LUENTO 23

Booting, multimedia systems

Exam issues

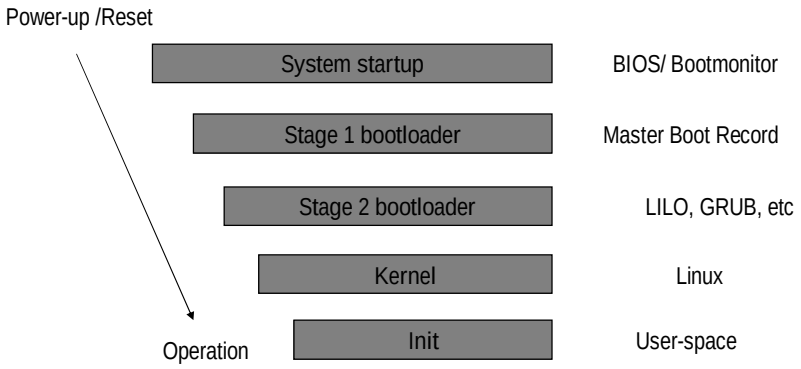
1

Booting

- Key problem: How do you initiate a system using only itself?
- Booting <- Bootstraping
- Bootstraping - why this term?
  - Baron Munchausen pulled himself out of swamp using his boot straps

2

View of the Linux boot process



Source: http://www-128.ibm.com/developerworks/library/l-linuxboot/index.html 3

BIOS tasks

- BIOS – Basic Input / Output System
  - BIOS refers to the firmware code run by a personal computer when first powered on.
  - BIOS can also be said to be a coded program embedded on a chip that recognizes and controls various devices that make up x86 personal computers. (source: wikipedia)
- Execution starts from a fixed location (on x386 that location is 0xFFFF0000)
- Check hardware, locate the boot device
  - Disk, CD, ...
- Load the first 'loader' from Master Boot Record

4

BIOS example sequence

1. Check the CMOS Setup for custom settings
2. Load the interrupt handlers and device drivers
3. Initialize registers and power management
4. Perform the power-on self-test (POST)
5. Display system settings
6. Determine which devices are bootable
7. Initiate the bootstrap sequence

5

BIOS loading the boot loader – pseudocode example

- 0: set the P register to 8
- 1: check paper tape reader ready
- 2: if not ready, jump to 1
- 3: read a byte from paper tape reader to accumulator
- 4: if end of tape, jump to 8
- 5: store accumulator to address in P register
- 6: increment the P register
- 7: jump to 1

6

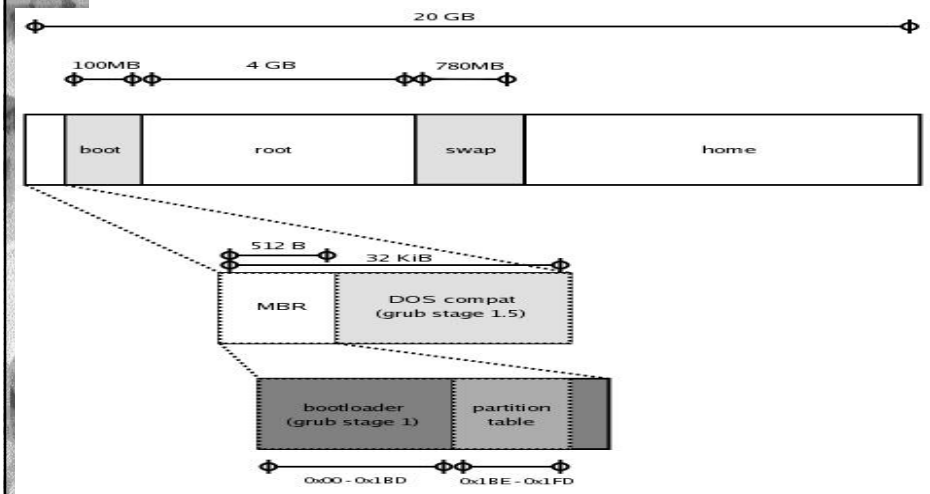
Boot loader

- Linux: Lilo, GRUB
- Windows NT: NTLDR
- Initial location on primary disk, for example: Master Boot Record

- Main task: Load and start operating system
- Dual boot: offer menu and start the chosen one

7

GRUB on disk



http://www.pixelbeat.org/docs/disk/

8

GRUB - GRand Unified Bootloader

- GRUB is independent of any particular operating system and may be thought of as a tiny, function-specific OS.
- It's primary task is to load the actual operating system and pass control to it.
- GRUB is large service:
  - It does not fit to MBR: needs to be handled in two phases
  - Phase 1 (loaded from MBR by BIOS) load phases 1.5 and 2
- A lot of features: file systems, command line interface, chain-loading other boot loaders, ...

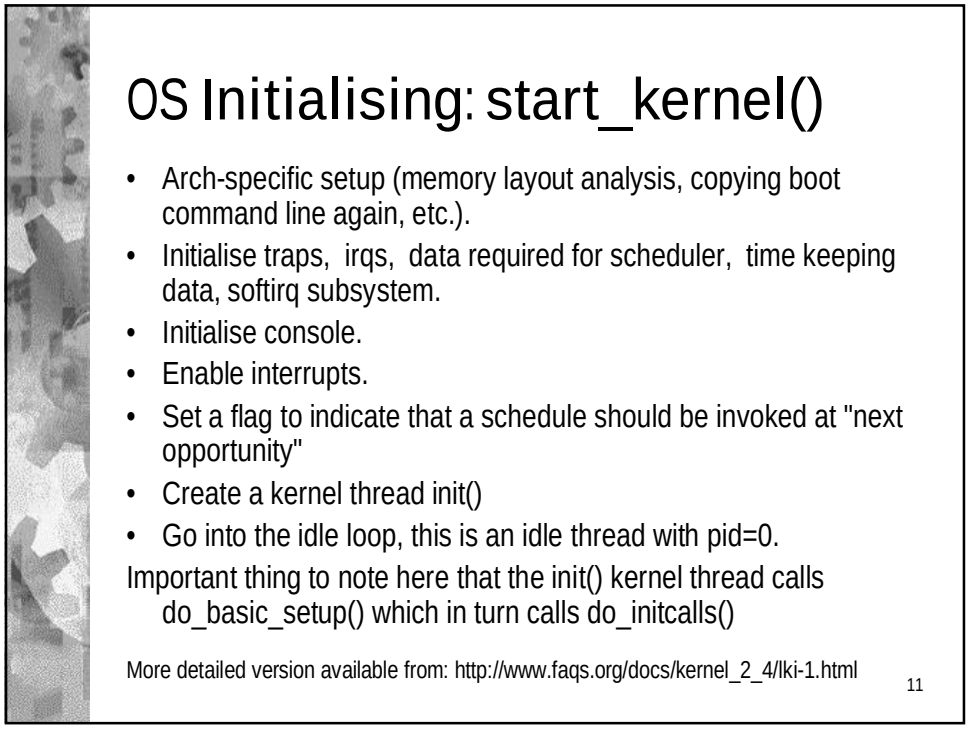
9

Loading Linux

- Bootsector loads setup, decompression routines and compressed kernel image.
- The kernel is uncompressed in protected mode.
- Low-level initialisation is performed by asm code arch/i386/kernel/head.S:
  - Initialise segment values and page tables.
  - Enable paging by setting PG bit in %cr0.
  - Copy the first 2k of bootup parameters (kernel commandline).
  - The first CPU calls start_kernel(), all others call initialize_secondary().
- High-level C initialisation

