**B.E 4/4 IT I SEM (A,B) Internal -II**

Subject : Distributed Systems Subject Code: BIT-406

Academic Year: 2016-2017 Date:

**Course Outcome Mapping Table**

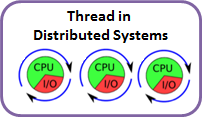
|  |  |  |
| --- | --- | --- |
| **Unit** | **Course Out Come** | **Question** |
| 1 | 1 | - |
| 2 | 2 | - |
| 3 | 3 | 1,4a,4b |
| 4 | 4 | 2,5 |
| 5 | 5 | 3,6 |

**Part – A**

**Answer the following questions (2\*3=6)**

1. Write about the Threads in Distributed Systems.

[Thread in Distributed Systems](http://lycog.com/distributed-systems/thread-distributed-systems/)

[](http://lycog.com/wp-content/uploads/2011/03/thread-in-distributed-systems.png)

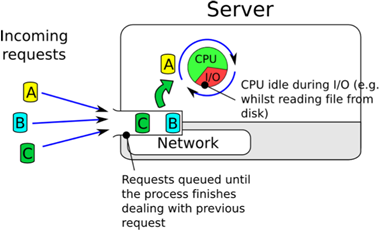
Thread in Distributed Systems

One challenge in distributed systems is concurrency. Without supporting multithread, distributed systems is almost meaningless as it needs to support multiple users. It is typical in distributed systems to have tasks that are partly CPU intensive and partly I/O bound. Generating database query, rendering an image, etc are examples of CPU intensive. Sending files or requests over the network are I/O intensive related to Socket, RMI, etc.

Now we will lookup 3 main architectures that thread impact the performance of distributed systems.

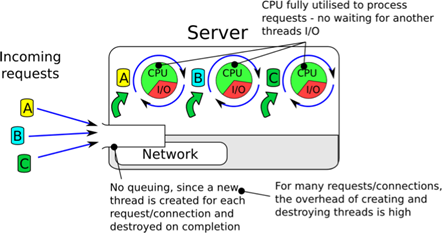
* Single thread
* Thread per request or connection
* Thread pool

**Single Thread**

[](http://lycog.com/wp-content/uploads/2011/03/single-thread-server.png)

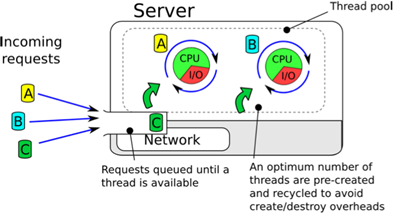
Single Thread

**Thread per request or connection**

[](http://lycog.com/wp-content/uploads/2011/03/thread-per-request.png)

Thread per Request or Connection

**Thread pool**

[](http://lycog.com/wp-content/uploads/2011/03/thread-pool.png)

Thread Pool

2. Explain about globe.

**GLOBE**

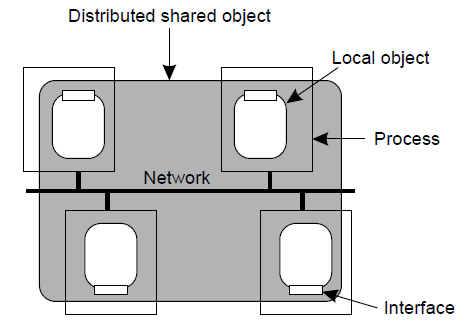
1. Experimental wide-area system currently being developed at Vrije Universiteit

2. Unique for its focus on scalability by means of truly distributed objects.

3. Prototype version up and running across multiplemachines distributed in NL and across Europe and the US.

**Object Model**

**Essence:** A Globe object is a **physically distributed shared object**: the object’s state may be physically distributed across several machines



**Local object:** A non-distributed object residing a single address space, often representing a distributed shared object

**Contact point:** A point where clients can contact the distributed object; each contact point is described through a **contact address**

3. What are the characteristics of multimedia data?

**CHARACTERISTICS OF TYPICAL MULTIMEDIA STREAMS**

It is likely that multimedia applications will remain in the window of scarcity (Say resources sufficient or not) for the foreseeable future. Advances in system performance are likely to be used to improve the quality of multimedia data, to include higher frame rates and greater resolution for video streams or to support many media streams concurrently, for example in **a video conferencing** system. More demanding applications, including virtual reality and real-time stream manipulation (“special effects”) can extend the window of scarcity almost indefinitely.

