# Rate-Based Execution Models For Real-Time Multimedia Computing

## Introduction

Kevin Jeffay
Department of Computer Science
University of North Carolina at Chapel Hill
*jeffay@cs.unc.edu*
September 22, 1997

***http://www.cs.unc.edu/~jeffay/courses/pisa***



# Rate-Based Execution Models For Real-Time Multimedia Computing

## *Outline*

- Rate Based Execution: The case against Liu & Layland style models of real-time computing
- A Liu & Layland extension for rate-based execution?
- Fluid-flow models of resource allocation for real-time services
- Proportional share CPU scheduling
- On the duality of proportional share and traditional Liu & Layland style resource allocation

# Rate-Based Execution

## The case against Liu & Layland models

- What is “real-time” about multimedia?
  - » The structure of a canonical distributed, interactive, multimedia application
- Performance requirements of real-time multimedia applications
- Realizing multimedia application requirements with periodic and sporadic tasks
  - » Do they fit?



# Distributed Multimedia Applications

## Examples

- Entertainment
  - » Video-on-demand
  - » Multi-player games
- Collaborative work
  - » Remote consultation
- Distance learning
  - » Interactive television
  - » Content-on-demand
- Communication
  - » Internet telephony & videoconferencing

# Interactive Multimedia Applications
## Performance requirements


- High-level goal: support human-to-human communications
- Primary performance parameters
  - » latency
  - » throughput
  - » continuity of playout
  - » media synchronization



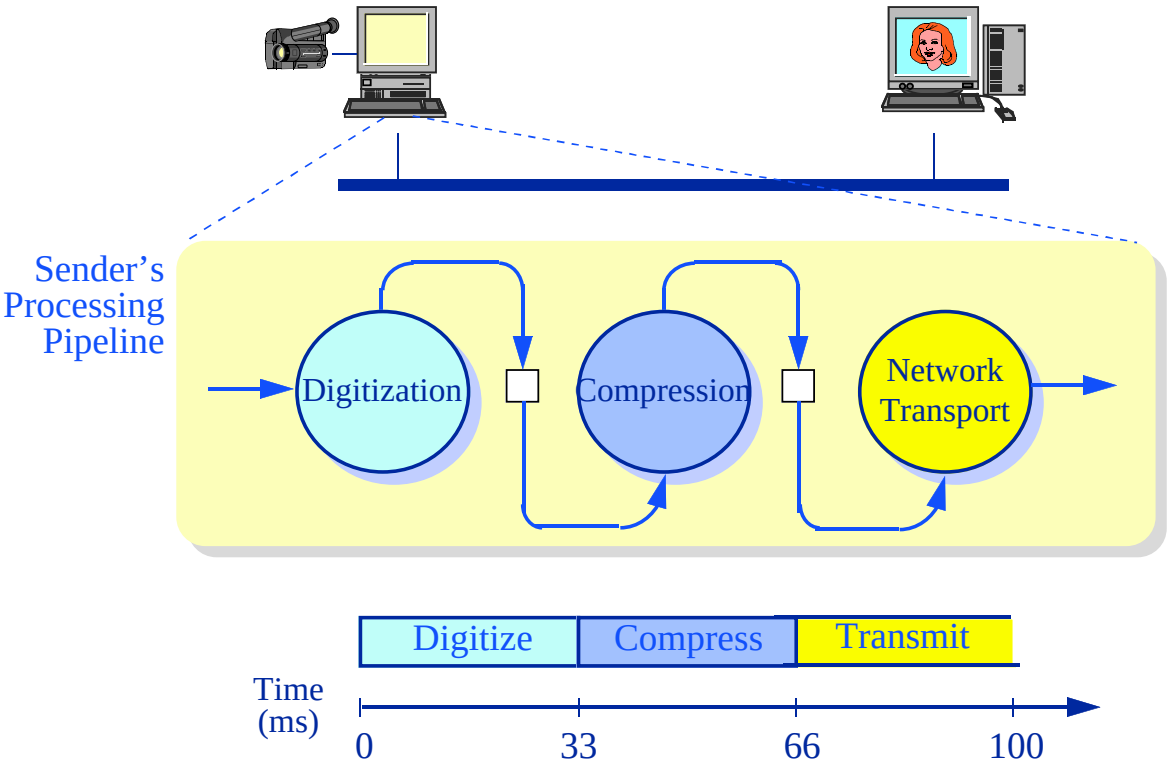
# Interactive Multimedia Applications
## Performance requirements

- Latency — the duration between acquisition of a signal and its display

- Videoconferencing latency requirements
  - » telephony literature — 100 ms roundtrip
  - » multimedia networking literature — 250 ms one-way
  - » CSCW literature — tolerance of latency as high as 400 ms

