Distributed Multimedia Systems

IN5020/9020 Autumn 2023
Lecturer: Amir Taherkordi

October 30, 2023

UiO : University of Oslo

1

Outline

- Introduction
  - Support for Continuous Media
- Media Synchronization
- Quality of Service
- Resource Management
- Multimedia Streaming
- Stream Compression
- Stream Adaptation
- Media Distribution Services
- Summary

2

What is Multimedia?

- Digital multimedia
  - Computer-controlled integration of text, graphics, drawings, still and moving pictures, animation, audio, and any other medium
  - All digital
- A **Multimedia system**
  - is characterized by the generation, processing, manipulation, rendition and storage of multimedia information.
- **Discrete** vs. **Continuous** media
  - Discrete media type is **time independent** or static media: e.g. text or single images
  - A continuous media type has an implicit time dimension, e.g. video, animation and audio.
    - Timing plays a crucial role in continuous media (e.g. correct play out time of audio samples)
- Focus of this lecture: **continuous media** (audio/video)

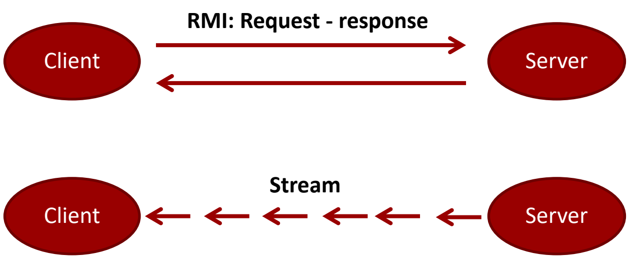
Continuous Media as Data Streams

- Data Streams
  - is a sequence of data units
  - can be applied to both continuous and discrete media
- **Timing** is crucial in continuous data streams
- Timing aspects
  - *asynchronous* transmission: typical for discrete media, e.g. transferring a file as a data stream
  - *synchronous* transmission: e.g. a sensor that samples temperature at a certain rate
  - *isochronous* transmission: particularly interesting for continuous media

Key Requirements

- The need to **represent** continuous media in distributed systems
  - Programming models (middleware abstractions)
  - Representation
- The need for **synchronization** mechanisms
- The need to specify and dynamically change the **Quality of Service** (QoS) of the transmission (and thus presentation) of continuous media
  - e.g. balance cost and quality

Programming Models



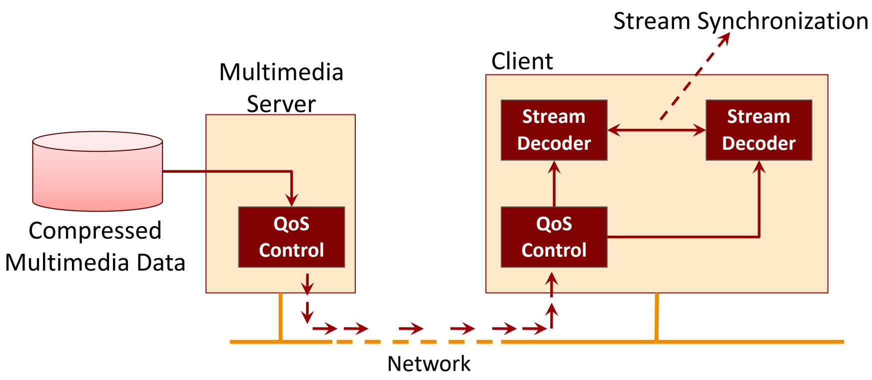
- Distributed multimedia systems typically **use streams** to transmit media
- Continuous transfer of data units which
  - can **not** be **interrupted**
  - includes some **timing constraints** between the data units
- When programming continuous media, we **commonly** use a **streaming programming model** for data transfer.