The term ‘continuous’ refers to the **user’s view of the data**. Internally, continuous media are represented as sequences of discrete values which replace each other over time. For example, the value of an image array is replaced 25 times per second to give the impression of a TV-quality view of a moving scene; a sound amplitude value is replaced 8000 times per second to convey telephone quality speech.

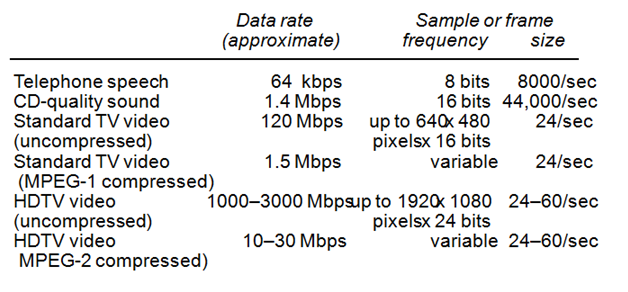
Multimedia streams are said to be time-based (or isochronous) because timed data elements in audio and video streams define the semantics or ‘content’ of the stream. The times at which the values are played or recorded affect the validity of the data. Hence systems that support multimedia applications need to preserve the timing when they handle continuous data.

Compression can reduce bandwidth requirements by factors between 10 and 100, but the timing requirements of continuous data are unaffected. Various compressed data formats such as GIF, TIFF and JPEG for still images and MPEG-1, MPEG-2 and MPEG-4 for video sequences.

For Compression we use of special-purpose hardware to process and despatch video and audio information – the video and audio coders/decoders (Codecs) found on video cards manufactured for personal computers. The compression method used for the MPEG video formats is asymmetric, with a complex compression algorithm and simpler decompression.

**The window of scarcity for computing and communication resources**





**TYPICAL INFRASTRUCTURE COMPONENTS FOR MULTIMEDIA APPLICATIONS**



**Part – B**

**Answer any two questions (2\*7=14)**

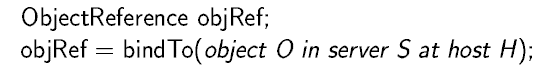
4. a. Discuss about the implementation of namespace.

**NAMING**

In CORBA, it is essential to distinguish specification-level and implementation-level object References

**Specification level:** An object reference is considered to be the same as a proxy for the referenced object having an object reference means you can directly invoke methods; there is no separate client to- object binding phase

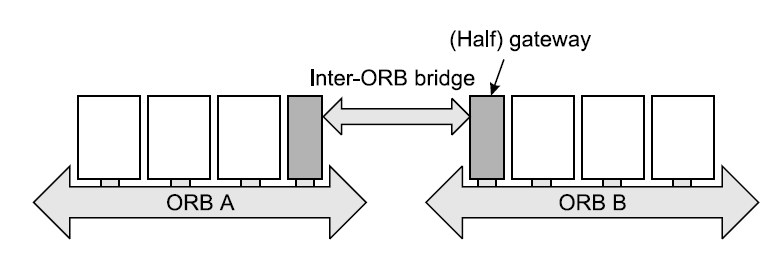
**Implementation level:** When a client gets an object reference, the implementation ensures that, one way or the other, a proxy for the referenced object is placed in the client address space.



**Conclusion:** Object references in CORBA used to be highly **implementation dependent**: different implementations of CORBA could normally not exchange their references.