# Latency in Computer-Based Video Systems

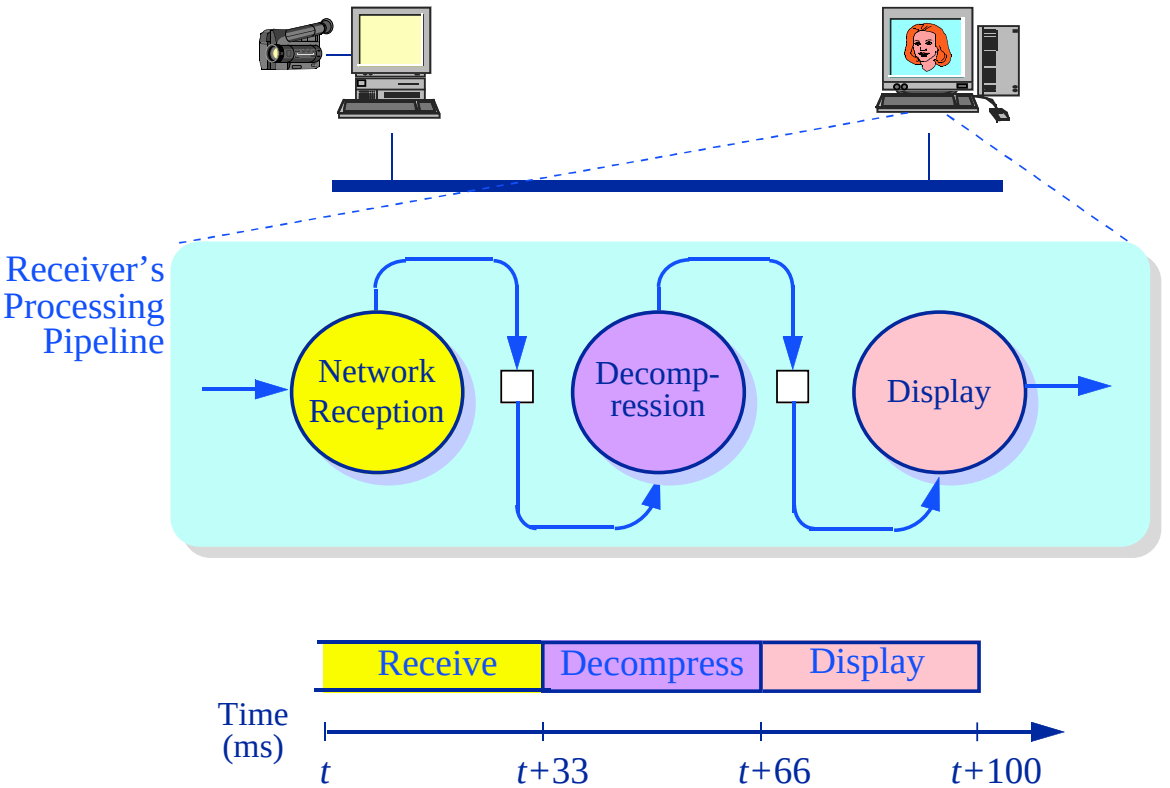
## Canonical application structures





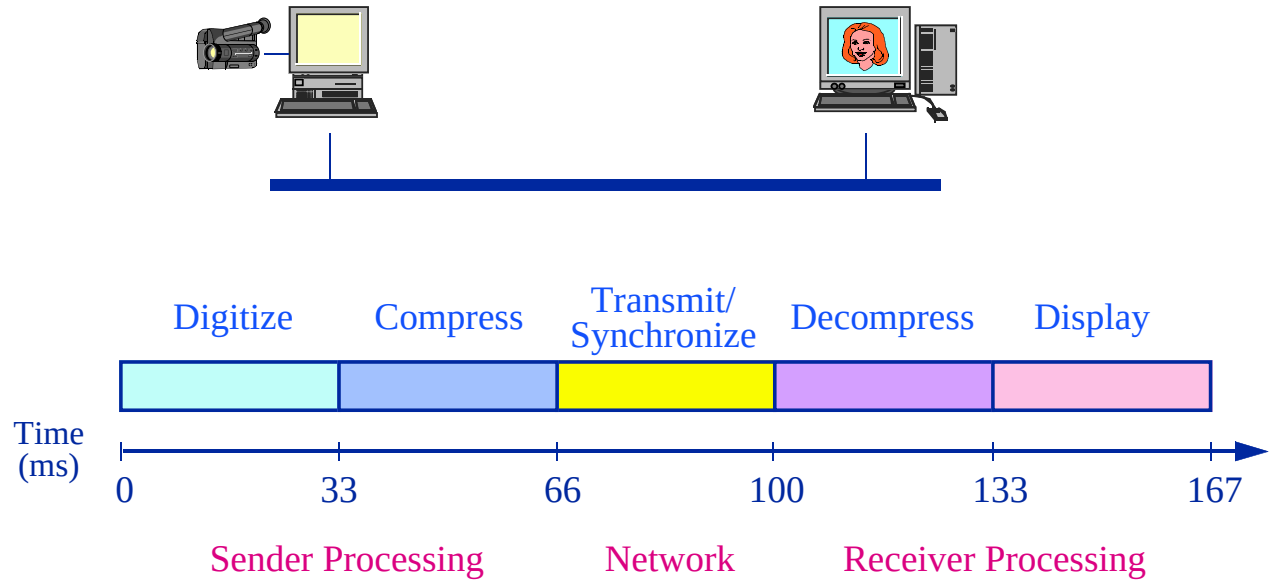

# Latency in Computer-Based Video Systems

## Canonical application structures



# Latency in Computer-Based Video Systems

## Best case end-to-end latency






# Latency in Computer-Based Audio Systems

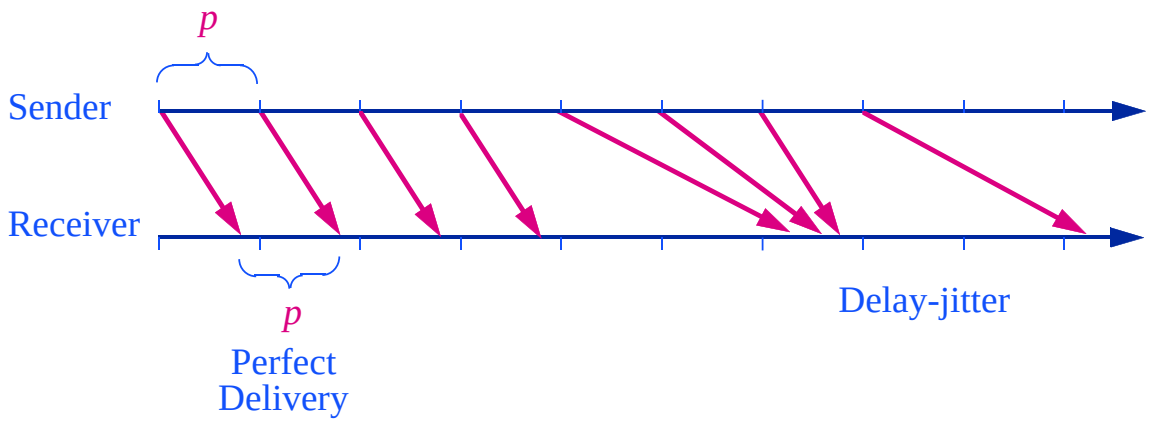
## How bad can audio latency be?

- Just as bad as video if lip-synchronization is required
- Otherwise, it depends on how one manages the network interface
  - » Video frames are typically too large to fit into a single network packet
  - » Multiple audio samples can be transmitted together