Source: http://www.faqs.org/docs/kernel_2_4/lki-1.html

10



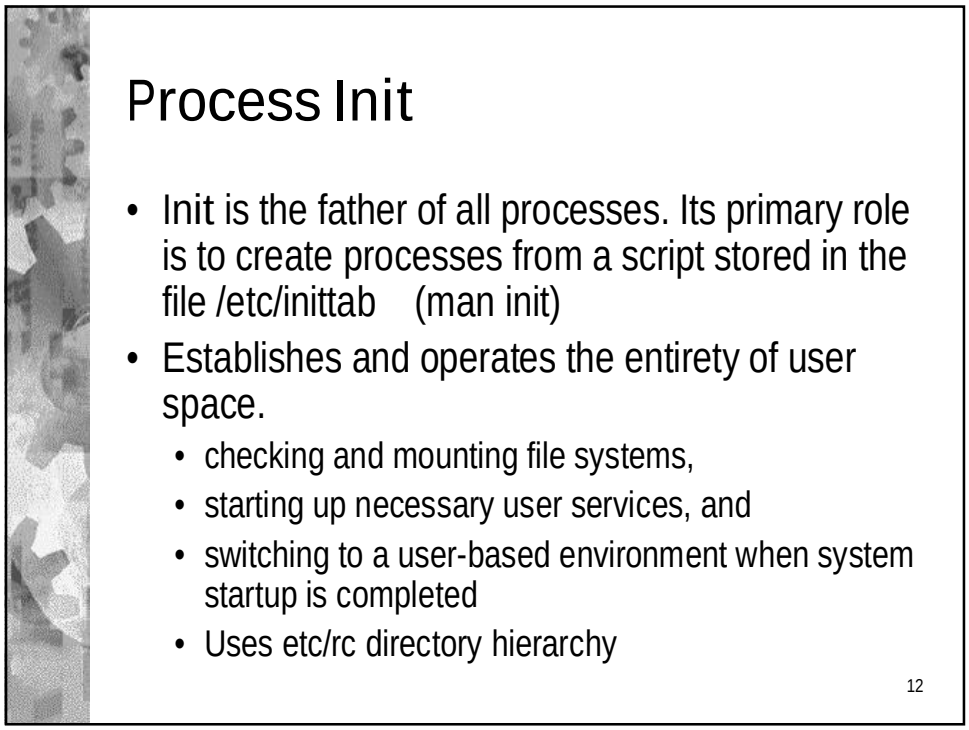
OS Initialising: start_kernel()

- Arch-specific setup (memory layout analysis, copying boot command line again, etc.).
- Initialise traps, irqs, data required for scheduler, time keeping data, softirq subsystem.
- Initialise console.
- Enable interrupts.
- Set a flag to indicate that a schedule should be invoked at "next opportunity"
- Create a kernel thread init()
- Go into the idle loop, this is an idle thread with pid=0.

Important thing to note here that the init() kernel thread calls do_basic_setup() which in turn calls do_initcalls()

More detailed version available from: http://www.faqs.org/docs/kernel_2_4/lki-1.html

11



Process Init

- Init is the father of all processes. Its primary role is to create processes from a script stored in the file /etc/inittab (man init)
- Establishes and operates the entirety of user space.
  - checking and mounting file systems,
  - starting up necessary user services, and
  - switching to a user-based environment when system startup is completed
  - Uses etc/rc directory hierarchy

12

etc/rc.d

- debian: run /etc/init.d/rcS which runs:
- /etc/rcS.d/S* scripts
- /etc/rc.boot/* (depreciated)
- run programs specified in /etc/inittab

- Scripts in /etc/rc*.d/* are symlinks to /etc/init.d
- Scripts prefixed with S will be started when the runlevel is entered, eg /etc/rc5.d/S99xdm
- Scripts prefixed with K will be killed when the runlevel is entered, eg /etc/rc6.d/K20apache
- Executed in numerical order

13

Runlevel	Scripts Directory (Red Hat/Fedora Core)	State
0	/etc/rc.d/rc0.d/	Shutdown/halt system
1	/etc/rc.d/rc1.d/	Single user mode
2	/etc/rc.d/rc2.d/	Multiuser with no network services exported
3	/etc/rc.d/rc3.d/	Default text/console only start. Full multiuser
4	/etc/rc.d/rc4.d/	Reserved for local use. Also X-windows (Slackware/BSD)
5	/etc/rc.d/rc5.d/	XDM X-windows GUI mode (Redhat/System V)
6	/etc/rc.d/rc6.d/	Reboot
s or S		Single user/Maintenance mode (Slackware)
M		Multiuser mode (Slackware)

http://www.yolinux.com/TUTORIALS/LinuxTutorialInitProcess.html

14

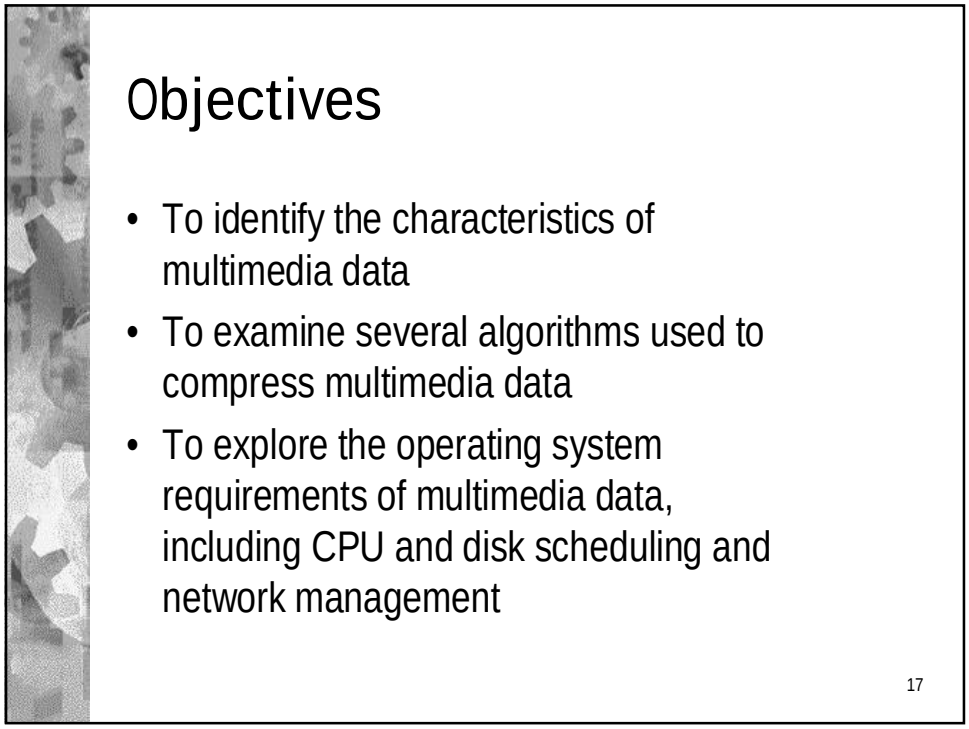
Windows NT

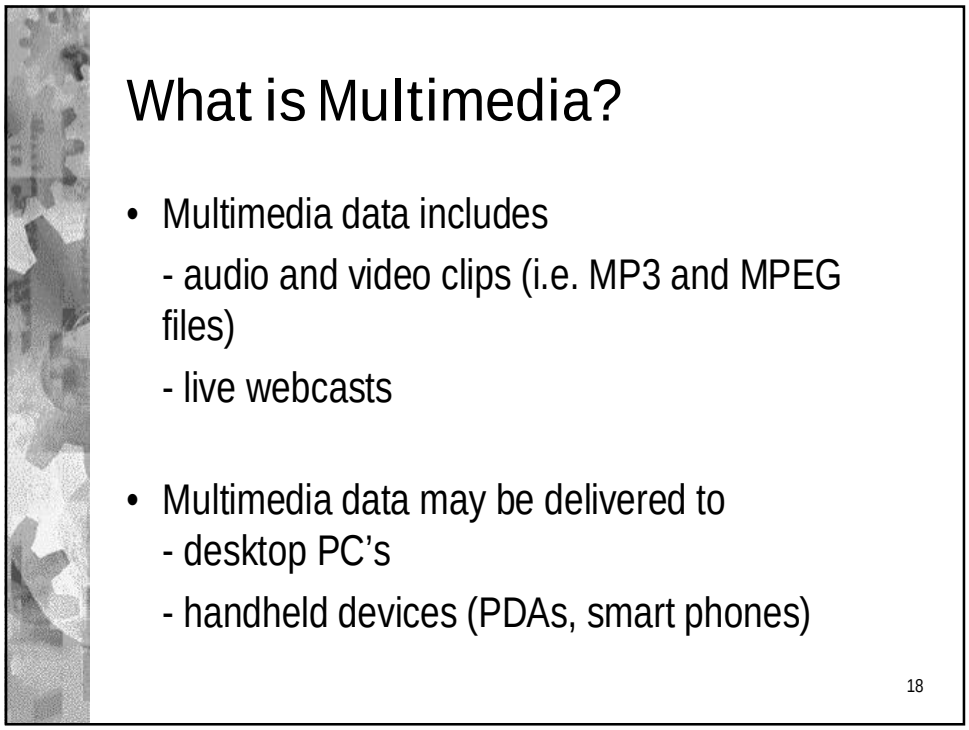
- Power-on self test (POST) phase
- Initial startup phase
- Boot loader phase: NTLDR
  - Startup phase: create interrupt descriptor table, page tables and enable paging
- Detect and configure hardware phase
  - osloader.exe: knows file systems,
    - read boot.ini, check if hibernated,
    - Load Ntoskrnl.ece, hal.dll (hardware abstraction)
    - Load boot drivers (hard drive, file system, etc)
- Kernel loading phase
  - Interrupt controller, memory manager, object manager, process manager, System idle process
  - Load and initialize system drivers
- Logon phase
  - Start Session manager subsystem

