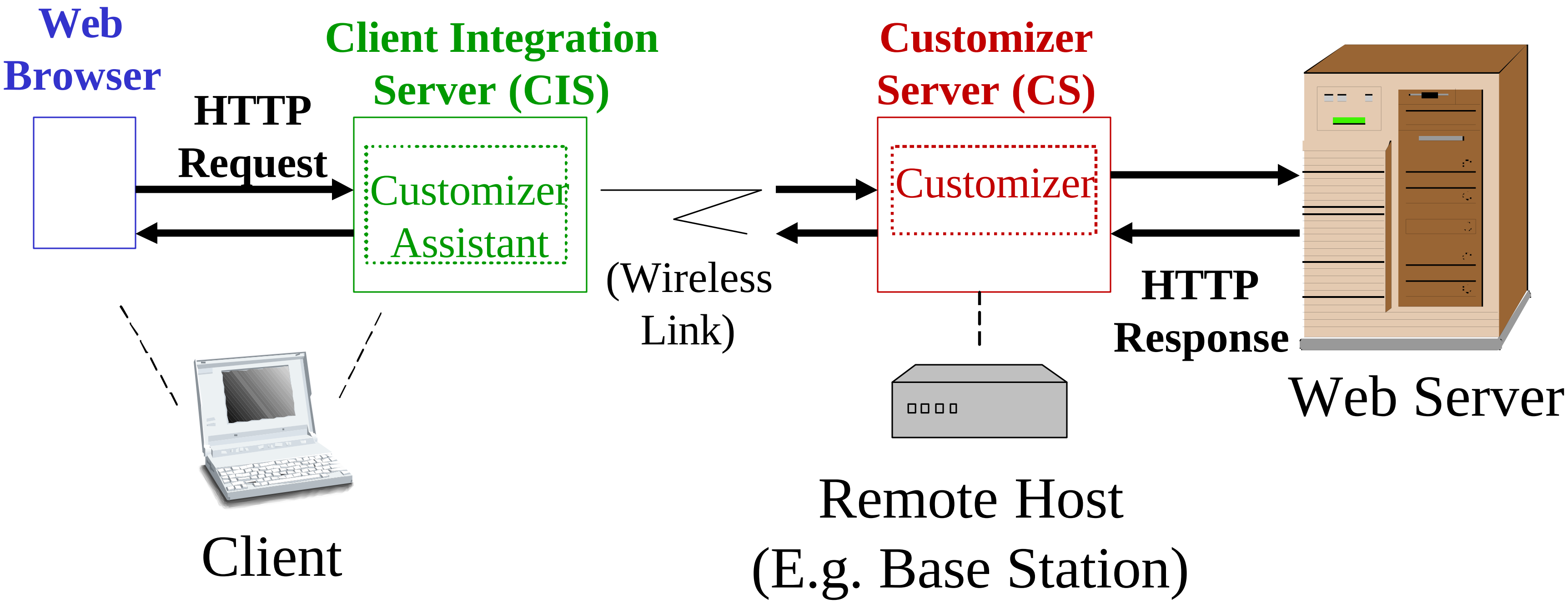


Web Stream Customizers



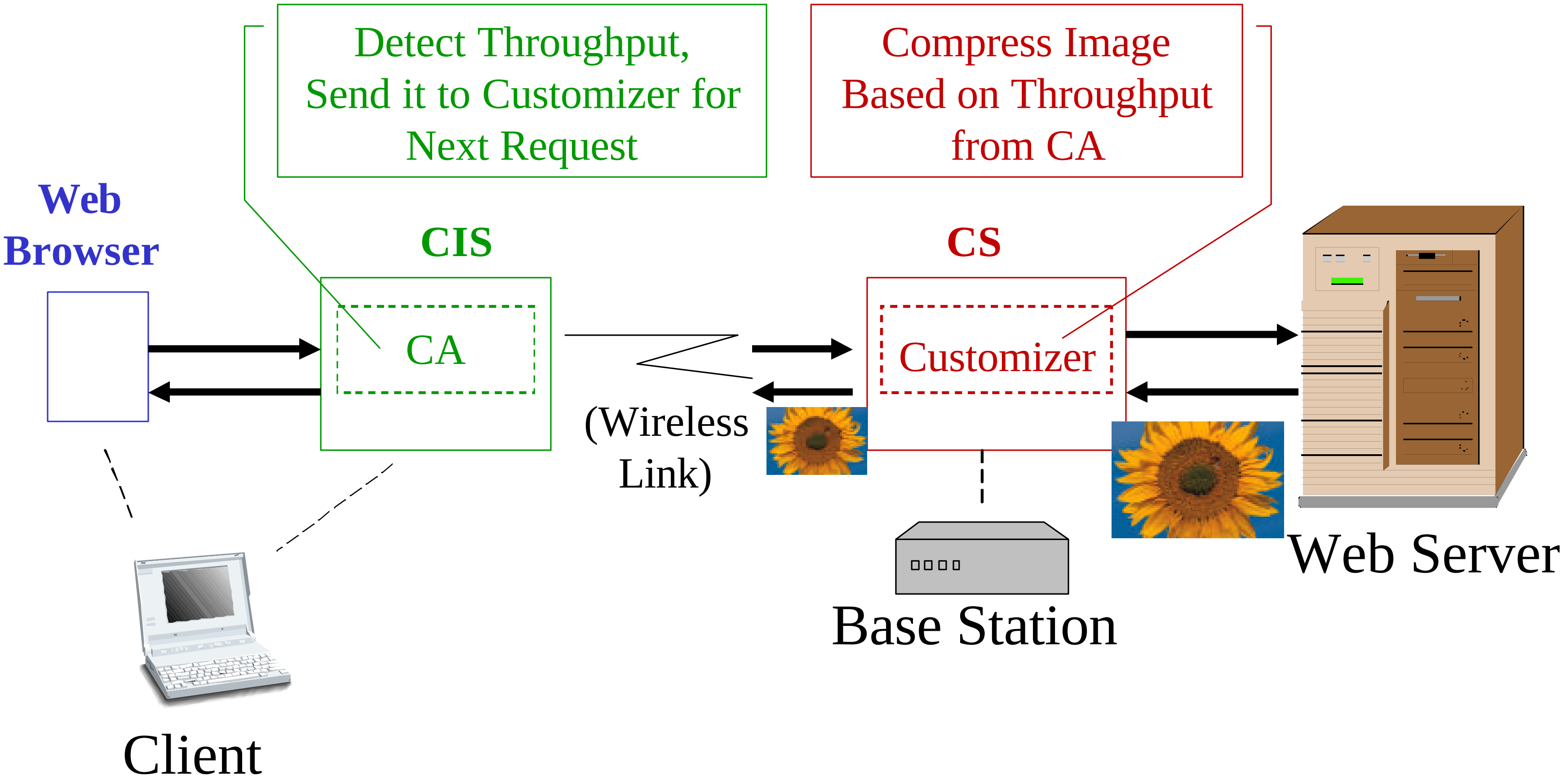
*Two points of control
between client and server

Communication Path



*Customizer and Customizer Assistant can be dynamically deployed

Example: Adaptive Image Filter



Detect Throughput, Send it to Customizer for Next Request

Compress Image Based on Throughput from CA

Web Browser

CIS

CA

CS

Customizer

(Wireless Link)