- Example: An audio codec generating 1 byte of data every 125 $\mu$s
  - » Building 500 byte packets requires 62.5 ms
  - » Building 1,500 byte packets requires 187.5 ms

# Performance Requirements

## Delay-jitter

- Latency
  - » 250 ms one-way
- Delay-jitter — Variation in end-to-end latency

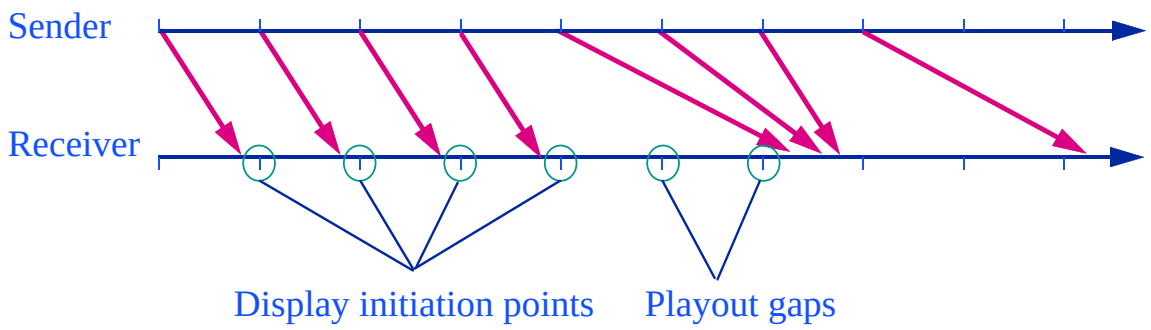





# Performance requirements

## The impact of delay-jitter

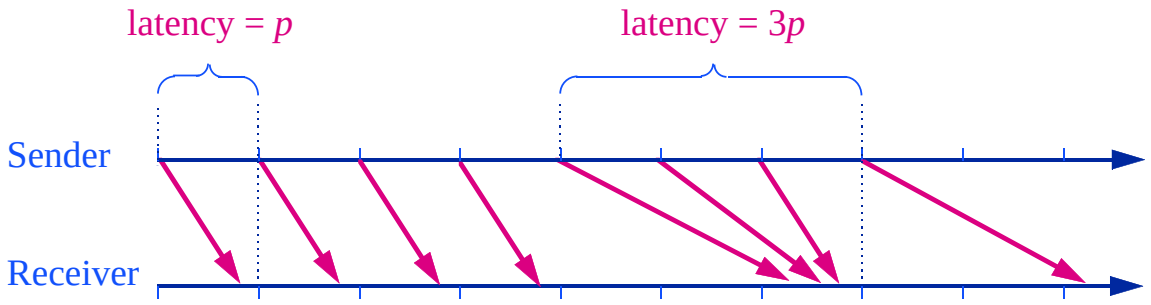
- Delay-jitter leads to “gaps” in the playout of media and increases playout latency



# Performance requirements

## The impact of delay-jitter

- Delay-jitter increases playout latency




- Delay-jitter requirements are application dependent but are largely closely coupled with latency requirements