Representation

- Continuous representation media
  - the **temporal relationship** between data items of the stream must be preserved
  - Audio
    - built up of series of audio samples (e.g. 16 bit) representing amplitudes
    - must be played back at same rate as it was sampled (e.g. 44100 Hz)
  - Motion (video)
    - built up of series of images (frames)
    - must be displayed at a uniform spacing in time (e.g. 30-40 msec per image)

Synchronization: General Architecture for Streaming

- QoS-aware streaming of stored multimedia data over a network

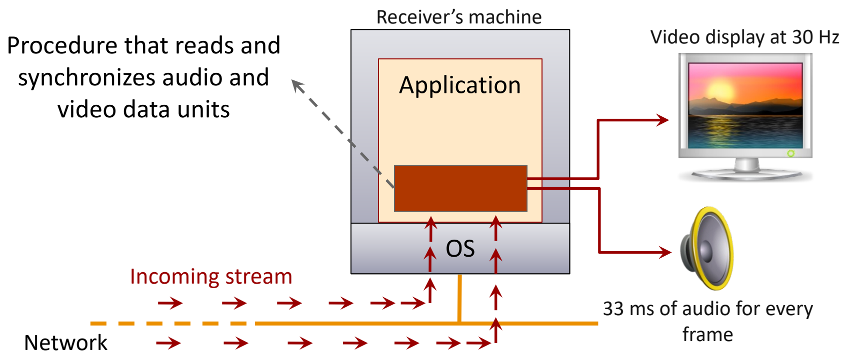


Different Forms of Synchronization

- **Intra media:** maintain uniform time spacing of a single continuous media stream
- **Inter media:** synchronization between several streams, for example
  - lip synchronization between video and audio streams
  - synch of video and text streams (subtitles)
- **Distributed state:** stop video operation by a user should be observed by all within 500 ms
- **External:** synchronization of time based streams with data in other formats, e.g. in conferencing and cooperative applications: animations, slides, whiteboards, shared documents, etc.

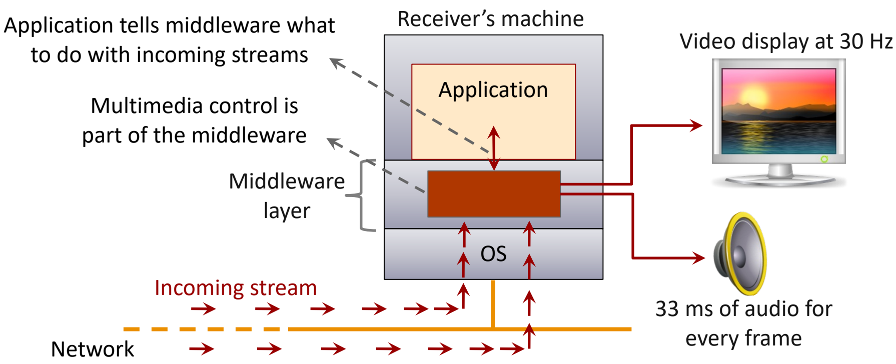
Synchronization Mechanisms (1/2)

- The principle of explicit synchronization on the level of data units



Synchronization Mechanisms (2/2)

- The principle of synchronization as supported by high level interfaces



11

Distribution of Synchronization Mechanisms

- Receiving side of a complex stream (stream consisting of many substreams) needs to know how to do the synchronization (**synchronization specification**)
- Common practice: **multiplex substreams** into one stream when single source (implicit synch spec)
  - The approach of MPEG: each data element in multiplexed stream is time stamped (playout time)
- **Synchronizing** independent substreams at **receiving side** can be extremely **difficult** as delay may vary unpredictably between different channels
  - may use timestamps also here

12

Key Requirements

- The need to **represent** continuous media in distributed systems
  - Programming models (middleware abstractions)
  - Representation
- The need for **synchronization** mechanisms
- The need to specify and dynamically change the **Quality of Service** (QoS) of the transmission (and thus presentation) of continuous media
  - e.g. balance cost and quality