**Interoperable Object References**

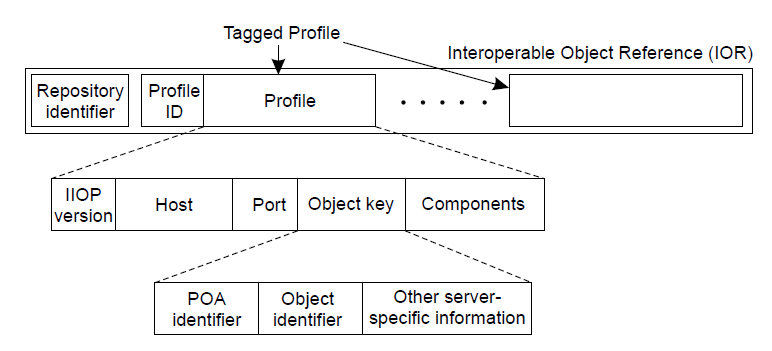
**Observation:** Recognizing that object references are implementation dependent, we need a separate referencing mechanism to cross ORB boundaries



**Solution:** Object references passed from one ORB to another are transformed by the bridge through which they pass (different transformation schemes can be implemented)

**Observation:** Passing an object reference from ORB A to ORB B circumventing the A-to-B bridge may be useless if ORB B doesn’t understand ref A.

**Observation:** To allow all kinds of *different* systems to communicate, we standardize the reference that is passed between bridges:



**NAMING SERVICE**

**Essence:** CORBA’s naming service allows servers to associate a name to an object reference, and have clients subsequently bind to that object by resolving its name

**Observation:** In most CORBA implementations, object references denote servers at specific hosts; naming makes it easier to relocate objects

**Observation:** In the naming graph all nodes are objects; there are no restrictions to binding names to objects 􀀀 CORBA allows arbitrary naming graphs

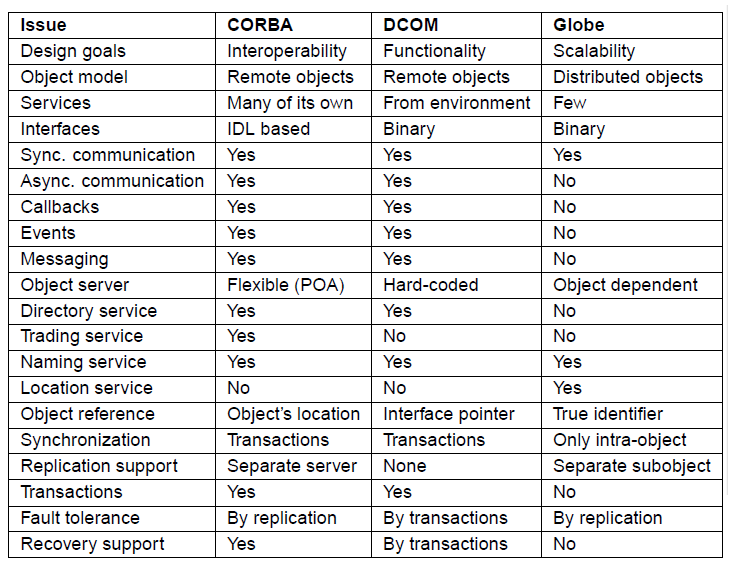
**Question:** How do you imagine cyclic name resolution stops?

**Observation:** There is no single root; an initial context node is returned through a special call to the ORB. Also: the naming service can operate *across* different ORBs🡺 **interoperable naming service**

b. Explain about stream synchronization.



5. Explain about the differences between CORBA, DCOM and GLOBE.



6. Explain about Quality of Service Management.

**QUALITY OF SERVICE MANAGEMENT**

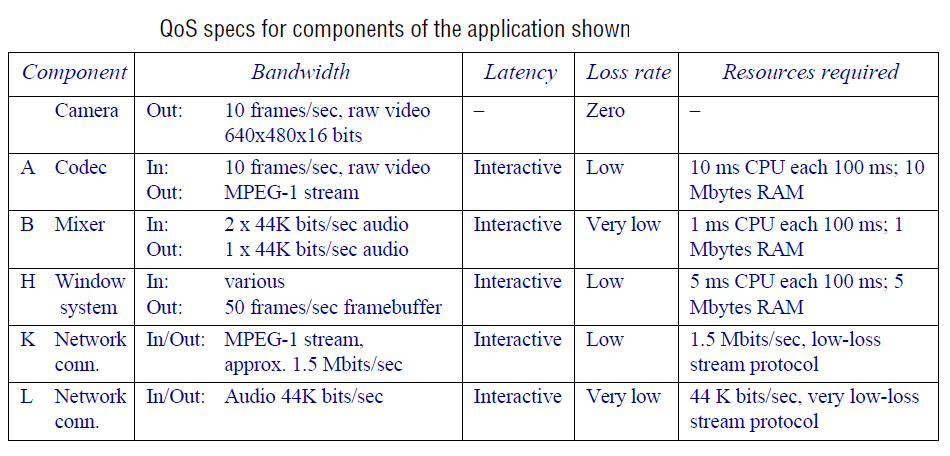
The management and allocation of resources to provide such guarantees is referred to as **quality of service management.**