# Performance Requirements

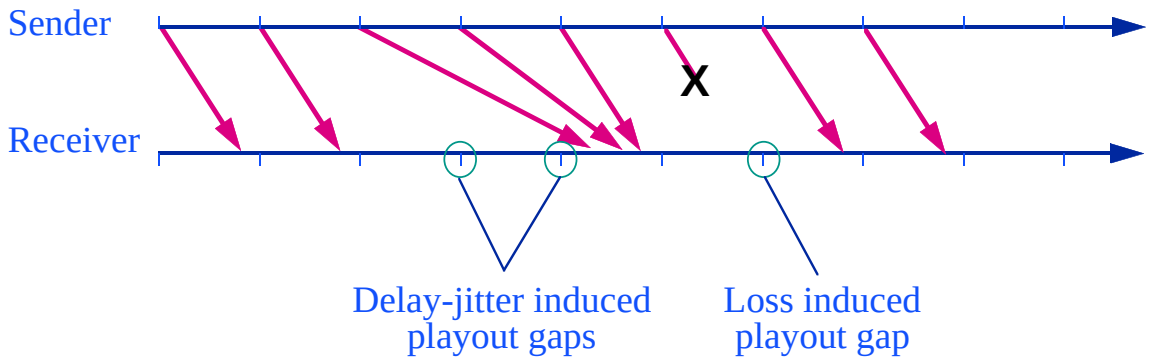
- Latency
  - » 250 ms one-way
- Delay-jitter

- Throughput —the effective delivered frame or sample rate
  - » For video the issue is *motion perception*
  - » For audio the issue is comprehension
- Loss — the complement of throughput

# Performance requirements

## Loss

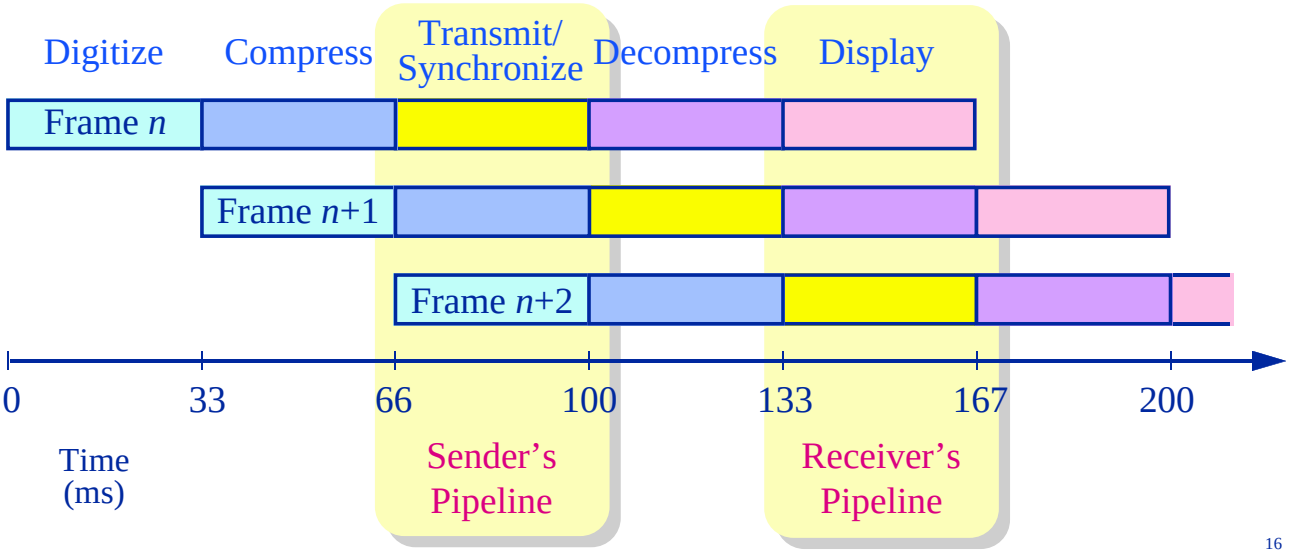
- Loss has the same effect as delay-jitter: *gaps*
  - » With a potentially beneficial effect of potentially lower latency






# Avoiding Loss in the End System

## Real-time management of a processing pipeline



# Performance requirements

## Loss requirements

- Audio — 1-2% sample loss
  - » individual sample losses (depending on sample size) are noticeable
  - » 5-10 lost samples per minute are tolerable
    (the distribution of loss is critical)
- Video — 10-15 frames/s required for minimal motion perception
  - » highly application dependent
  - » video loss raise issues of “network citizenship”



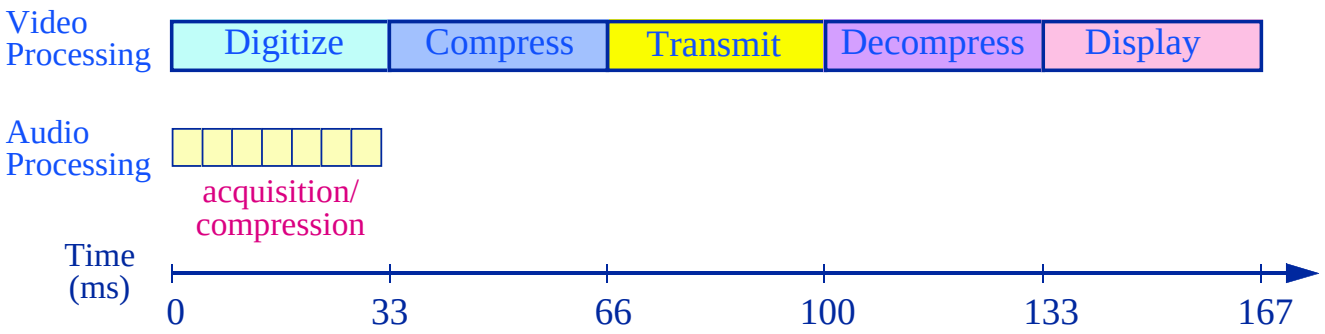
# Performance Requirements

- Latency
  - » 250 ms one-way
- Delay-jitter
- Throughput —the effective delivered frame or sample rate
  - » For video the issue is *motion perception*
  - » For audio the issue is comprehension
- Loss
- Lip synchronization
  - » The temporal relationship between an audio and video stream representing a human speaking