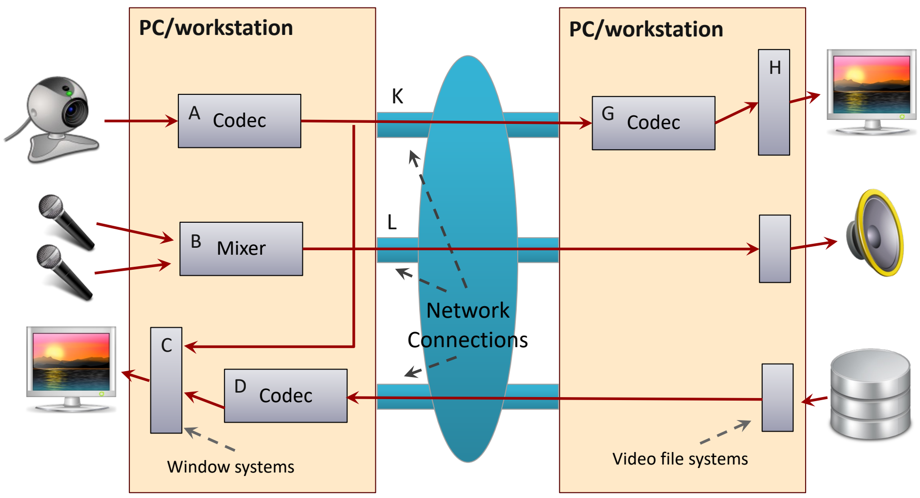
Quality of Service (QoS)

- IDL: **functional** specification which tells us “what” can or should be done
- Quality of Service: **non-functional** specification that tells us “how” the function should perform
- Quality of Service (QoS)
  - An abstract specification of the non-functional requirements to a service
- QoS Management
  - **Monitoring** and **control** of a system to ensure that it fulfills the required QoS

Question of Resource Management

- QoS guarantees require that **resources are allocated and scheduled to multimedia** applications under timeliness requirements
- Need for QoS-driven **resource management** when resources are **shared** between several applications and **some** of these have time **deadlines**
- Resource management are **mechanisms** that ensure that such **guarantees** are met.

Example: Resource Needs A/V Streaming App.



QoS Management

- QoS aware applications require QoS management
  - QoS specification
  - QoS parameter translation and distribution
  - QoS negotiation
  - Admission control / reservation
  - QoS monitoring
  - QoS renegotiation / resource adaptation / QoS adaptation
  - Resource deallocation

- QoS management relies on QoS models
  - For describing QoS requirements and QoS parameters

QoS Models for Streaming

- Usually expressed as a set of QoS dimensions and categories
- **QoS dimension**: an aspect of QoS that can be measured on a stream
  - delay, throughput, ...
- **QoS category**: a grouping of QoS dimensions
  - Represents a type of user or application requirements
  - Example (QML)

```
type Performance = contract {
    delay: decreasing numeric msec;
    throughput: increasing numeric mb/sec;
};
```

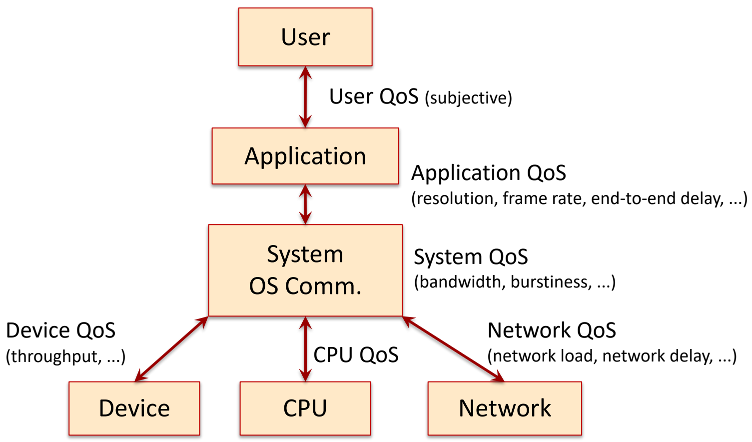
QoS Categories for Streaming

QoS categories	Ex. QoS-dimentions for stream interaction
Timeliness	End-to-end delay, max allowed jitter
Volume	Observed throughput as frames per second
Reliability	% frame loss, bit error rate per frame