**QOS MANAGER** is a system component of **QUALITY OF SERVICE MANAGEMENT** responsible for the allocation and scheduling of those resources.

When multimedia applications run in networks of personal computers they compete for resources at the workstations running the applications (processor cycles, bus cycles, buffer capacity) and in the networks (physical transmission links, switches, gateways). Workstations and networks may have to support several multimedia and conventional applications. There is competition between the multimedia and conventional applications, between different multimedia applications and even between the media streams within individual applications.

The concurrent use of physical resources for a variety of tasks has long been possible with multi-tasking operating systems and shared networks. In multi-tasking operating systems the central processor is allocated to individual tasks (or processes) in a round-robin or other scheduling scheme that shares the processing resources on a best-efforts basis amongst all of the tasks currently competing for the central processor.

Collisions can occur and when they do sending nodes wait for random back off periods in order to prevent repeated collisions.



**QOS MANAGER’S TWO MAIN SUB-TASKS:**

1. Quality of service negotiation. 2. Admission control.

**QUALITY OF SERVICE NEGOTIATION**

The application indicates its resource requirements to the QoS manager. The QoS manager evaluates the feasibility of meeting the requirements against a database of the available resources and current resource commitments and gives a positive or negative response. If it is negative, the application may be reconfigured to use reduced resources and the process is repeated.

To negotiate QoS between an application and its underlying system, an application must specify its QoS requirements to the QoS Manager. This is done by the transmission of a set of parameters. Three parameters are of primary interest when it comes to processing and transporting multimedia streams: bandwidth, latency, and loss rate.

**BANDWIDTH:** It of a multimedia stream or component is the rate at which data flows through it. Bandwidth is the characterization of burstiness.

**LATENCY:** It is the time required for an individual data element to move through a stream from the source to the destination. Of course this may vary depending on the volume of other data in the system and other characteristics of the system load. This variation is termed jitter – formally, jitter is the first derivative of the latency.

**LOSS RATE:** Since the late delivery of multimedia data is of no value, data elements will be dropped when it is impossible to deliver them before their scheduled delivery time. To avoid the data loss we either use leaky bucket or token bucket algorithm.

**TRAFFIC SHAPING ALGORITHMS**



**NOTE:**

The model of linear-bounded arrival processes (LBAP) used in defines the maximum number of messages in a stream during any time interval t as Rt + B where R is the rate and B is the maximum size of burst. The burst parameter defines the amount of buffer space required to avoid loss.

**ADMISSION CONTROL**

If the result of the resource evaluation is positive, the requested resources are reserved and the application is given a Resource Contract, stating the resources that have been reserved. The contract includes a time limit. The application is then free to run. If it changes its resource requirements it must notify the QoS Manager. If the requirements decrease, the resources released are returned to the database as available resources. If they increase, a new round of

negotiation and admission control is initiated.

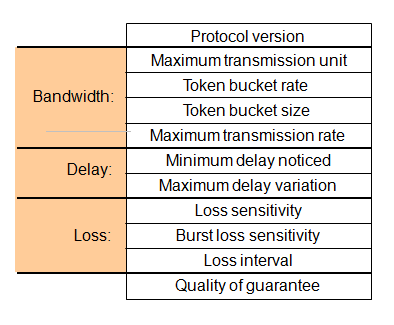
**TRAFFIC SHAPING**

Traffic shaping is the term used to describe the use of output buffering to smooth the flow of data elements. The bandwidth parameter of a multimedia stream typically provides an idealistic approximation of the actual traffic pattern that will occur when the stream is transmitted. The closer the actual traffic pattern matches the description, the better a system will be able to handle the traffic, in particular when it uses scheduling methods that are designed for periodic requests.