# Performance Requirements

## Lip synchronization

- Perfect lip synchronization requires audio playout at time 133

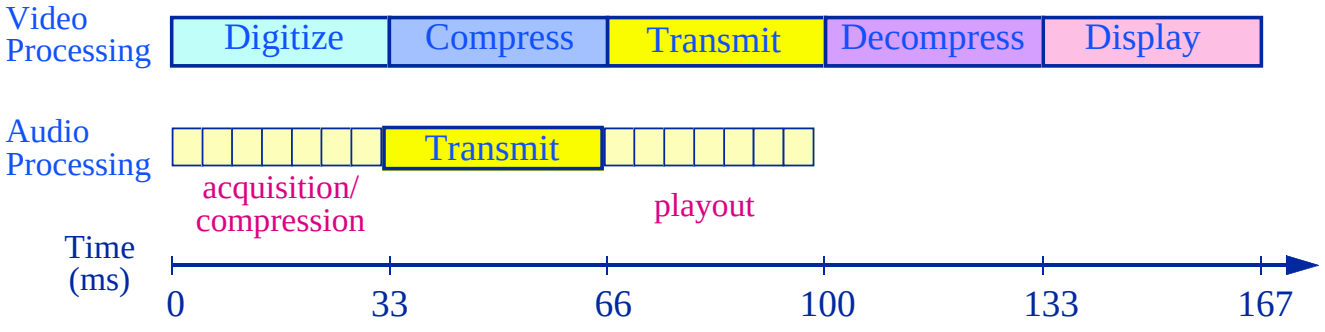





# Performance Requirements

## Lip synchronization

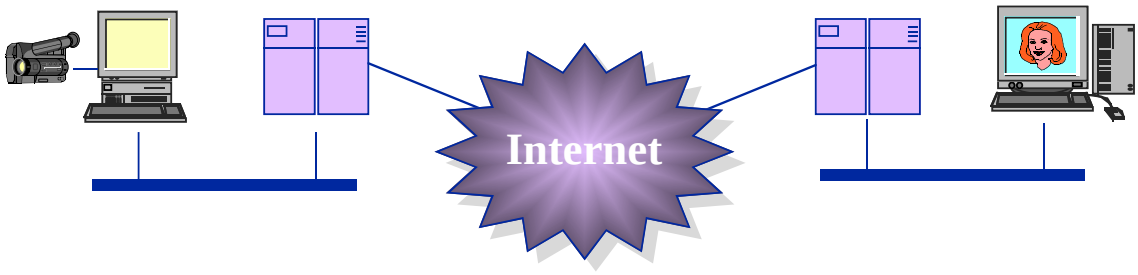
- Varying lip sync can be an effective technique in mitigating high video latency




- But... this is fundamentally unnatural!

# Interactive Multimedia Applications

## Performance requirements




- No more than 250 ms end-to-end, one-way latency
- Continuous audio
- Minimum of 10 frames per second video throughput
- “Loosely synchronized” playout — ± 80 ms skew



# Requirements For Multimedia Computing

## What might the operating system do?

- Support *adaptive* execution
- Provide *feedback* on actual performance
- Provide *integrated* real-time resource management
- Allow high-level specification of performance parameters

# Requirements For Multimedia Computing

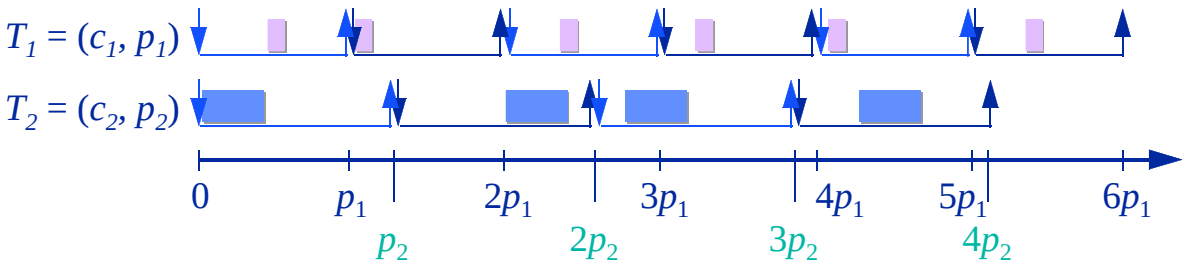
## The essential problem is...