Varying commitment levels: “best effort” vs. guaranteed

QoS Specification and Translation

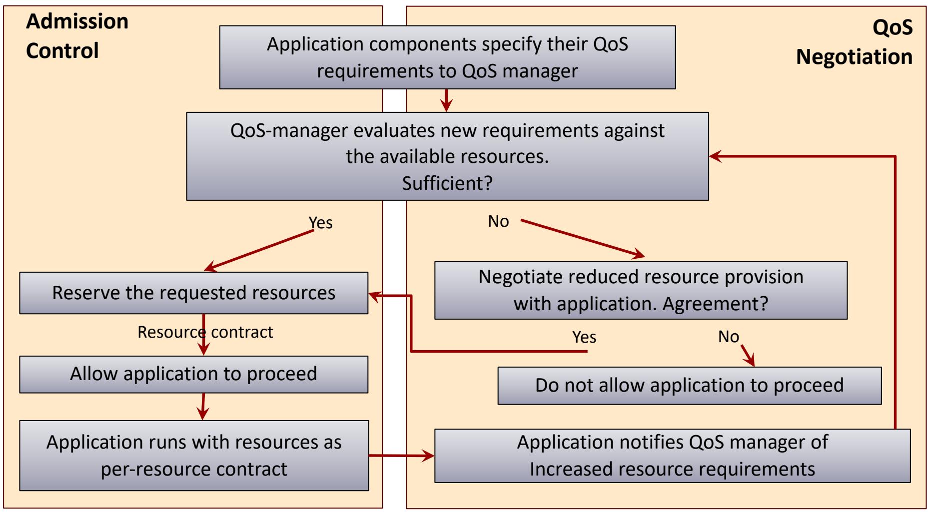
- Layer/component specific QoS-model



QoS-driven Resource Management

- Next step: **translation** of **application level** QoS requirements to lower level resource needs communicated to **resource managers**
- Resource manager
  - Performs **admission control** and **scheduling**
  - **Schedules** multimedia tasks such that resources are available when there is a need for them
- Resources
  - Shared: CPU, network adapter, buffer, comm. bandwidth, disc, …
  - Exclusive: camera, speaker, special hardware units, …

QoS Negotiation and Admission Control



Negotiation and admission control ensures that guarantees can be met.

Admission Control

- QoS values must be mapped to resource requirements
- Admission test for
  - Schedulability
    - can the CPU slots be assigned to tasks such that all tasks receives sufficient slots?
  - Buffer space
    - e.g. for encoding/decoding, jitter removal buffer, ...
  - Bandwidth
    - e.g. MPEG1 stream with VCR quality generates about 1.5 Mbps
  - Availability/capabilities of devices
  - ...

Resource Allocation Strategies

- Made according to service type
  - different services may have different policies
- **Pessimistic** allocation
  - consider worst case when reserving resources
  - guaranteed deterministic quality of service
  - under utilization of resources
- **Optimistic** allocation
  - considers average case when reserving resources
  - statistically guaranteed quality of service
- **No reservation**
  - "best effort"

CPU Management in End Systems

- **Make CPU available** for all multimedia applications when needed
- Real time requirements – OS must use real time scheduling
- Observation: time critical operations in multimedia applications are **often periodic**
- Common assumption
  - Processing of continuous media data **must occur in exact pre-determined, periodical intervals**.
  - Operations on these data occur again and again, and must be completed by **certain deadlines**
- Problem for scheduling
  - Find a feasible schedule that allows all time critical continuous media tasks to reach their deadlines

EDF and RM

Two algorithms for scheduling of **periodic tasks**

- ***Earliest Deadline First (EDF)***
  - Deadline associated with each work item
  - Tasks with the earliest deadline have highest priority
  - Proven to be optimal for allocating a single resource based on timing criteria
  - Dynamic and optimal algorithm
    - by arrival of new task, must calculate a new priority order
- ***Rate Monotonic (RM)***
  - Tasks with shortest period have highest priority
  - Optimal for periodic tasks
- *Deadline* ***violations***
  - aborts task that can not reach their deadlines
  - application specific handling by suitable language mechanisms (e.g. callbacks)

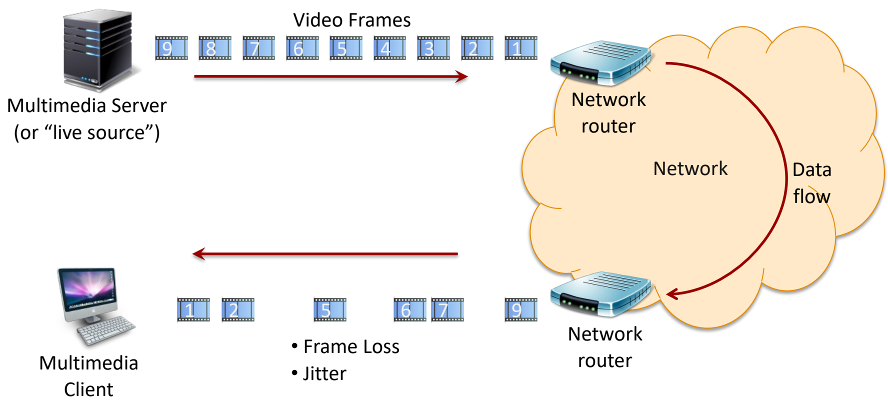
Challenges When Streaming over the Internet