The LBAP model of bandwidth variations calls for regulation of the burstiness of multimedia streams. Any stream can be regulated by inserting a buffer at the source and by defining a method by which data elements leave the buffer.

**NOTE:**

**For traffic shaping we user flow specification format (RFC 1363 Flow Spec.).**



**FLOW SPECIFICATIONS**

A collection of QoS parameters is typically known as a flow specification, or flow spec for short. Several examples of flow specs exist and are all similar. In Internet RFC 1363, a flow spec is defined as eleven 16-bit numeric values (Figure Above) that reflect the QoS parameters discussed above in the following way:

• The maximum transmission unit and maximum transmission rate determine the maximum bandwidth required by the stream.

• The token bucket size and rate determine the burstiness of the stream.

• The delay characteristics are specified by the minimum delay that an application can notice (since we wish to avoid over-optimization for short delays) and maximum jitter it can accept.

• The loss characteristics are defined by the total acceptable number of losses over a certain interval and the maximum number of consecutive losses.

**NEGOTIATION PROCEDURES**

For distributed multimedia applications, the components of a stream are likely to be located in several nodes. There will be a QoS manager at each node. A straightforward approach to QoS negotiation is to follow the flow of data along each stream from the source to the target. A source component initiates the negotiation by sending out a flow spec to its local QoS manager.

**ADMISSION CONTROL** regulates access to resources to avoid resource overload and to protect resources from requests that they cannot fulfil. It involves turning down service requests should the resource requirements of a new multimedia stream would violate existing QoS guarantees.

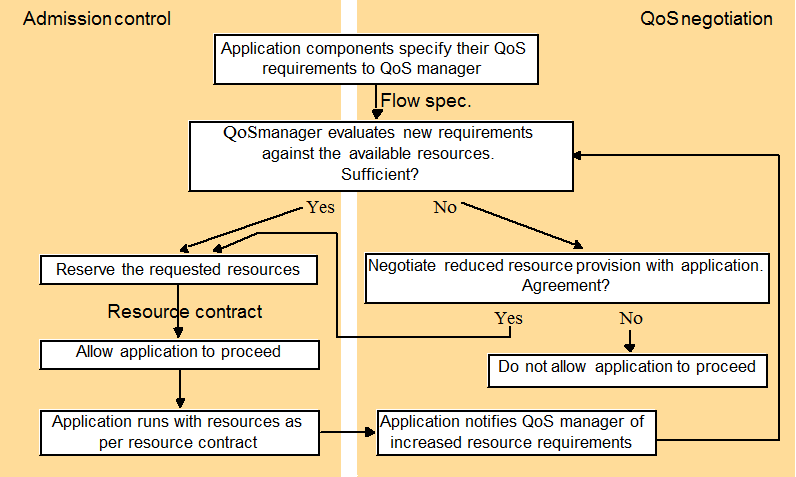
An admission control scheme is based on some knowledge of both the overall system capacity and the load generated by each application. The bandwidth requirement specification for an application may reflect the maximum amount of bandwidth that an application will ever require, the minimum bandwidth it will need to function, or some average value in between. Correspondingly, an admission control scheme may base its resource allocation on any of these values.

For resources that have a single allocator, admission control is straightforward. Resources that have distributed access points, such as many local area networks, require either a centralized admission control entity or some distributed admission control algorithm that avoids conflicting concurrent admissions. Bus arbitration within workstations falls into this category – however, even multimedia systems that perform bandwidth allocation extensively do not control bus admission as bus bandwidth is not considered to be in the window of scarcity.

**STATISTICAL MULTIPLEXING** Because of the potential under-utilization that can occur, it is common to overbook resources. The resulting guarantees, often called statistical or soft

guarantees to distinguish them from the deterministic or hard guarantees introduced before, are only valid with some (usually very high) probability. Statistical guarantees tend to provide better resource utilization as they do not consider the worst case.