15

Multimediajärjestelmät

16


Objectives
• To identify the characteristics of multimedia data
• To examine several algorithms used to compress multimedia data
• To explore the operating system requirements of multimedia data, including CPU and disk scheduling and network management
17


What is Multimedia?
• Multimedia data includes
- audio and video clips (i.e. MP3 and MPEG files)
- live webcasts
• Multimedia data may be delivered to
- desktop PC's
- handheld devices (PDAs, smart phones)
18

Media Delivery

- Multimedia data is stored in the file system like other ordinary data.
- However, multimedia data must be accessed with specific timing requirements.
- For example, video must be displayed at 24-30 frames per second. Multimedia video data must be delivered at a rate which guarantees 24-30 frames/second.
- Continuous-media data is data with specific rate requirements.

19

Streaming

- Streaming is delivering a multimedia file from a server to a client - typically the deliver occurs over a network connection.
- There are two different types of streaming:

  1. Progressive download - the client begins playback of the multimedia file as it is delivered. The file is ultimately stored on the client computer.

  2. Real-time streaming - the multimedia file is delivered to - but not stored on - the client's computer.

20

Real-time Streaming

- There are two types of real-time streaming:

  (1) Live streaming - used to deliver a live event while it is occurring.

  (2) On-demand streaming - used to deliver media streams such as movies, archived lectures, etc. The events are not delivered in real-time.

21

Multimedia Systems Characteristics

- Multimedia files can be quite large.
- Continuous media data may require very high data rates.
- Multimedia applications may be sensitive to timing delays during playback of the media.

22

Compression

- Because of the size and rate requirements of multimedia systems, multimedia files are often compressed into a smaller form.
- MPEG Compression:

  (1) MPEG-1 - 352 X 240 @ 30 frames/second

  (2) MPEG-2 - Used for compressing DVD and high-definition television (HDTV)

  (3) MPEG-4 - Used to transmit audio, video, and graphics. Can be delivered over very slow connections (56 Kbps)

23

Operating Systems Issues

- The operating system must guarantee the specific data rate and timing requirements.
- Such requirements are generally known as Quality-of-Service (QoS) guarantees.
- Guaranteeing QoS has effects in a computer system:

  (1) CPU processing

  (2) Scheduling

  (3) File systems

  (4) Network protocols

24

Requirement of Multimedia Operating Systems

- There are three levels of QoS

  (1) Best-effort service - the system makes a best effort with no QoS guarantees.

  (2) Soft QoS - allows different traffic streams to be prioritized, however no QoS guarantees are made.

  (3) Hard QoS - the QoS rquirements are guaranteed.

25

Parameters Defining QoS

- Throughput - the total amount of work completed during a specific time interval.
- Delay - the elapsed time from when a request is first submitted to when the desired result is produced.
- Jitter - the delays that occur during playback of a stream.
- Reliability - how errors are handled during transmission and processing of continuous media.

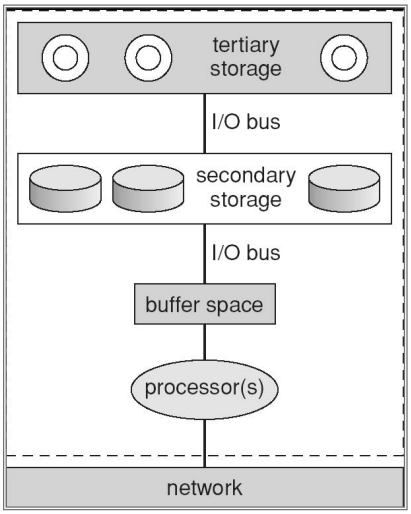
26

Further QoS Issues

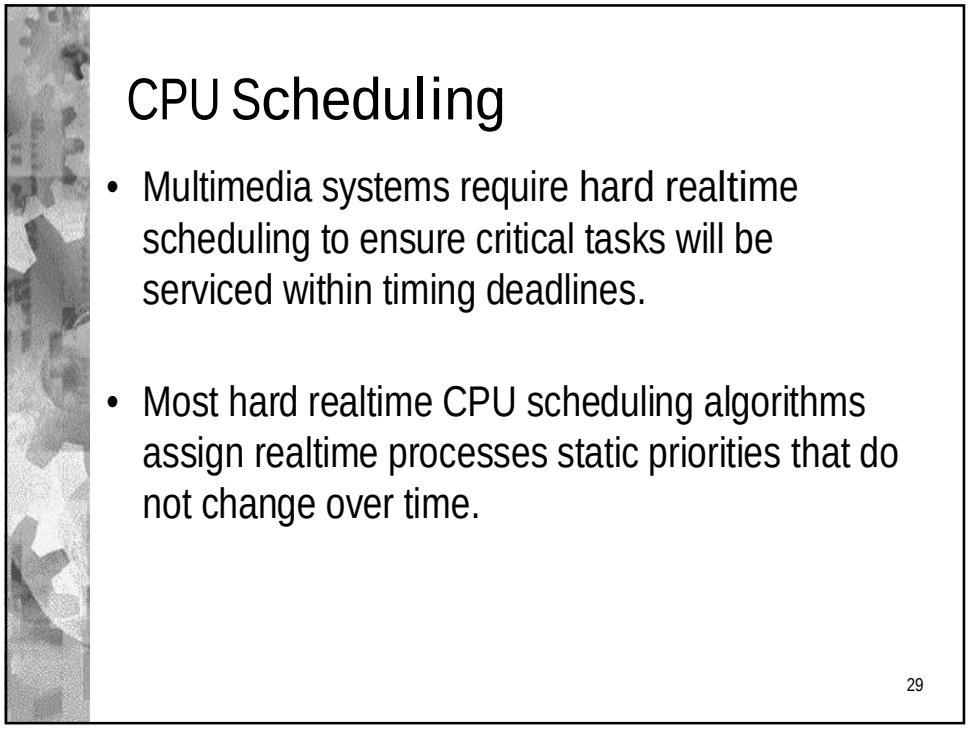
- QoS may be negotiated between the client and server.
- Operating systems often use an admission control algorithm that admits a request for a service only if the server has sufficient resources to satisfy the request.

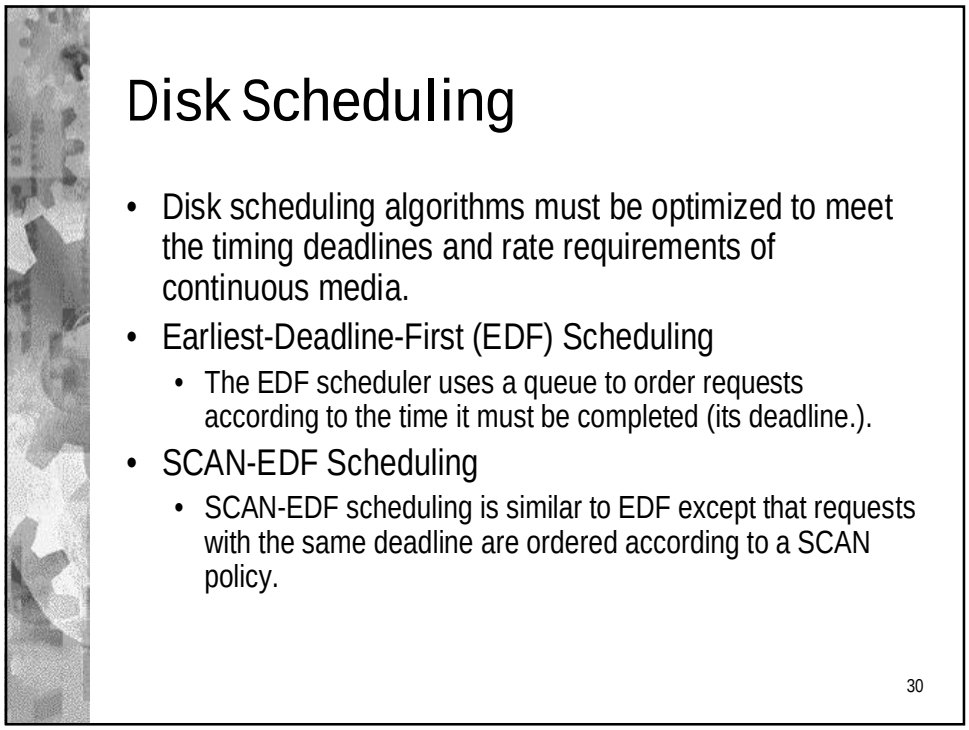
27

Figure 20.1 Resources on a file server



28


CPU Scheduling
• Multimedia systems require hard realtime scheduling to ensure critical tasks will be serviced within timing deadlines.
• Most hard realtime CPU scheduling algorithms assign realtime processes static priorities that do not change over time.
29