Web Server

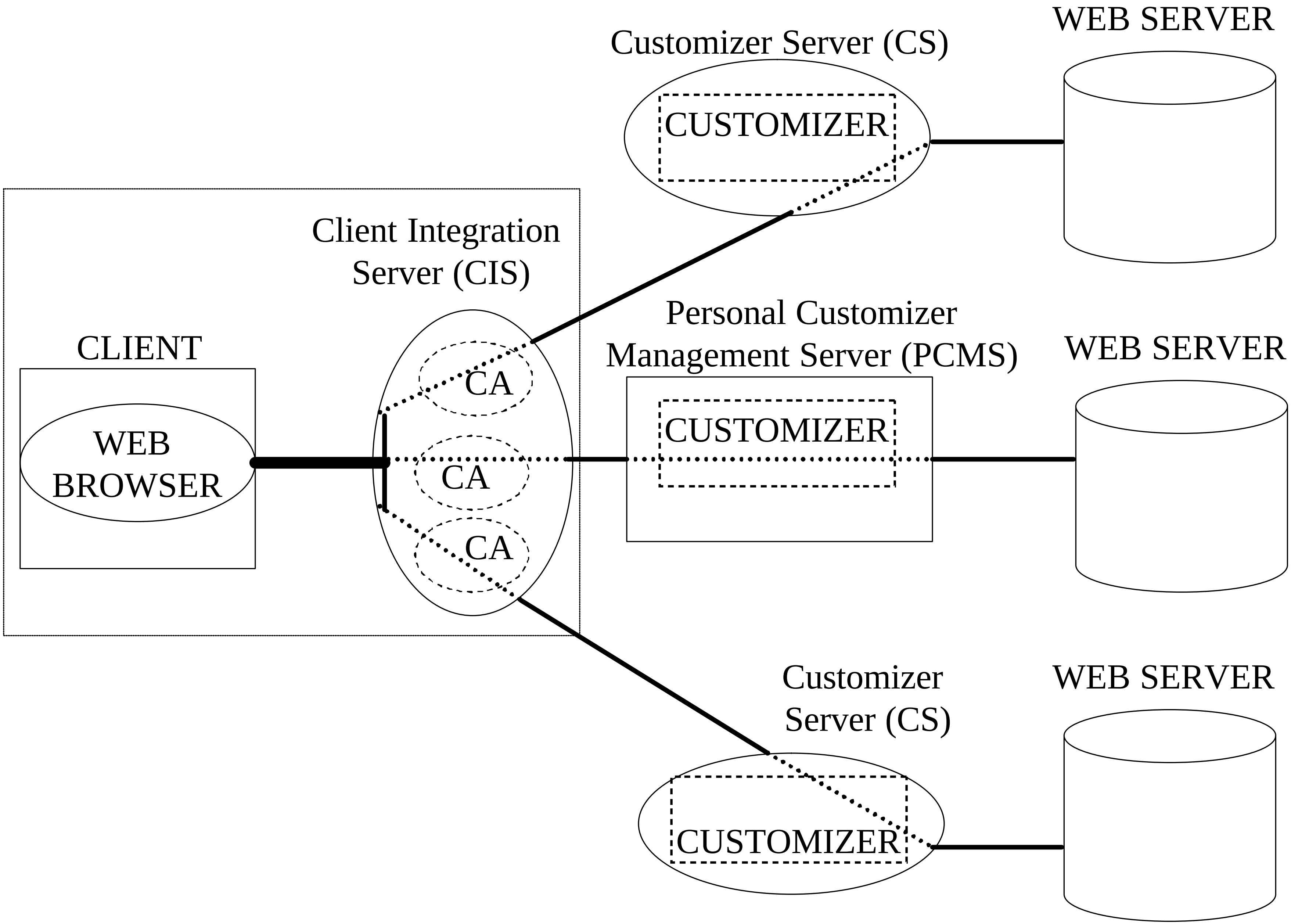
Base Station

Client

Where Do Customizers Run?

- Can be a third party server
  - Flexibility of location
- A personal server can be used
  - Personal Customizer Management Server (PCMS)
  - Take advantage of availability of user owned host or account
  - Can use resources such as persistent storage