- Internet is based on IP
  - Efficient due to packet switching
  - Best effort, **no guarantees** wrt bandwidth and delay
  - No resource reservation is possible in intermediate routers, **packets can be dropped**
- Internet applications commonly use TCP
  - Reliable
  - Implemented over most network types
  - Enable a wide spectrum of applications (file transfer, email, etc.)
  - Reliable: Retransmission ensures all packets are delivered
  - **No timing guarantees**

Protocols for Streaming Media

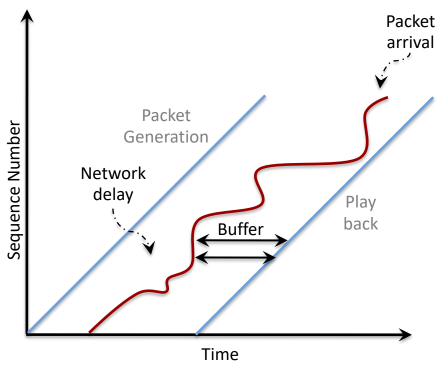
- Streaming and signalling protocols: e.g. RTP and RTCP
- Transport protocols
  - Provides end-to-end network transport functions for streaming applications
  - E.g. TCP and UDP
  - TCP: Reliability based on retransmissions
  - UDP: No retransmissions – suitable for live streaming
  - Real Time Transport Protocol (RTP): an application-level data transfer protocol with timing requirements in each packet
- Session control protocols
  - Defines the messages and procedures to control the delivery of the multimedia data during an established session
  - Define control sequences, such as SETUP, PLAY and PAUSE
  - E.g. RTSP (uses RTP) and SIP (Independent of transport protocol)

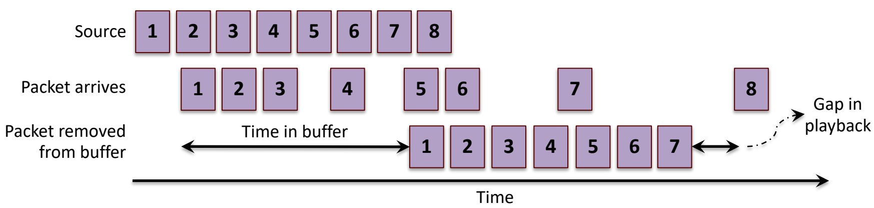
Quality Degradation in IP Networks



Compensating for Jitter

- Playout buffer:
  - Converts variable network delay ("jitter") into fixed delay
  - Typical method for Internet streaming
  - Real Networks, Windows Media Player, QuickTime





Reducing Resource Needs: Compression

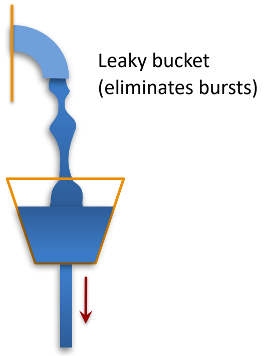
- Three reasons for compression
  - multimedia data requirements to storage **capacity**
  - relatively **slow** external storage devices
  - transmission capacity in **networks**
- Illustrative calculations
  - 1280x720 pixels pr. frame, 24 bits per pixel => ca. 2.7 MB per frame
  - Rate: 30 frames per sec => 81 MB/s (or 648 Mbit/s)
  - In comparison: 4G has a theoretical capacity up to 100 Mbit/s