Disk Scheduling
• Disk scheduling algorithms must be optimized to meet the timing deadlines and rate requirements of continuous media.
• Earliest-Deadline-First (EDF) Scheduling
• The EDF scheduler uses a queue to order requests according to the time it must be completed (its deadline.).
• SCAN-EDF Scheduling
• SCAN-EDF scheduling is similar to EDF except that requests with the same deadline are ordered according to a SCAN policy.
30

Network Management

- Three general methods for delivering content from a server to a client across a network:

  (1) Unicasting - the server delivers the content to a single client.

  (2) Broadcasting - the server delivers the content to all clients, regardless whether they want the content or not.

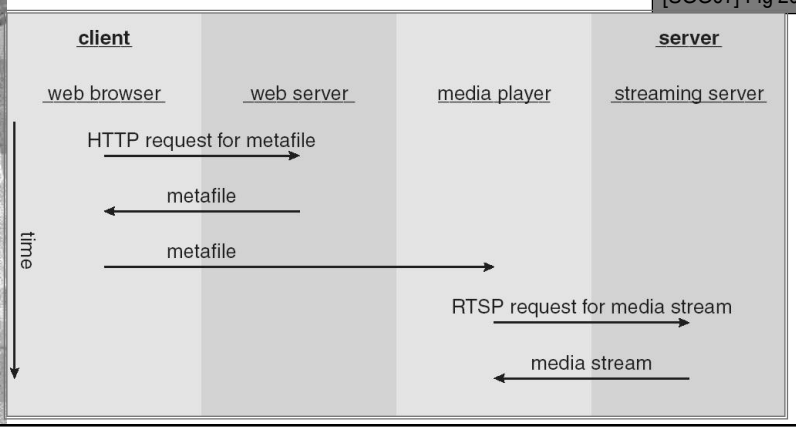
  (3) Multicasting - the server delivers the content to a group of receivers who indicate they wish to receive the content.

31

RealTime Streaming Protocol (RTSP)

- Standard HTTP is stateless whereby the server does not maintain the status of its connection with the client.

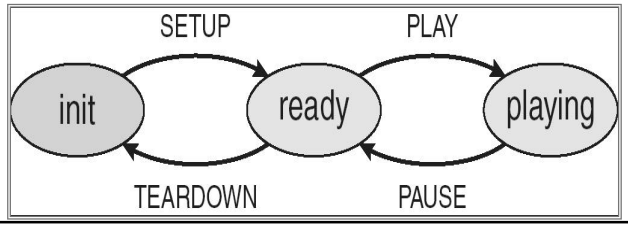
[SGG07] Fig 20.3



32

RTSP States

- SETUP - the server allocates resources for a client session.
- PLAY - the server delivers a stream to a client session.
- PAUSE - the server suspends delivery of a stream.
- TEARDOWN - the server breaks down the connection and releases the resources allocated for the session.



[SGG07] Fig 20.4

33

CineBlitz Multimedia Server

- CineBlitz supports both realtime and non-realtime clients.
- CineBlitz provides hard QoS guarantees to realtime clients using an admission control algorithm.
- The disk scheduler orders requests using C-SCAN order.

34

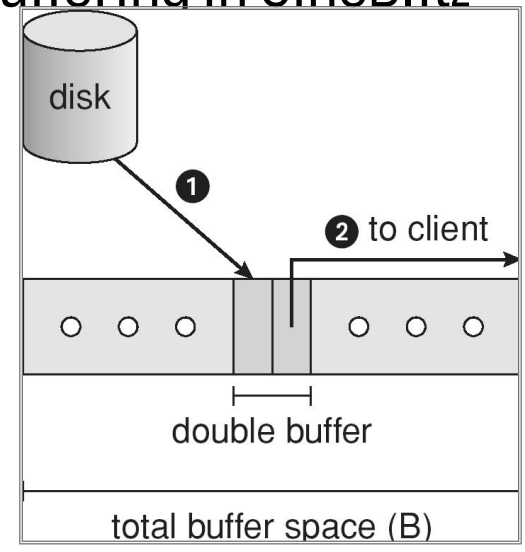
CineBlitz Admission Controller

- Total buffer space required for *N* clients where client has rate requirement of r_i

$$\sum_{i=1}^{N} 2 \times T \times r_i \leq B.$$

35

Figure 20.05 Double buffering in CineBlitz



36

LUENTO 24

2. Course exam

Thursday 13.12 klo 16.00 A111
Exam covers:
- Book chapters 9-16 (not 13) + Appendix B.4
- Lectures 11-24, exercises 7-12
- Team task 3

37

Scheduling

38

When?

- Long-term
  - Accept new process?
  - Enough memory? swap?
- Medium-term
  - Bring a suspended process back, when?
  - Enough free memory?
  - MPL, multiprocessing level?
- Short-term
  - Which process is switched to processor?
- I/O
  - Service order of I/O requests (of processes) ?

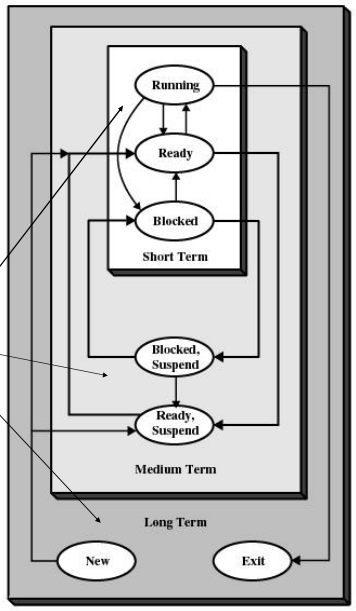
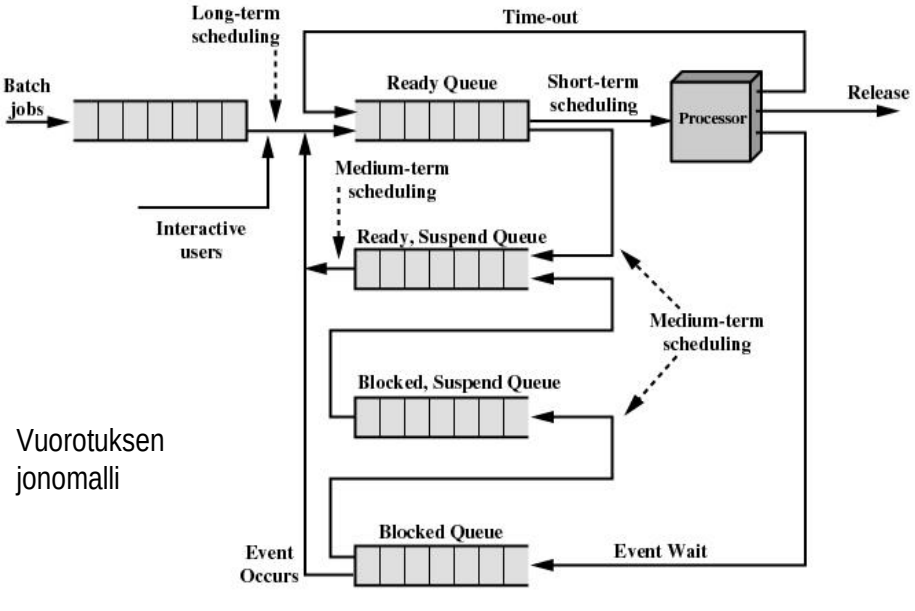