Multiple Active Customizers

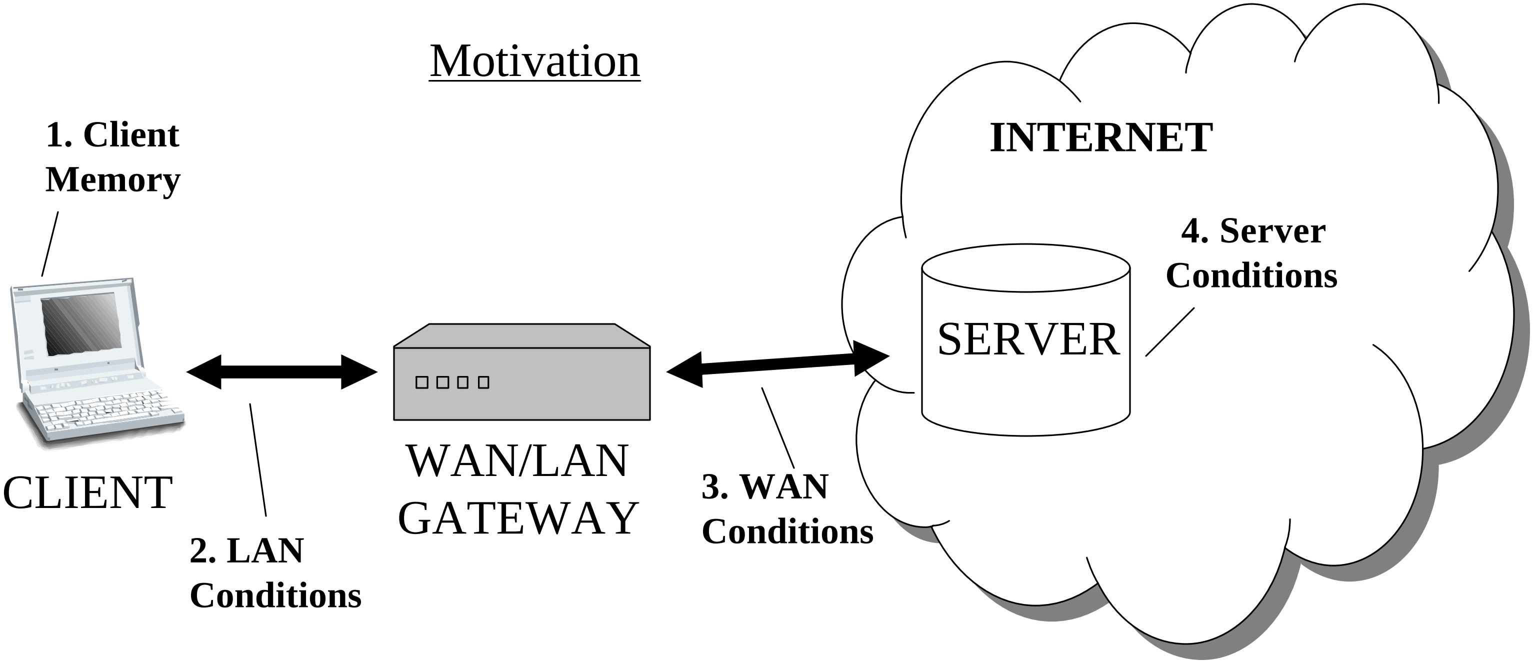


Customizer Applications

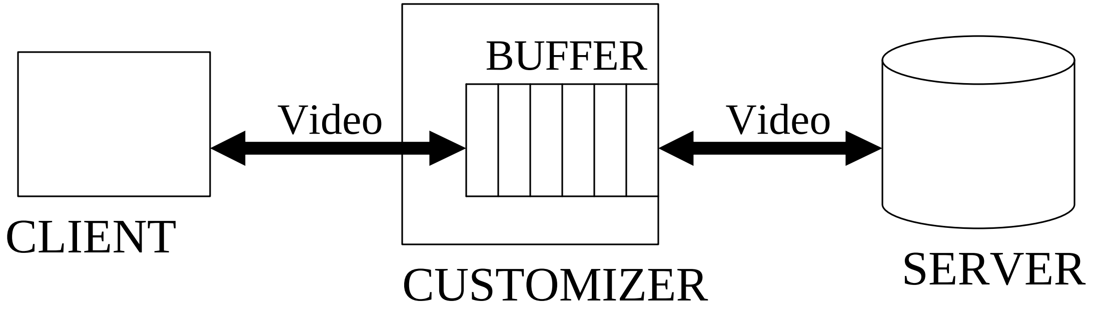
- Adaptive Compression
  - Text, image filtering
- Transaction Reliability
  - Mask failures, store results at CS
- Selective Encryption
- Network Flow Buffering
  - Buffer and regulate streaming traffic
  - E.g. Streaming multimedia

Network Flow Buffer: Closer Look

Motivation

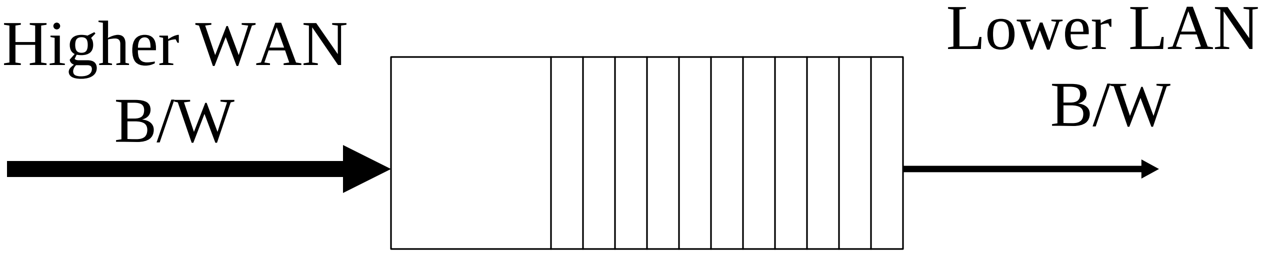


NFB Customizer



NFB Smoothing

Case 1:

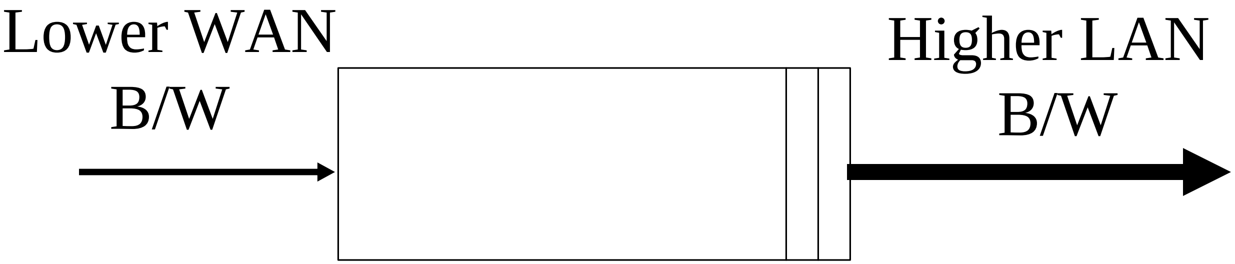


Higher WAN B/W

Lower LAN B/W

Buffer Fills Over Time

Case 2:



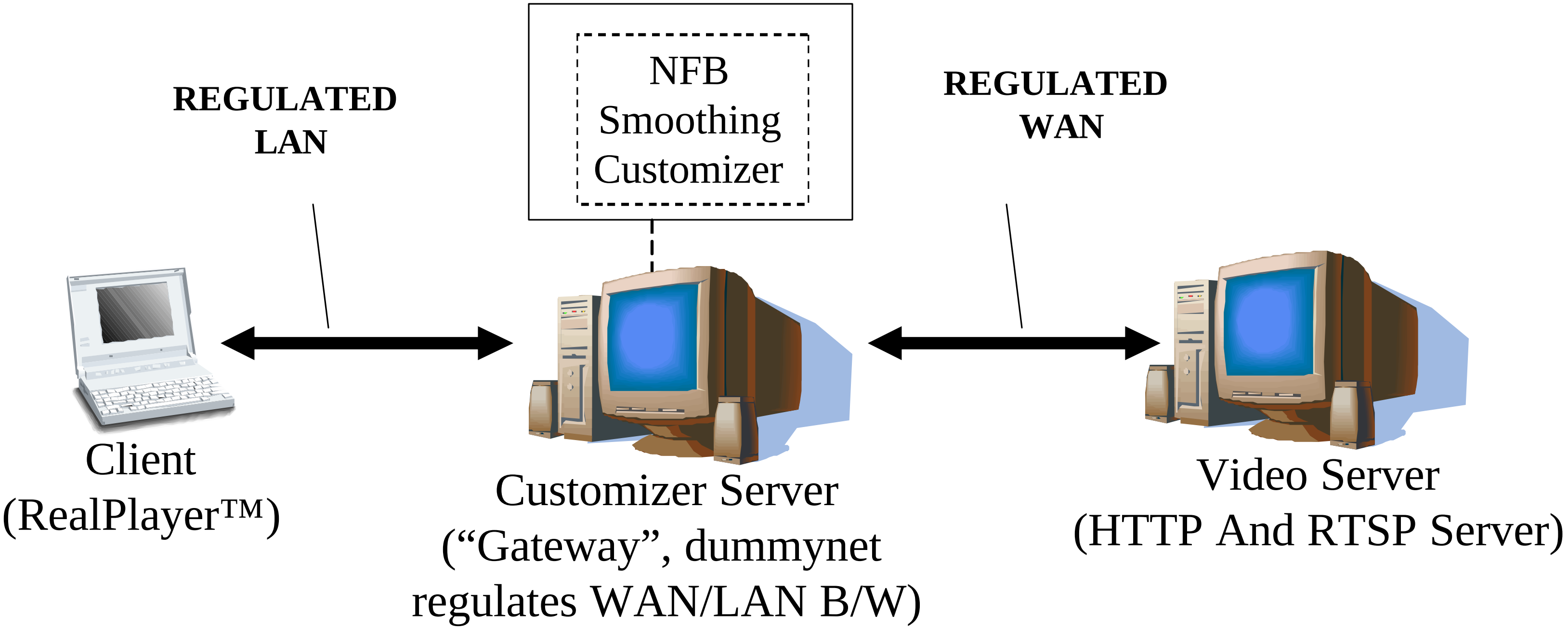
Lower WAN B/W

Higher LAN B/W

Buffer Drains Over Time

* Goal: Maintain Smooth, Uninterrupted Video Playback

Smoothing Performance Evaluation



REGULATED LAN