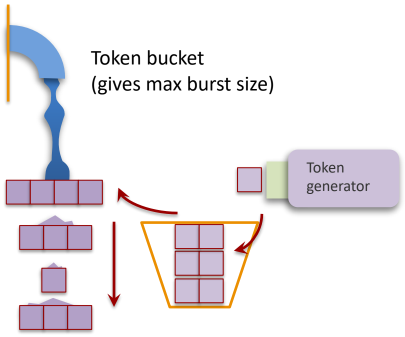
Use of Compression

- Compress prior to storing/transmission
- Decompress prior to presentation
- Typical compression rates for modern open image and video compression standards
  - MJPEG: < 70 : 1 (studio quality: 8 - 10 Mbps)
  - MPEG-1: < 200 : 1 (VCR quality: 1.5 Mbps)
  - MPEG-2: < 200 : 1 (HDTV/DVD quality: 10 - 20 Mbps)
  - MPEG-4/H.264 AVC: many profiles, flexible
  - H.264 AVC with SVC extension: independently scalable in many quality dimensions, flexible
- Compression algorithms can be lossless or lossy and are typically asymmetric

Handling “Stream Burstiness” by Traffic Shaping

- Compression leads to temporal variation in bandwidth consumption
- Regulating the degree of variation in bandwidth consumption of a stream (burst: #media packets with too early arrival)
- Regulating by “smoothing” buffer at sender side





Stream Adaptation when no QoS-guarantees

- Compensating for **variation in bandwidth**
- Insufficient bandwidth and no adaptation
  - => arbitrary data is lost (=> visual noise in video)
- Alternative
  - applications **adapt** to **changes** in **resource availability**
  - for continuous media streams: **adjust presentation quality**
- Basis for adaptation
  - Change the stream so that less data needs to be transmitted

Video Scaling

- Temporal scaling
  - reduce frame rate
  - simplest for streams based on intra frame coding (e.g. Motion JPEG)
  - more complex for streams based on inter frame coding (delta compression) which are most modern encoding schemes
- Spatial scaling
  - reduce number of pixels in each frame in video-stream
  - (often) based on hierarchical encoding (e.g. JPEG and MPEG-2)
- Quality / SNR scaling (Frequency scaling)
  - filtering higher frequencies in video signal
  - implies loss of quality (i.e. loss of details)
- SVC extension of H264 AVC combines all of the above
- Amplitude scaling
  - reduce color depth for every pixel, e.g. in H.261 to achieve constant bandwidth
- Color space scaling
  - reduce resolution of color space
  - e.g. switch from color to grey scale

IN5020, IFI/UiO 35

35

Video Scaling Example

- Example: H.264 quality scaling

IN5020, IFI/UiO 36

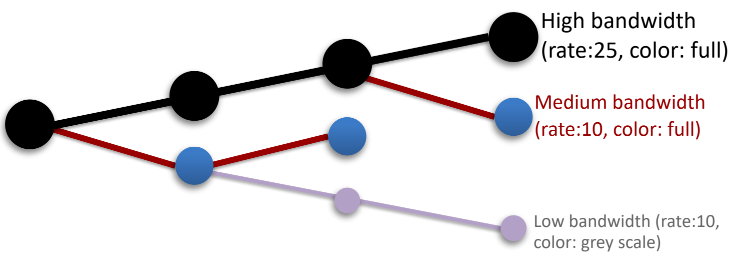
36

Continuous Media Distribution Services

- Usually overlay networks (on top of IP) designed with the aim providing QoS and delivering continuous media to many receivers in a cost effective way
  - Different content to many receivers (like in VoD)
    - **Content replication**: caching, mirroring (e.g. Akamai)
  - Same content to many receivers (like in broadcast)
    - **Application level multicast** (IP multicast not ubiquitous)
    - **P2P streaming** (CoolStreaming)
  - Same content to heterogeneous receivers
    - Adjust to resource poorest receiver, or
    - (Overlay) **Network filtering** (based on media scaling)

Using Network Filtering

- Filtering in network (e.g. using overlay). Example:
  - Distribution-tree with filtering, adapting QoS to each receiver
  - applies scaling in every relevant node in path from sender to receiver



- Network filtering can also be used in P2P streaming

Summary

- Multimedia applications require mechanisms that enable them to handle large amounts of time dependent data (i.e., continuous media)
- Most important mechanism: QoS management
- QoS is a question of resource management
- Resource management implies
  - admission control
  - scheduling function
- When resources can not be reserved, adaptation (media scaling & rate control) is the (only) alternative