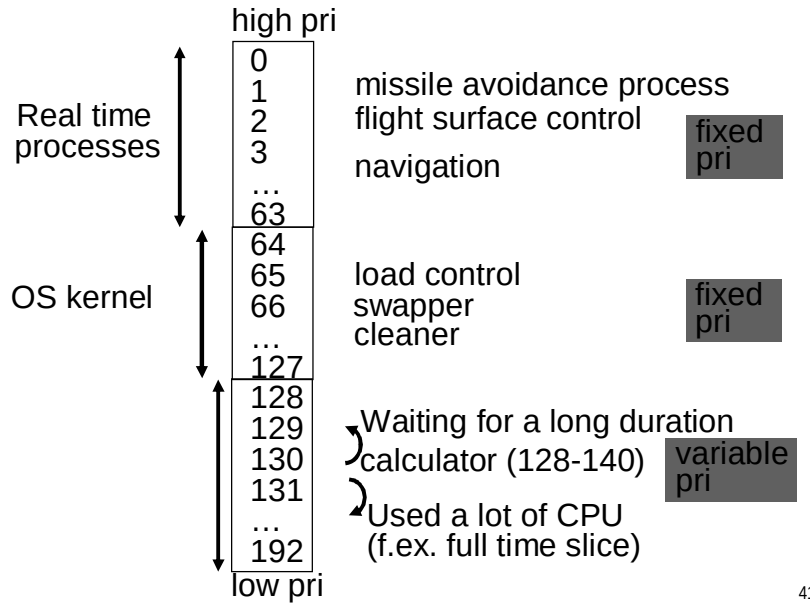
Figure 9.2 Levels of Scheduling



Vuorotuksen jonomalli

Figure 9.3 Queuing Diagram for Scheduling

Priorities



41

Algoritmit

- First-Come-First-Served FCFS
- Round Robin RR
- Virtual Round Robin VRR
- Shortest Process Next SPN
- Shortest Remaining Time SRT
- Highest Response Ratio Next HRRN
- Multilevel Feedback feedback

- Fair Share Scheduling FSS

42

RT scheduling

- Timely execution the most important issue
  - No fairness, no minimising the response time
- Os can only guess the actual execution time
  - Programmer gives some estimates, period gives hints
- Quite often: just schedule rt-tasks fast
  - High priority, predetermined schedule plan, others can suffer
- Two main alternative for dynamic scheduling
  - Earliest Deadline First (EDF)
  - Rate Monotonic Scheduling (RM)

43

Schedulability test

Tbl 10.4 [Stal05]

- Obviously for any schedule, the schedulable utilisation:

$$U_i = C_1/T_1 + C_2/T_2 + .. + C_n/T_n <= 1$$

- Sufficient condition for schedulability using RM scheduling

$$C_1/T_1 + C_2/T_2 + .. + C_n/T_n <= n(2^{1/n} - 1)$$

- N is number of tasks, the upper limit for the value is

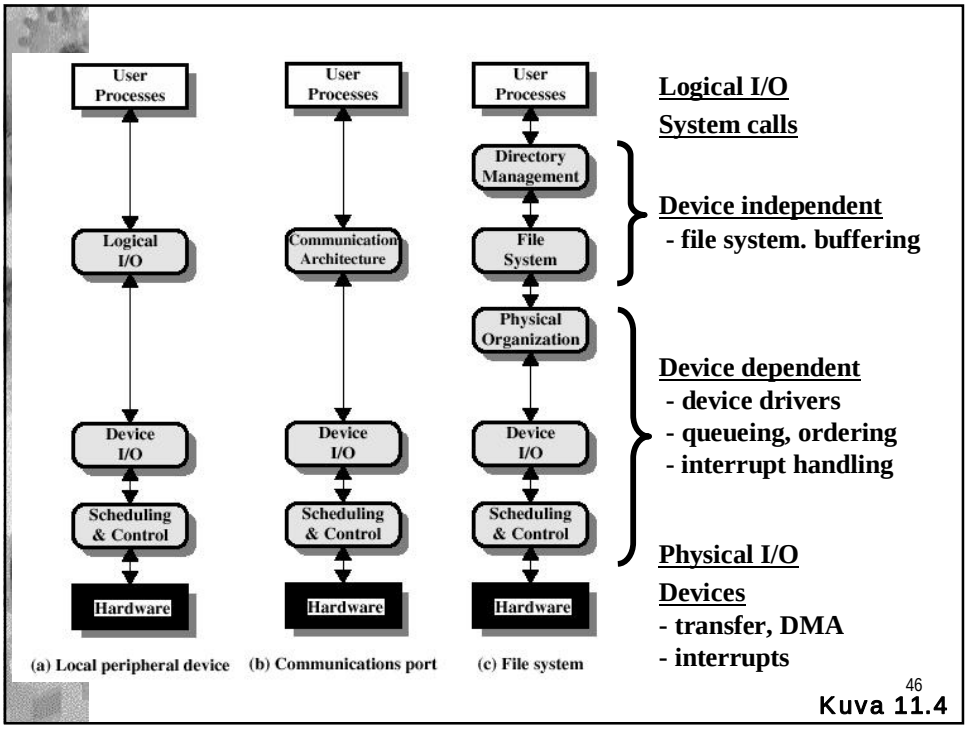
$$\ln 2 \sim 0.693 \rightarrow \text{for any } i \; U_i < 0.693$$

- For EDF the sufficient and effective condition is

$$U_i = C_1/T_1 + C_2/T_2 + .. + C_n/T_n <= 1$$

I/O

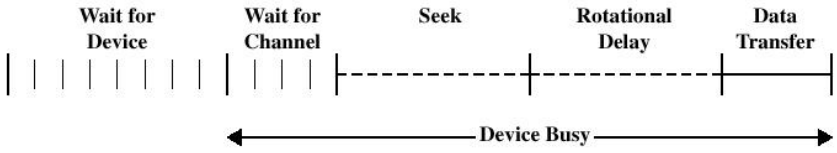
45



46

Kuva 11.4

Disk access delays & scheduling



(Fig 11.6 [Stal05])

- Random? FIFO? PRI? LIFO?
  - Not good, no usage of current position of the arm
- Considers the position
  - SSTF
  - SCAN
  - C-SCAN
  - N-step-SCAN ja FSCAN

47

Levyn vuorotusalgoritmeja

Table 11.2 Comparison of Disk Scheduling Algorithms

(a) FIFO (starting at track 100)		(b) SSTF (starting at track 100)		(c) SCAN (starting at track 100, in the direction of increasing track number)		(d) C-SCAN (starting at track 100, in the direction of increasing track number)	
Next track accessed	**Number of tracks traversed**	**Next track accessed**	**Number of tracks traversed**	**Next track accessed**	**Number of tracks traversed**	**Next track accessed**	**Number of tracks traversed**
55	45	90	10	150	50	150	50
58	3	58	32	160	10	160	10
39	19	55	3	184	24	184	24
18	21	39	16	90	94	18	166
90	72	38	1	58	32	38	20
160	70	18	20	55	3	39	1
150	10	150	132	39	16	55	16
38	112	160	10	38	1	58	3
184	146	184	24	18	20	90	32
Average seek length	55.3	**Average seek length**	27.5	**Average seek length**	27.8	**Average seek length**	35.8

48

RAID - Redundant Array of Independent Disks

- RAID 0 (no redundancy, no replication)
- RAID 1 (mirror, replicated disk)
- RAID 2 (Hamming)
- RAID 3 (parity bit)
- RAID 4 (parity block)
- RAID 5 (distributed parity block)
- RAID 6 (2 distributed parity blocks)

49

block buffers

Ch 11.7 [Stal 05]

50

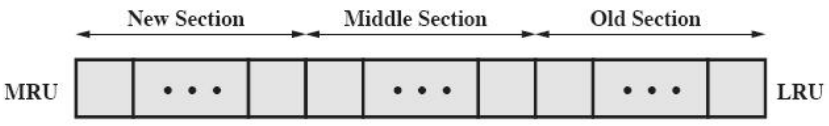
block buffers, disk buffers

- Buffer area in main memory to temporarily store disk blocks
  - Principle of locality
  - Prefetch / delayed write
- Replacement algorithms
  - LRU: Least Recently Used
  - LFU: Least Frequently Used
  - Most Recently Used – MRU FIFO (Frequency Based Replacement)