NFB Smoothing Customizer

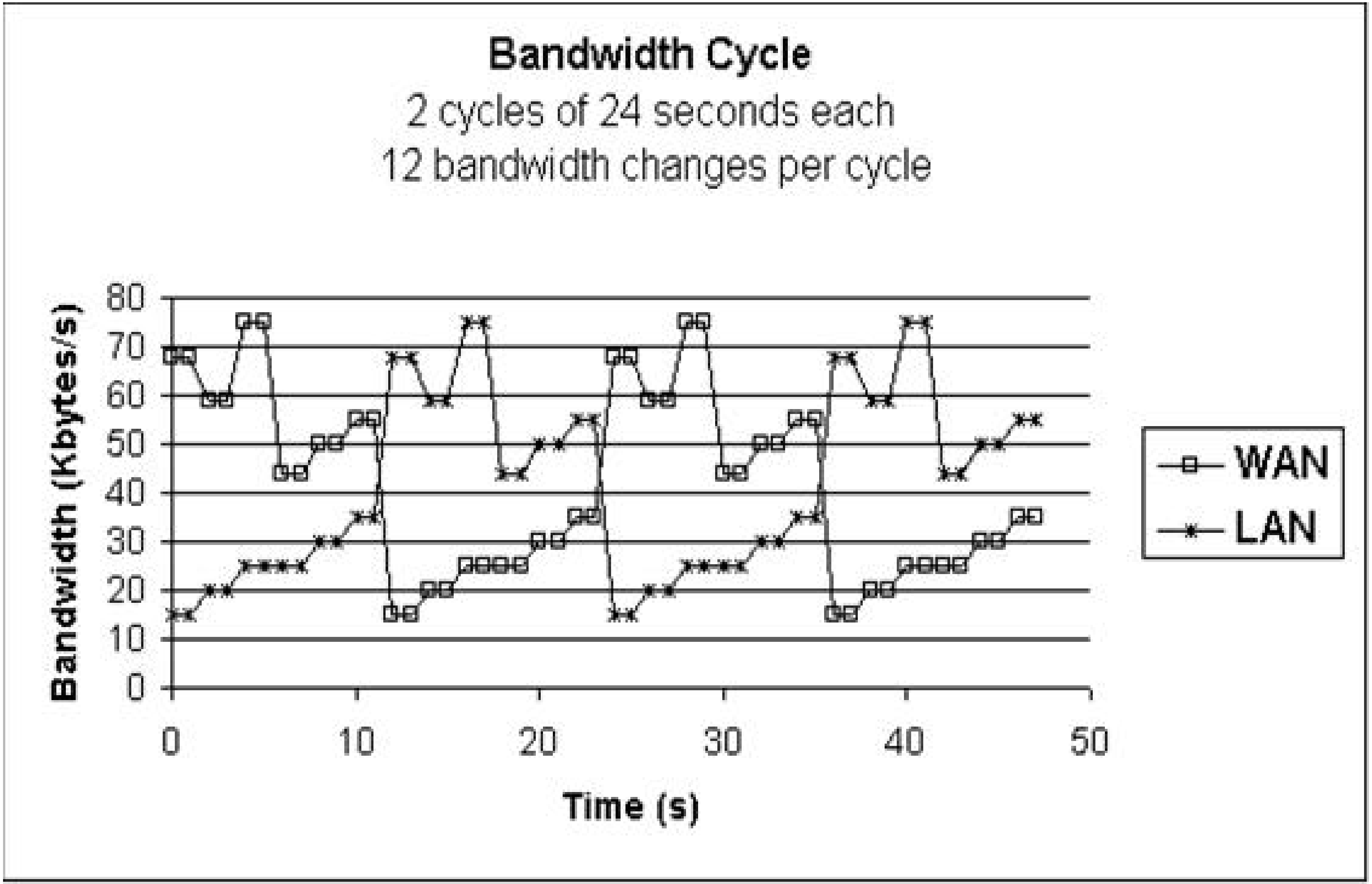
REGULATED WAN

Client (RealPlayer™)

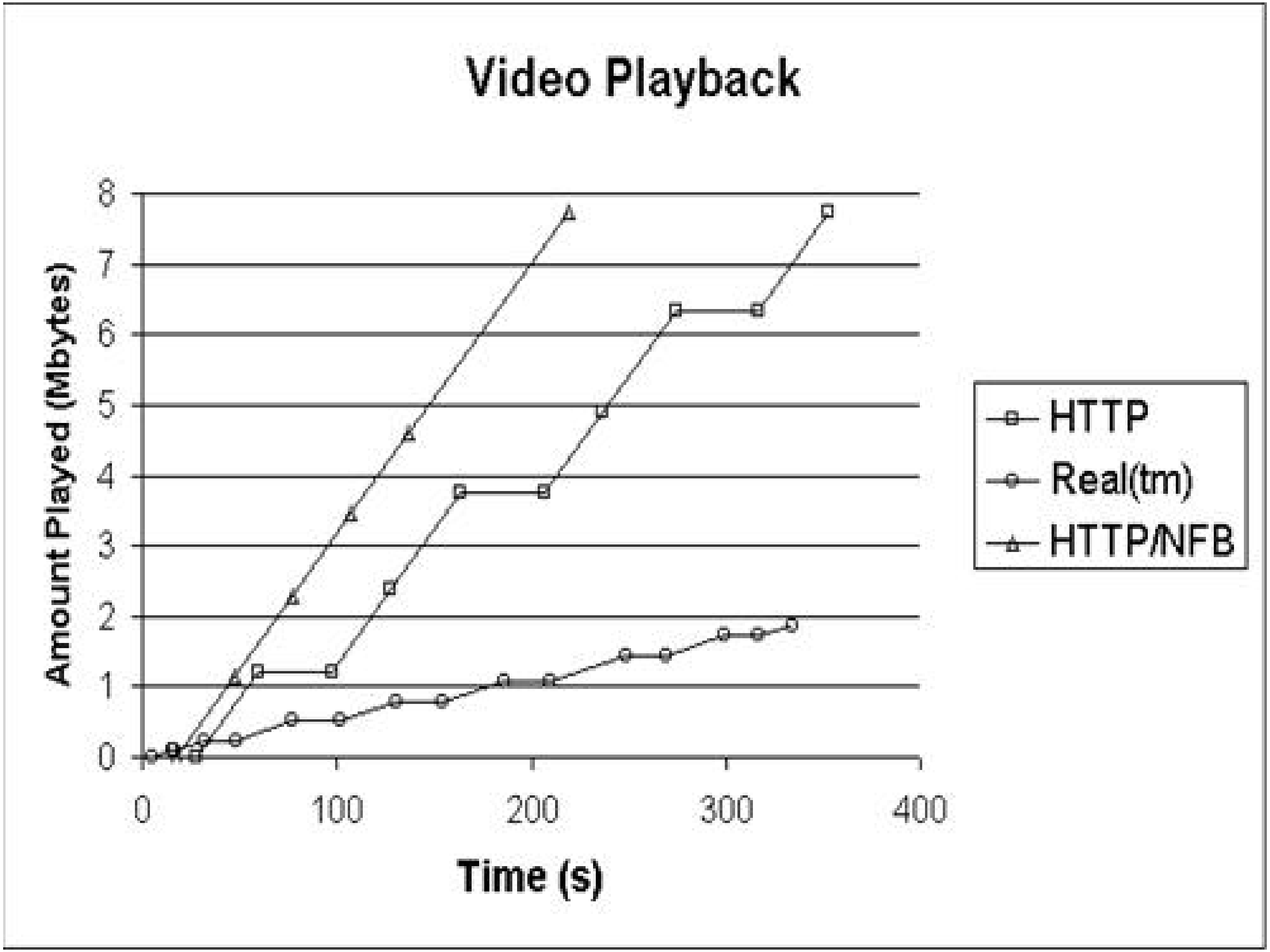
Customizer Server ("Gateway", dummynet regulates WAN/LAN B/W)