- Operating system overhead
  - » The communications centric OS group
- Allowing applications fine-grained control over resource allocation
  - » The extensible/customizable OS group
- Providing predictable, real-time communication and computation services
  - » The real-time OS group



# Traditional Real-Time Operating Systems Support

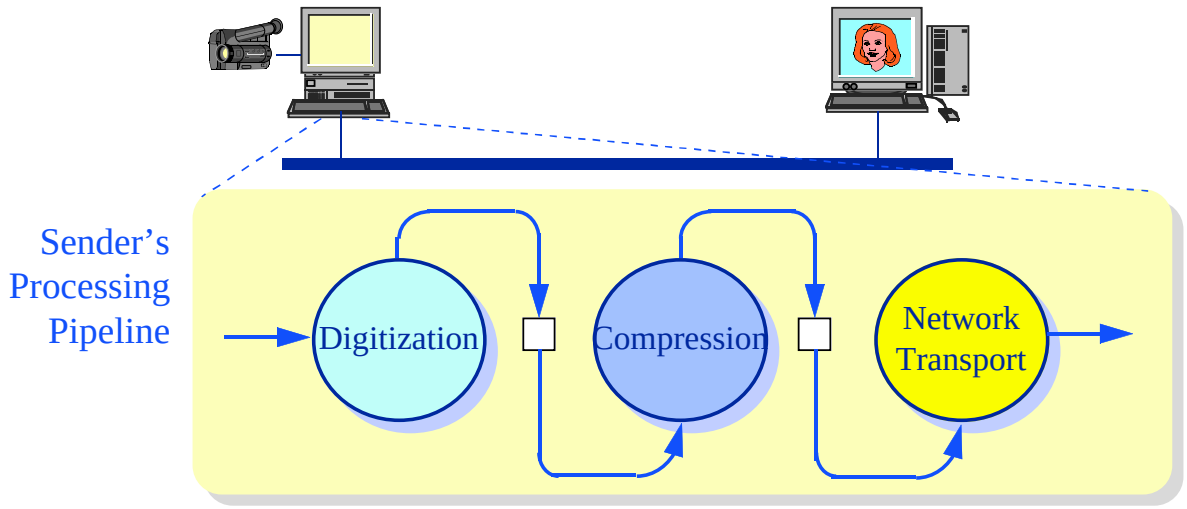
- A real-time system is a collection of *periodic* or *sporadic* tasks
  - » each task $T_i = (c_i, p_i)$, $c_i$ is the cost of executing $T_i$, $p_i$ is the period of of $T_i$




- ... that are scheduled using *rate-monotonic* or *earliest deadline first* algorithms

# Barriers to Adopting RT Systems Technology

## Send-side media processing

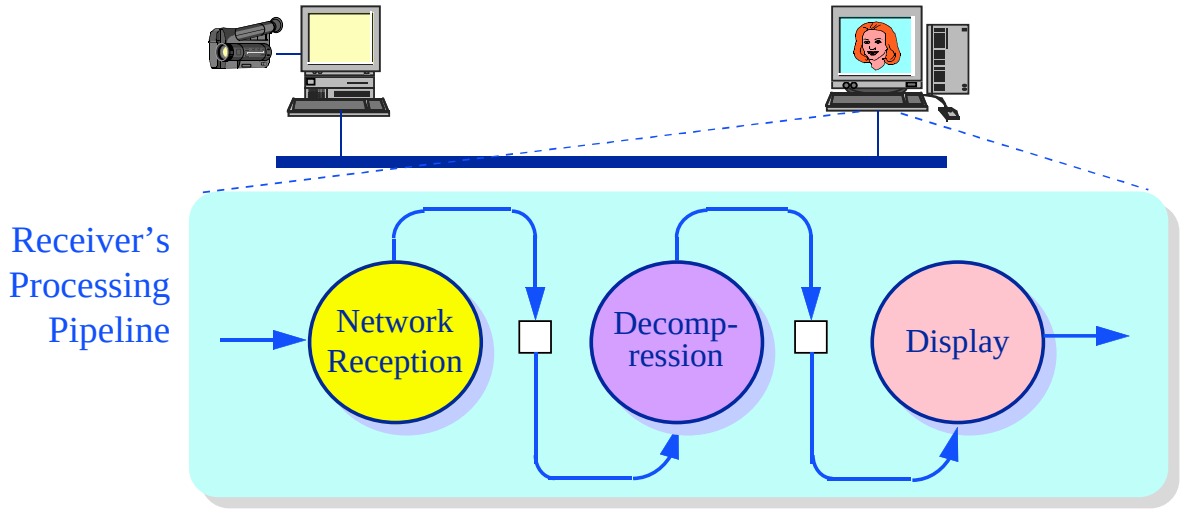



- Real-time processing issues:
  - » Acquire all samples from input devices and deliver to the network interface in real-time
  - » Support adaptive pacing of network transmissions



# Barriers to Adopting RT Systems Technology

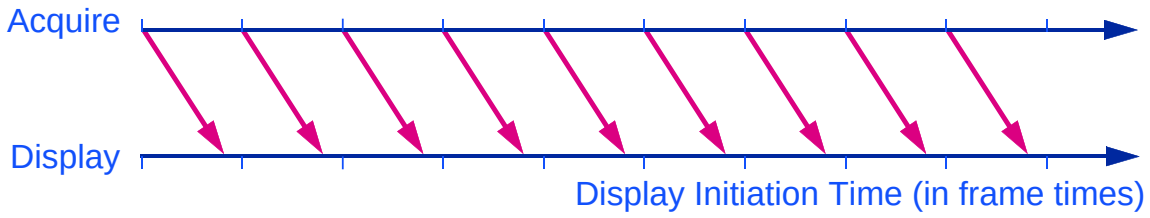
## Display-side media processing




- Real-time processing issues:
  - » Receive all samples from the network and deliver to the display application in real-time