51

Most Recently Used – MRU Three Sections

- Most Recently Used – MRU Three Sections
  - parannus: jaa jono kolmeen osaan
    - poistot aina viimeisestä osasta
    - etuosasta pudonneelle jää aikaa vanheta
  - tulos: parempi algoritmi kuin LRU tai LFU



(b) Use of three sections

Fig 11.9 (b) [Stal 05]

52

File system

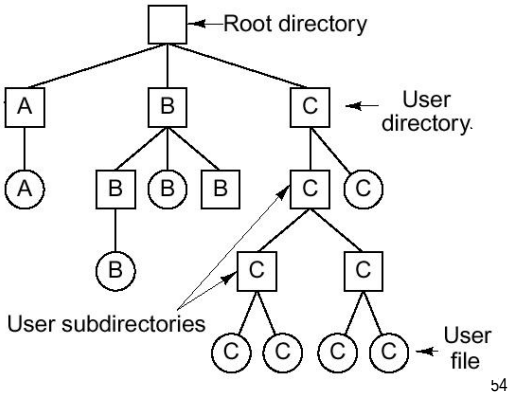
53

Directory

= File, that contains information about other files

- Only OS is allowed directly to access these files
  - All changes through system calls only

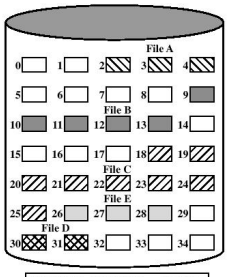
- Root directory, home directories
- Processes can create subdirectories
- Fixed location for root directory on disk



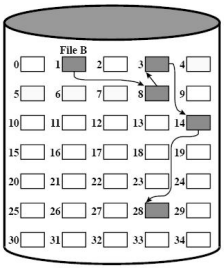
54

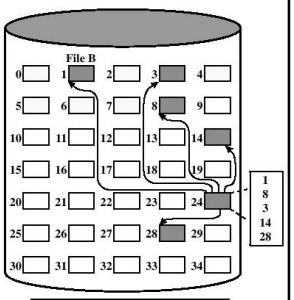
Allocation: Three alternatives

- Contiguous block
- Chained
- Indexed



File Name	Start Block	Length
File A	2	3
File B	9	5
File C	18	8
File D	30	2
File E	26	3

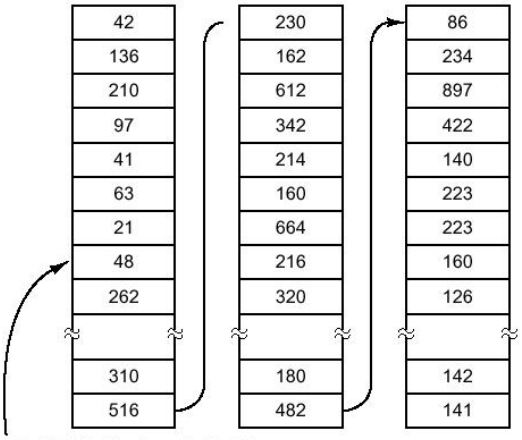




File Name	Index Block
File B	24

55

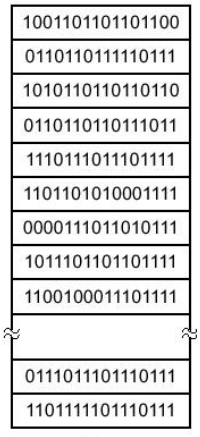
Free space Tan01 6-21



A 1-KB disk block can hold 256 32-bit disk block numbers

(a)

Block list

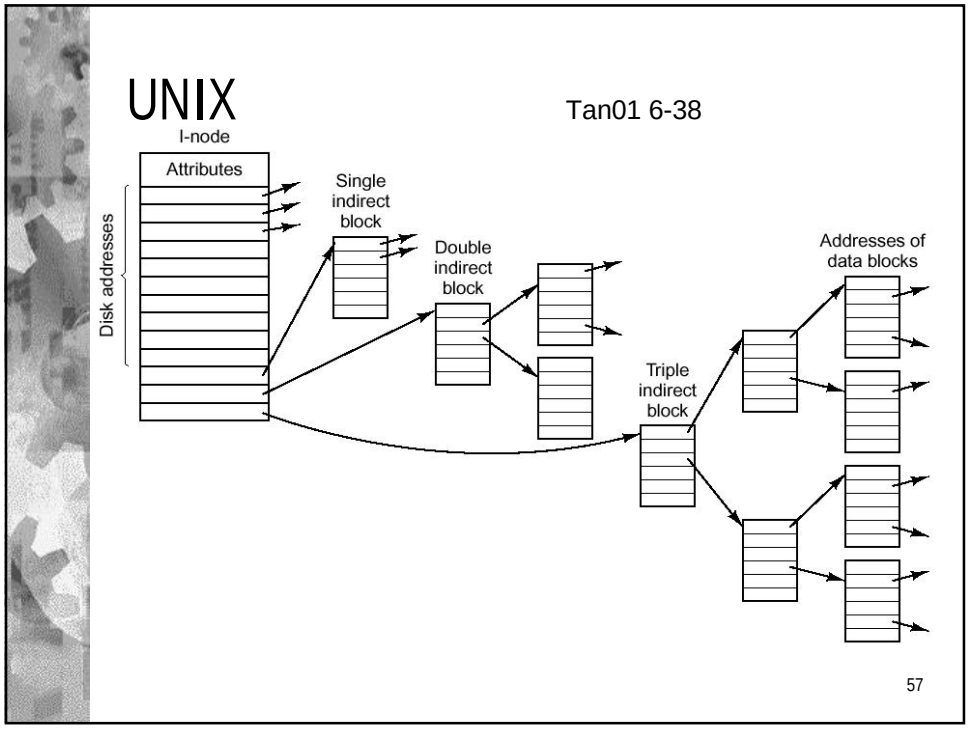


A bitmap

(b)

Bit map

56



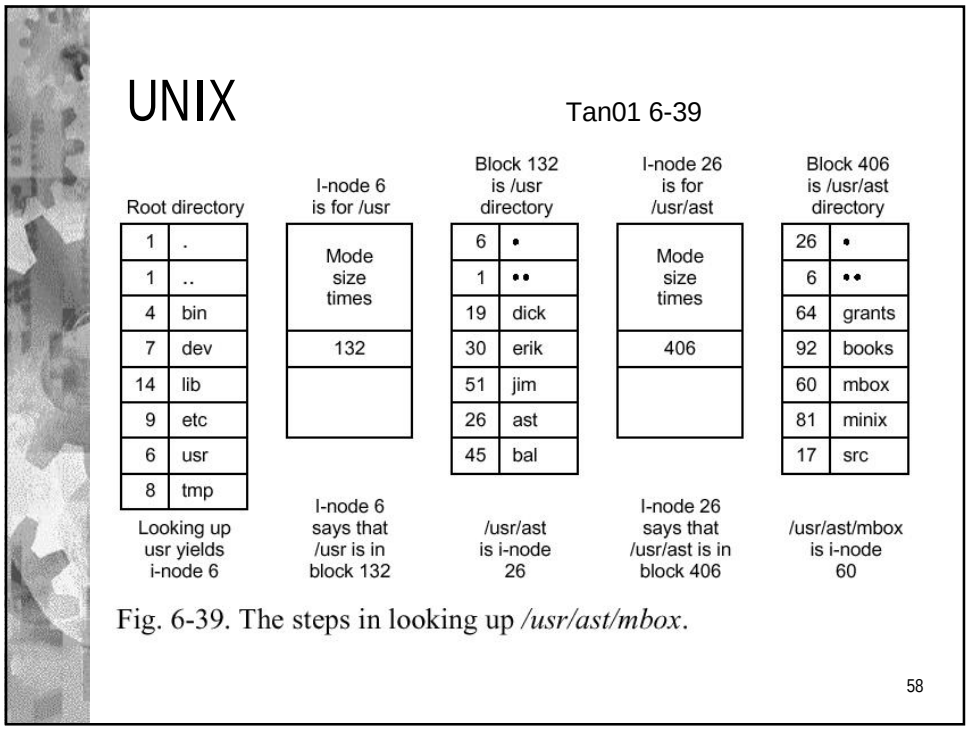


Fig. 6-39. The steps in looking up */usr/ast/mbox*.