Statistical multiplexing is based on the hypothesis that for a large number of streams the aggregate bandwidth required remains nearly constant regardless of the bandwidth of individual streams.



**RESOURCE MANAGEMENT**

To provide a certain QoS level to an application, not only does a system need to have sufficient resources (performance), it also needs to make these resources available to an application when they are needed (scheduling).

**RESOURCE SCHEDULING**

Processes need to have resources assigned to them according to their priority. A resource scheduler determines the priority of processes based on certain criteria. Traditional CPU

schedulers in time-sharing systems often base their priority assignments on responsiveness and fairness: I/O intensive tasks get high priority to guarantee fast response to user requests, CPU-bound tasks get lower priorities and overall, processes in the same class are treated equally.

**RESOURCE SCHEDULING TYPES**

1. Fair scheduling. 2. Real-time scheduling.

**FAIR SCHEDULING** If several streams compete for the same resource it becomes necessary

To consider fairness and to prevent ill-behaved streams from taking too much bandwidth. A straightforward approach to ensure fairness is to apply round-robin scheduling to all streams in the same class. Whereas in [Nagle 1987] such a method was introduced on a packet-by-packet basis, in [Demers et al. 1989] the method is used on a bit-by-bit basis which provides more fairness with respect to varying packet sizes and packet arrival times. These methods are known as fair queuing. It is done by bit by bit or packet by packet.

All basic round-robin schemes assign the same bandwidth to each stream. To take the individual bandwidth of streams into account, the bit-by-bit scheme can be extended so that for certain streams a larger number of bits can be transmitted per cycle. This method is called weighted fair queuing.

**REAL-TIME SCHEDULING** Traditional real-time scheduling methods suit the model of regular continuous multimedia streams very well. Earliest-deadline-first (EDF) scheduling has almost become a synonym for these methods. An EDF scheduler uses a deadline that is associated with each of its work items to determine the next item to be processed: the item with the earliest deadline goes first. In multimedia applications, we identify each media element arriving at a process as a work item. EDF scheduling is proven to be optimal for allocating a single resource based on timing criteria: if there is a schedule that fulfils all timing requirements, EDF scheduling will find it.

**RATE-MONOTONIC (RM) SCHEDULING** is a prominent technique for real-time scheduling of periodic processes that achieves just this. RM scheduling has been shown to be optimal for situations that only utilize a given bandwidth by less than 69% . Using such an allocation scheme, the remaining bandwidth could be given to non-real-time applications.

**STREAM ADAPTATION**

Whenever a certain QoS cannot be guaranteed or can only be guaranteed with a certain

probability, an application needs to adapt to changing QoS levels, adjusting its performance accordingly. For continuous-media streams, the adjustment translates into different levels of media presentation quality.

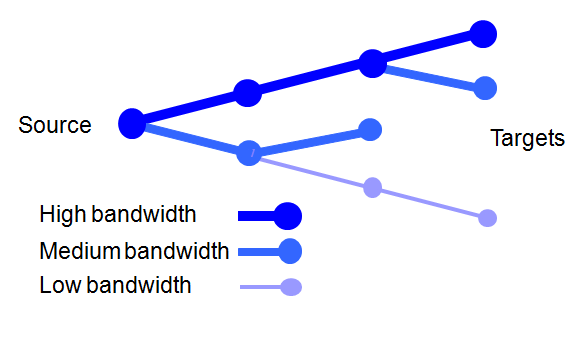
The simplest form of adjustment is to drop pieces of information. This is easily done in audio streams where samples are independent from each other, but it can immediately be noticed by the listener. Drop-outs in a video stream encoded in Motion JPEG, where each frame stands for itself are more tolerable. Encoding mechanisms such as MPEG, where the interpretation of a frame depends on the values of several adjacent frames, are less robust against omissions: it takes a longer time to recover from errors and the encoding mechanism may, in fact, amplify errors.