# Barriers to Adopting RT Systems Technology

## Display-side media processing

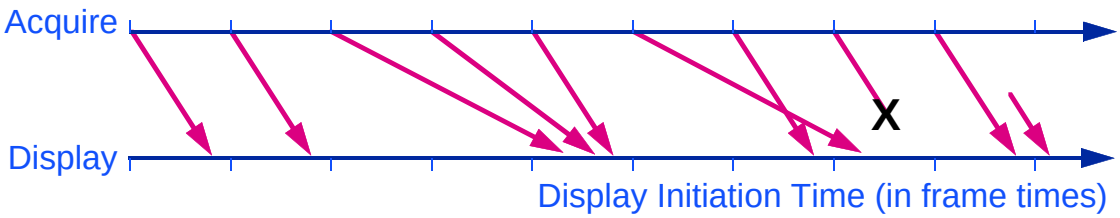



- The problem: Receive frames from the network and deliver to a display application so as to ensure
  - » Continuous playout
  - » Minimal playout latency
- The theory: Multimedia is easy — it's periodic!
  - » Apply existing theory of periodic and sporadic tasks



# Barriers to Adopting RT Systems Technology

## Display-side media processing: *The Practice*!

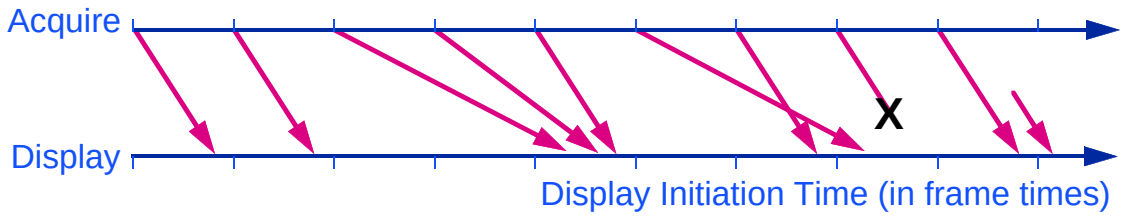



***Nothing is periodic in a distributed system!***

- The effects of distributed systems pathology:
  - » Variable message transmission times
  - » Out-of-order message arrivals
  - » Lost & duplicate messages
- Fundamental problems
  - » Systems: Real-time management of the network interface
  - » Application: Ameliorating the effects of the pathology

# Managing the Network Interface

## Issues



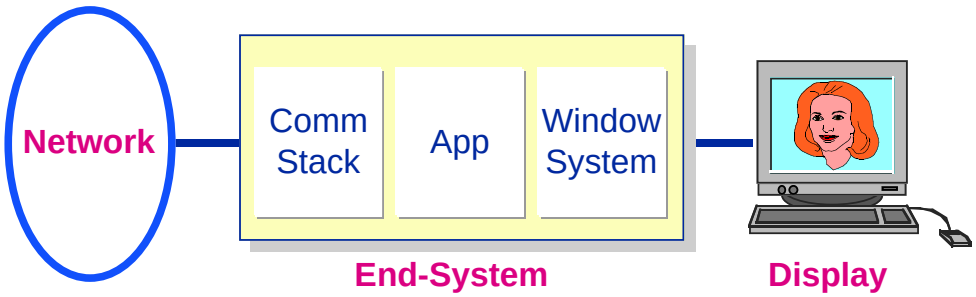

- Message transmissions are periodic but arrivals are not
  - » Standard IP delivery semantics
- Packets fragmented in the network must be reassembled
  - » Messages have deadlines, packets do not



# Barriers to Adopting RT Systems Technology

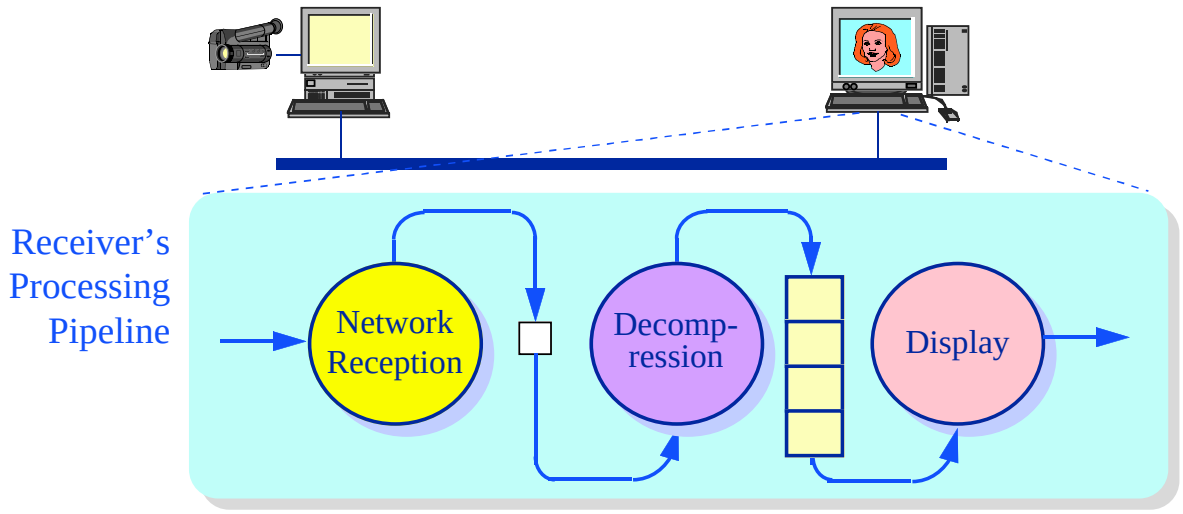
## Realizing integrated services

- Integrated quality of service management
  - » "From the wire to the glass"