LINUX
file system : see slides of week 8

59

Shared file

- Hard link
  - Direct link from several directories
  - Multiple owners, all with same rights
    - removed from one directory ⟶ available still from other locations
- Soft link (symbolic link)
  - New file, which has file type: symbolic link
  - Content of the new file: path to the actual file (string that contains directories and file name)
  - Original file exists only in one directory (-> just one owner)
    - owner removes ⟶ the others cannot access, the link is invalid

60

Tiedostojen yhteiskäyttö

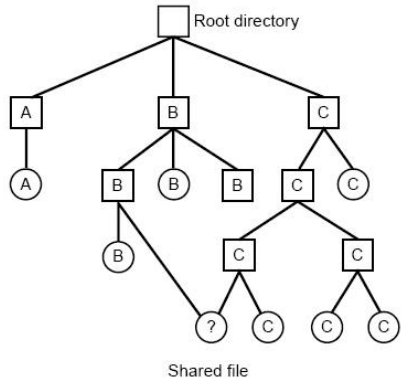


Fig. 6-18. File system containing a shared file.

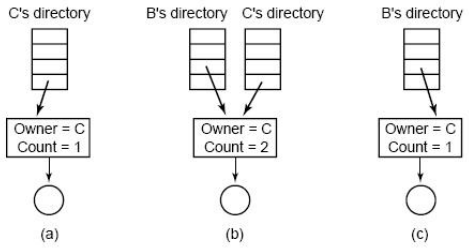


Fig. 6-19. (a) Situation prior to linking. (b) After the link is created. (c) After the original owner removes the file.

[Tane 01]

61

LINUX Virtual File System

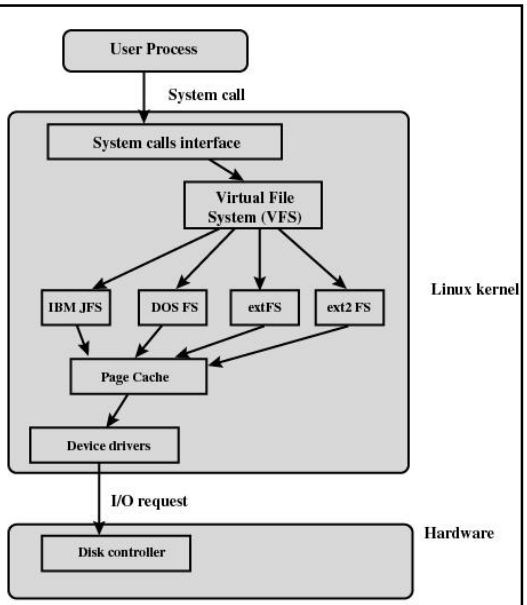


Figure 12.15 Linux Virtual File System Context

Linux file systems

- ext2fs (second extended file system)
- /proc
- ext3fs

63

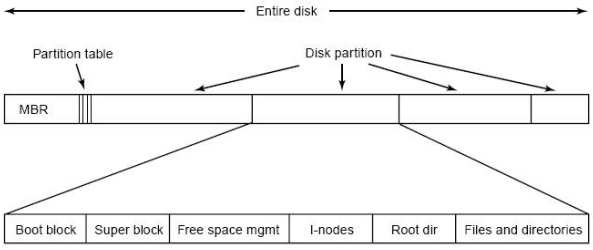


Fig. 6-11. A possible file system layout.

[Tane01]

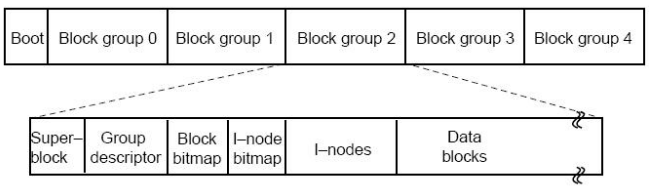


Fig. 10-35. Layout of the Linux Ext2 file system.

64

Linux ext2fs

- block groups
  - Each group is one continuous area of the disk
  - i-nodes and datablock of one group cose to each other
  - Shorter track distances within one file
- All block sizes are identical(1 KB)
- All i-node sizes are 128B (trad. UNIX 64B)

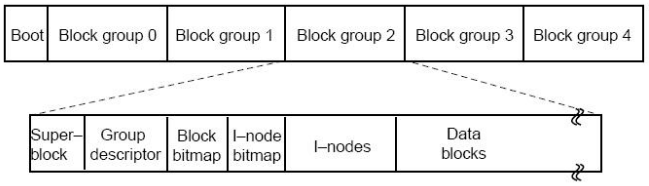


Fig. 10-35 [Tane 01]

Fig. 10-35. Layout of the Linux Ext2 file system.

65

ext2fs superlohko (superblock)

- 1 lohko
- Kuvaa koko ext2fs-partition rakenteen
- Kopio jokaisen lohkoryhmän alussa
  - luotettavuus, virheestä toipuminen

	0	1	2	3	4	5	6	7
0	Number of i-nodes				Number of blocks			
8	Number of reserved blocks				Number of free blocks			
16	Number of free i-nodes				First data block			
24	Block size				Fragment size			
32	Blocks per group				Fragments per group			
40	i-nodes per group				Time of mounting			
48	Time of last write				Status		Max. mnt cnt	
56	Ext2signat.		Status		Error behav.		Pad word	
64	Time of last test				Max test interval			
72	Operating system				File system revision			
80	RESUID		RESGID		Pad word			
blocksize	Pad words							

ext2fs i-node

	0	1	2	3	4	5	6	7
0	Mode		Uid		File size			
8	Access time				Time of creation			
16	Time of modification				Time of deletion			
24	Gid		Link counter		No. of blocks			
32	File attributes				Reserved (OS-dependent)			
40	12 direct blocks							
88	One-stage indirect block				Two-stage indirect block			
96	Three-stage indirect block				File version			
104	File ACL				Directory ACL			
112	Fragment address				Reserved (OS-dependent)			
120	Reserved (OS-dependent)							

Access Control List

67

Journaling (logging) file system

- Kirjataan kaikki muutokset (journalointi?)
- Pitää tiedostojärjestelmän eheänä
- Kirjataan
  - Vain metatieto muutoksista – journal
  - Sekä metatieto että itse data – loki
- Tarve:
  - Tiedostojärjestelmän tarkistus (check) kestää liian kauan, jos epänormaali 'kaatuminen'
  - Valtaosa levyoperaatioista on kirjoituksia, lukuoperaatiot tehdään puskureista
  - Useimmat kirjoitukset pieniä päivityksiä
    - levyn hakuvarsi liikkuu paljon, vähän dataa siirtyy

68

Perusidea

- Ongelma tavallisen tiedostojärjestelmän uuden tiedoston X luomisessa:
  - kirjoita hakemiston i-node, hakemisto, tiedoston i-node ja lopulta tiedosto
  - virta poikki (tms vika) kesken kaiken? Oooops.
- Ratkaisu: tapahtumaloki, joka takaa tiedostojärjestelmän eheyden - vrt tietokantojen loki
- Esim: Microsoft NTFS, Red Hat Linux ext3fs

69

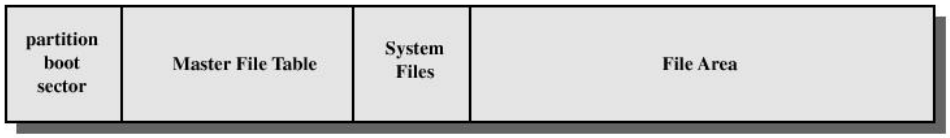
NTFS: Piirteitä

- Kaatumisista ja levyvirheistä toipuminen
  - LFS lokitiedoston avulla
- Käyttöoikeudet
  - pääsylistat (security descriptor)
- Sallii suuret levyt ja tiedostot
  - FAT32'ssa vain 2^{32} lohkoa, suuri allokointitaulu
- Tiedosto-oliot ovat *(arvo, attribuutti)* -pareja
- Mahdollisuus indeksointiin tiedoston käsittelyn nopeuttamiseksi
- Lohko, cluster
  - yksi tai useampi peräkkäinen sektori (esim. 512 B - 4 KB)
    - 32 GB levyllä 128 sektoria/lohko (⟶ lohko 64-512 KB)
  - varauksen ja kirjanpidon perusyksikkö
- Partitio, volume
  - fyysinen levyn looginen osa, jolla oma tiedostojärjestelmä