For non-interactive applications this may be acceptable, although it can eventually lead to buffer overflows as data is accumulated between the source and sink. For conferencing and other interactive applications, increasing delays are not acceptable, or must exist only for a short period.

**SCALING (RESIZING)**

If adaptation is performed at the target of a stream, the load on any bottleneck in the system is not decreased and the overload situation persists. It is useful to adapt a stream to the bandwidth available in the system before it enters a bottleneck resource in order to resolve contention. This is known as scaling. Scaling algorithms are media-dependent.

**FILTERING**



It can also be achieved by dropping a channel in a stereo transmission.

**Scaling methods can work at different granularities for video**

**1. TEMPORAL SCALING** reduces the resolution of the video stream in the time domain by decreasing the number of video frames transmitted within an interval. Temporal scaling is best suited for video streams in which individual frames are selfcontained and can be accessed independently. Delta compression techniques are more difficult to handle as not all frames can be easily dropped. Hence, temporal scaling is more suitable for Motion JPEG than for MPEG streams.

**2. SPATIAL SCALING** reduces the number of pixels of each image in a video stream. For spatial scaling, hierarchical arrangement is ideal because the compressed video is immediately available in various resolutions. Therefore, the video can be transferred over the network using different resolutions without recoding each picture before finally transmitting it. JPEG and MPEG-2 support different spatial resolutions of images and are well-suited for this kind of scaling.

**3. FREQUENCY SCALING** modifies the compression algorithm applied to an image. This result in some loss of quality, but in a typical picture, compression can be increased significantly before a reduction of image quality becomes visible.

**4. AMPLITUDINAL SCALING** reduces the colour depths for each image pixel. This scaling method is, in fact, used in H.261 encodings to arrive at a constant throughput although image content varies.

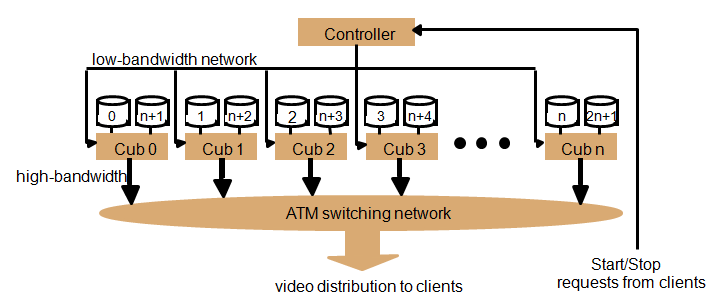
**5. COLOUR SPACE SCALING** reduces the number of entries in the colour space. One way to realize colour space scaling is to switch from colour to greyscale presentation. Obviously, combinations of these scaling methods are possible. A system to perform scaling consists of a monitor process at the target side and a scaler process at the source. The monitor keeps track of the arrival times of messages. When messages get delayed, it is an indication of some bottleneck in the system. The monitor then sends a Scale-Down message to the source and it reduces the bandwidth of the stream. After some period of time, the source scales the stream up again. Should the bottleneck still exist, the monitor will again detect a delay and scale the stream down 1993]. The fundamental problem of the scaling approach is to find good heuristics to avoid unnecessary Scale-Up operations and to prevent the system from oscillating.

**FILTERING**

As scaling modifies a stream at the source, it is not always suitable for applications that involve several receivers: when a bottleneck occurs on the route to one target, this target sends a Scale-Down message to the source and all targets receive the degraded quality although some would have no problem in handling the original stream.

Filtering is a method that provides the best possible QoS to each target by applying scaling at each relevant node on the path from the source to the target (Figure 15.9). RSVP is an example of a QoS negotiation protocol that supports filtering. Filtering requires that a stream can be partitioned into a set of hierarchical substreams, each adding a higher level of quality. The capacity of nodes on a path determines the number of sub-streams a target receives. All other sub-streams are filtered out as close to the source as possible (perhaps even at the source) to avoid transfer of data that is later thrown away. A sub-stream is not filtered at an intermediate node if somewhere downstream a path exists that can carry the entire sub-stream.

**TIGER VIDEO FILE SERVER HARDWARE CONFIGURATION**



**TIGER SCHEDULE**