# Realizing Integrated Services

## The "Hurry up and wait" phenomena

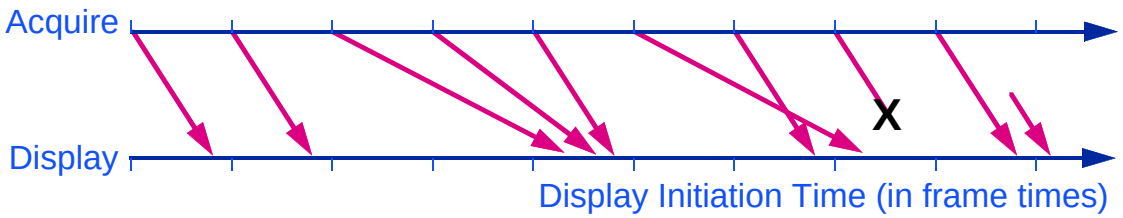



- All interactive playout applications use some form of *elastic queueing*
  - » Real-time constraints determined by the current depth of the playout buffer



# Polling-Based Solutions

## What about periodic and aperiodic "servers"?




- Use periodic and aperiodic processes to poll for arriving packets
  - » Potentially increases the latency of processing individual packets
  - » Begs the question of how packets are read in and buffered
  - » Potentially leads to a pessimistic analysis of the system

# Real-Time Requirements For Multimedia Computing

- Audio and video are a new & unique data type
  - » *Continuous* media with implicit timing
  - » Semantics imply *real-time* processing
  - » *Variable* reliability & loss requirements
- New system challenges
  - » *Communications centered* operating systems
  - » *Tunable* real-time communication and computation services
  - » Resource *monitoring* and *charging* policies
- Integrated solutions required
  - » *End-to-end* solutions required
  - » *Connection orientation* within operating systems & networks
  - » *Application-level control* of system resource allocation

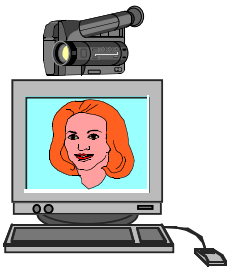


# The Case Against Liu & Layland Models of Real-Time Computing

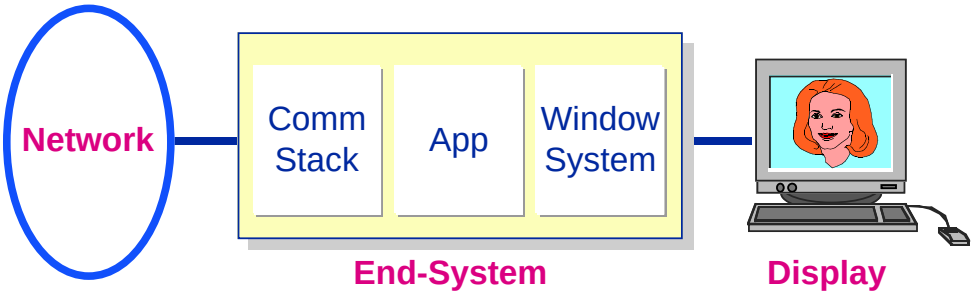
- The existing theory of periodic and sporadic tasks does not directly address the requirements of “soft-real-time” multimedia applications
  - » Managing the network interface
  - » Integrating packet processing and application processing
- Periodic and aperiodic server approaches are not entirely satisfying
  - » Most approaches require some encoding of a minimum inter-arrival time for an event
  - » Don’t leverage the fact that applications are adaptive
  - » Slave non-real-time processing to real-time processing

# Towards Rate-Based Execution Models
## Real-time requirements

- Device management
  - » CODECS (cameras)
    - » Bounded response times
  - » Network adaptors
    - » Pacing/policing



- Integrated protocol & application processing





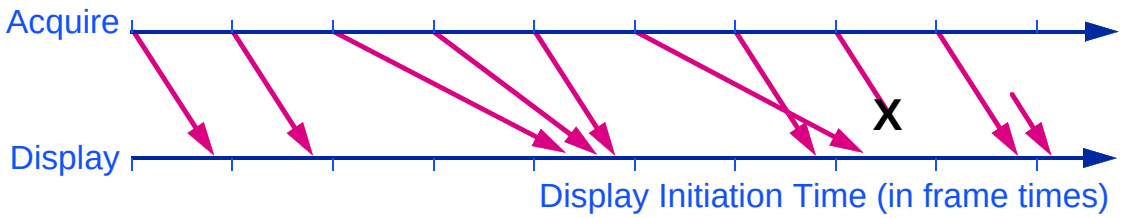

# Rate-Based Execution Models For Real-Time Multimedia Computing

## *Outline*

- Rate Based Execution: The case against Liu & Layland style models of real-time computing
- A Liu & Layland extension for rate-based execution?
- Fluid-flow models of resource allocation for real-time services
- Proportional share CPU scheduling
- On the duality of proportional share and traditional Liu & Layland style resource allocation

# Rate-Based Computing

## Beyond periodic & sporadic models




- An event-based model — *rate-based execution*
  - » Process make progress at the rate of processing *x* events every *y* time units, each event is processed within *d* time units
- A time-sharing model — *proportional share resource allocation*
  - » Processes make progress at a precise, uniform rate — as if executing on a dedicated processor with $1/n^{th}$ original capacity