Video Server (HTTP And RTSP Server)

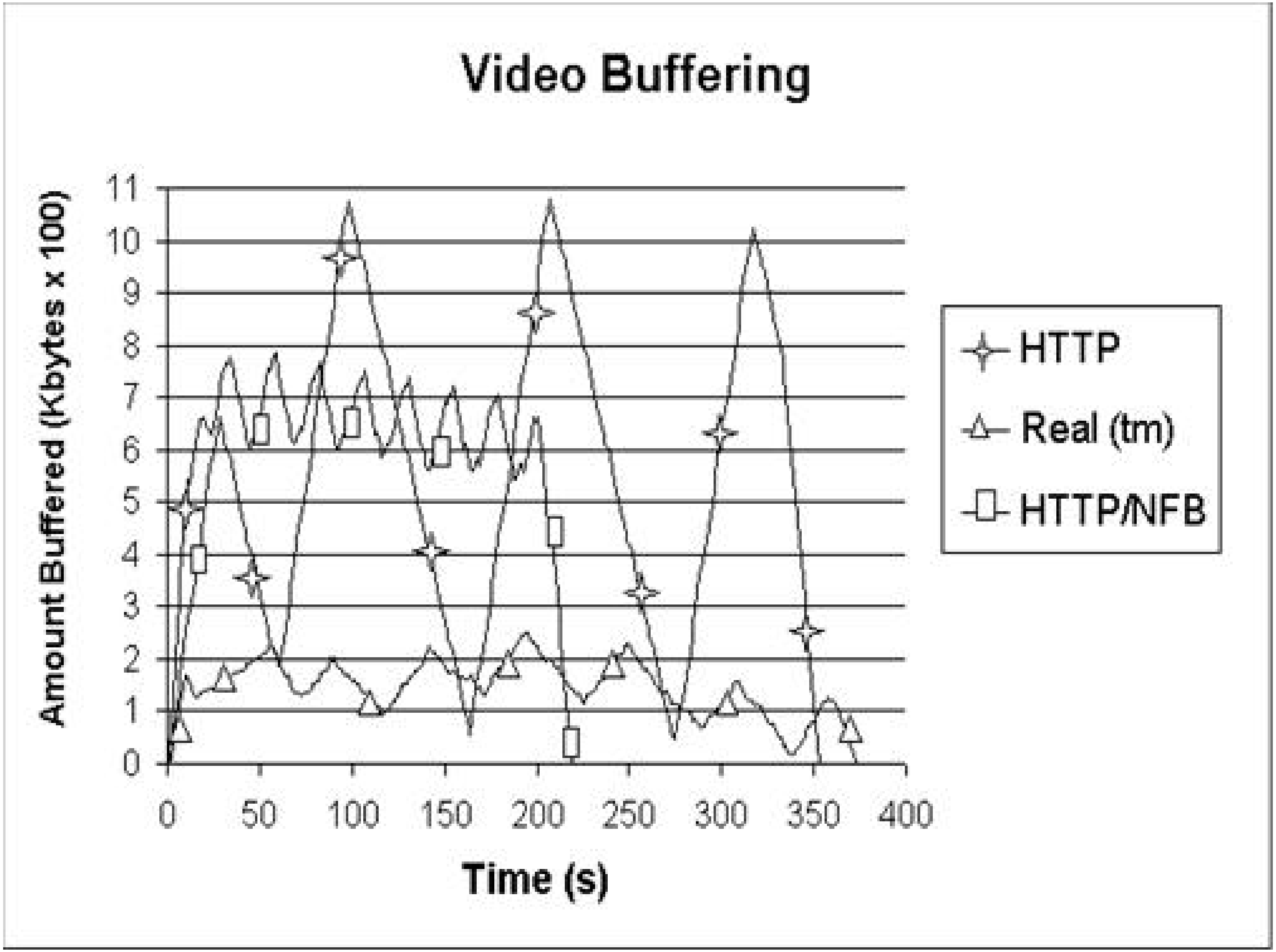
Smoothing Experiment



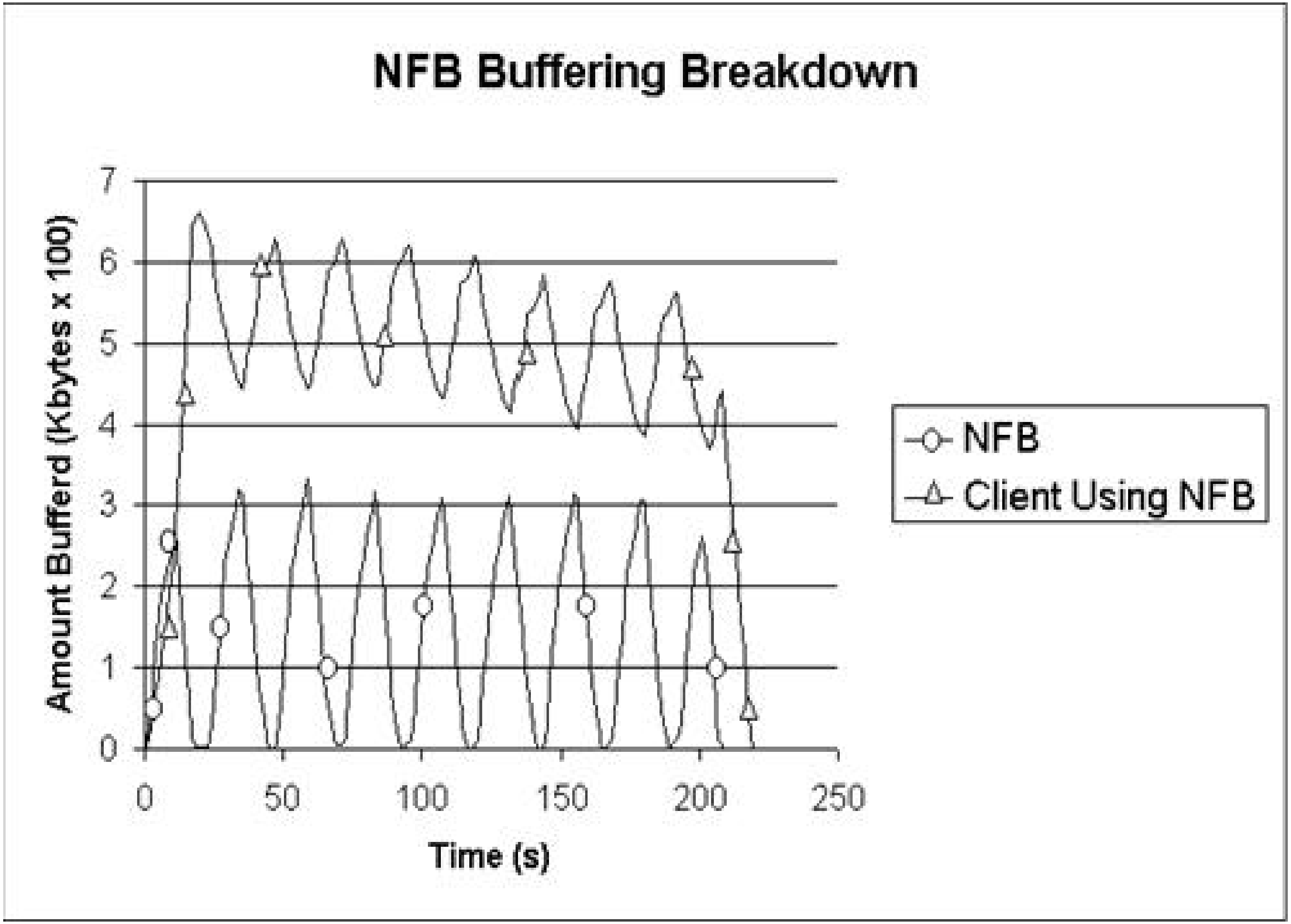
Smoothing Results: Playback



Smoothing Results: Buffering



NFB Buffering Breakdown



Summary

- Novel Web middleware architecture for improved wireless web access
  - Remote computation, dynamic deployment, two points of control, callback programming model
- Supports a variety of applications
  - Filtering, encryption, transaction recorder, video buffering
  - NFB smoothing can improve video playback
- Implementation
  - Java-based and uses existing Web mechanisms

Customizers Are Efficient

- Customizer overhead ~ 4.8 ms
  - Roughly 1-5% of typical transfer times
- Typical transfer times from UCSD:
  - www.yahoo.com ~ 128 ms
  - www.suntimes.com/index ~ 404 ms
  - www.cnn.com ~ 475 ms
- Above doesn’t consider performance improvements of the Customizer