70

NTFS-partitio

(Fig. 12.17 [Stal 05])



- Boottilohko
  - partition ja tiedostojärj. rakenne, boottitietue ja -koodi
  - MFT:n sijainti
- MFT
  - tietoa tiedostoista, hakemistoista (folders) ja vapaasta tilasta
- System Files (~ 1MB)
  - kopio MFT:n alkuosasta
  - virheistätoipumisloki, bittikartta vapaat/varatut lohkot, attribuuttien kuvaustaulu
- File Area - tiedostojen lohkoille

71

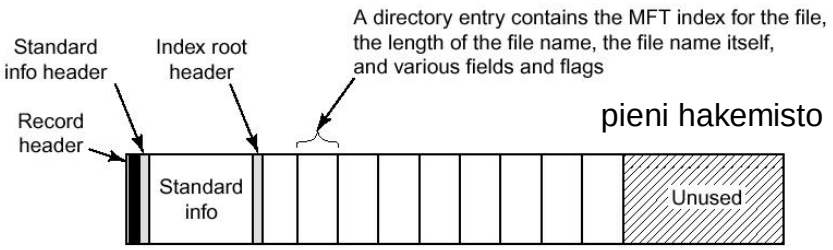
NTFS – MFT (Master File Table)

- 1 KB:n kokoisia MFT-tietueita
  - jokainen kuvaa yhden taltiolla olevan tiedoston
    - myös hakemisto on tiedosto
  - vaihtelevanmittainen osa käytössä
    - (attribuutti, arvo) pareja (ei paikkasidonnainen!)
    - data attribuutti, 'arvo' = lohkojen sijainti
- 16 ensimmäistä tietuetta varattu ns. metadatalle
  - 16 $-alkuista tiedostoa
- Jos pieni tiedosto, tietue sisältää myös datan
- Jos iso tiedosto, data erillisellä tallealueella
  - MFT-tietuessa lohkonumeroita
  - kuvaus voi jatkua useampaan MFT-tietueseen

72

Hakemiston MFT-tietue

(Fig. 11-38 [Tane01])



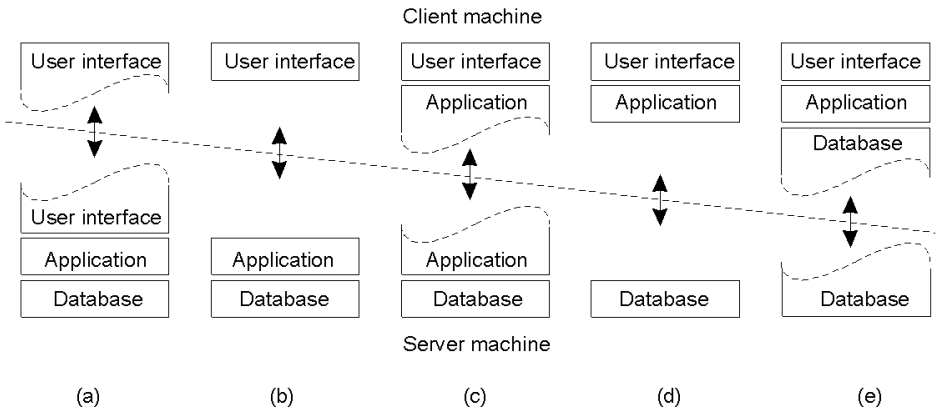
- Pienissä hakemistoissa MFT-tietueet peräkkäisjärjestyksessä
- Isoissa hakemistoissa MFT-tietueessa B-puun (B-tree) indeksirakenne
  - nimen etsintä ei ole peräkkäishakua

73

Hajautettu prosessi

74

Asiakas-palvelija luokitteluja

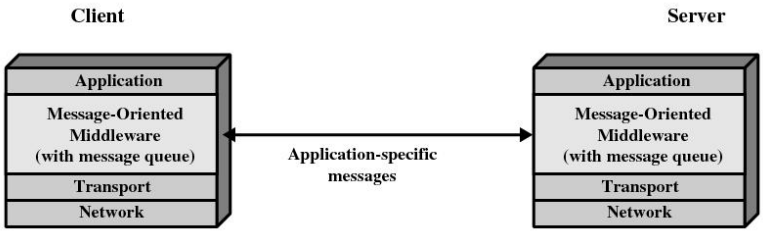


1-29 Alternative client-server organizations.

75

Process communication : Ch 14 message passing

- Reliable / unreliable
- Synchronous / asynchronous
- Persistent / transient
- Binding
  - Based on network topology or message content
  - Static / Dynamic

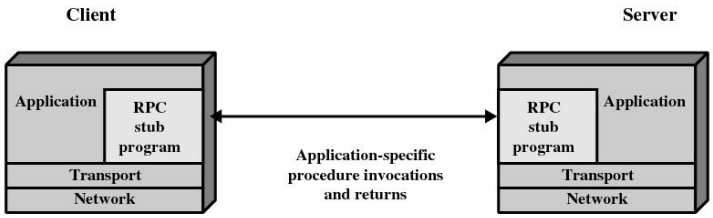


76

(a) Message-Oriented Middleware

Process communication: RPC

- Remote procedure call looks to the calling client as any other local procedure (or function or method) call.



(b) Remote Procedure Calls

77

RPC

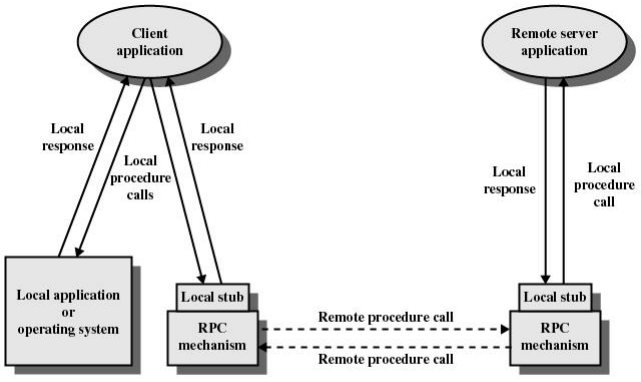
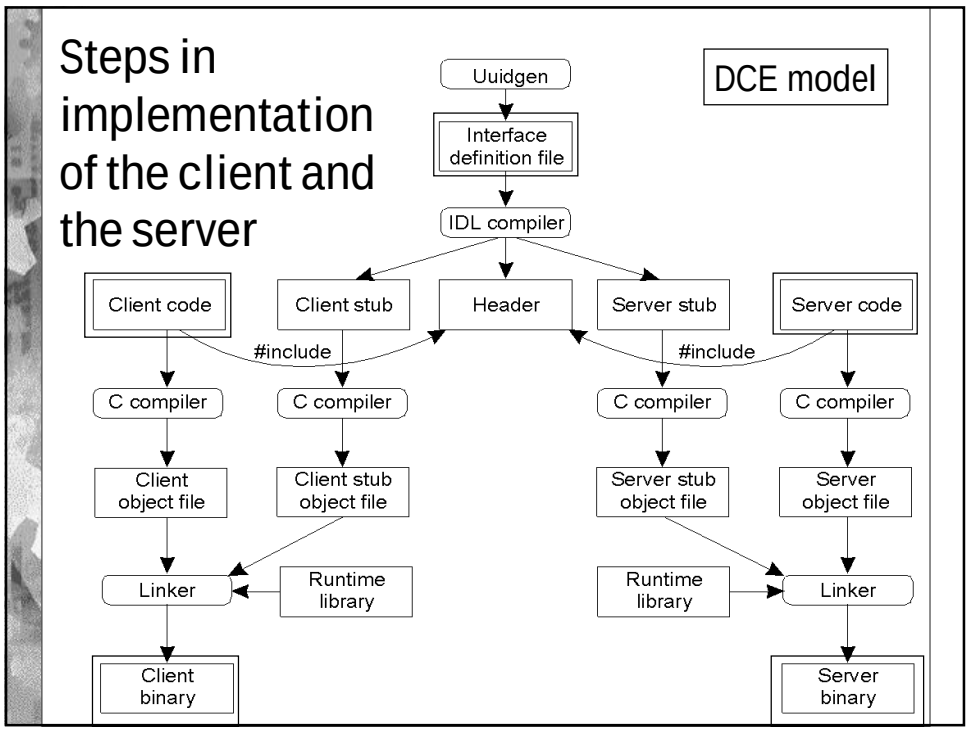
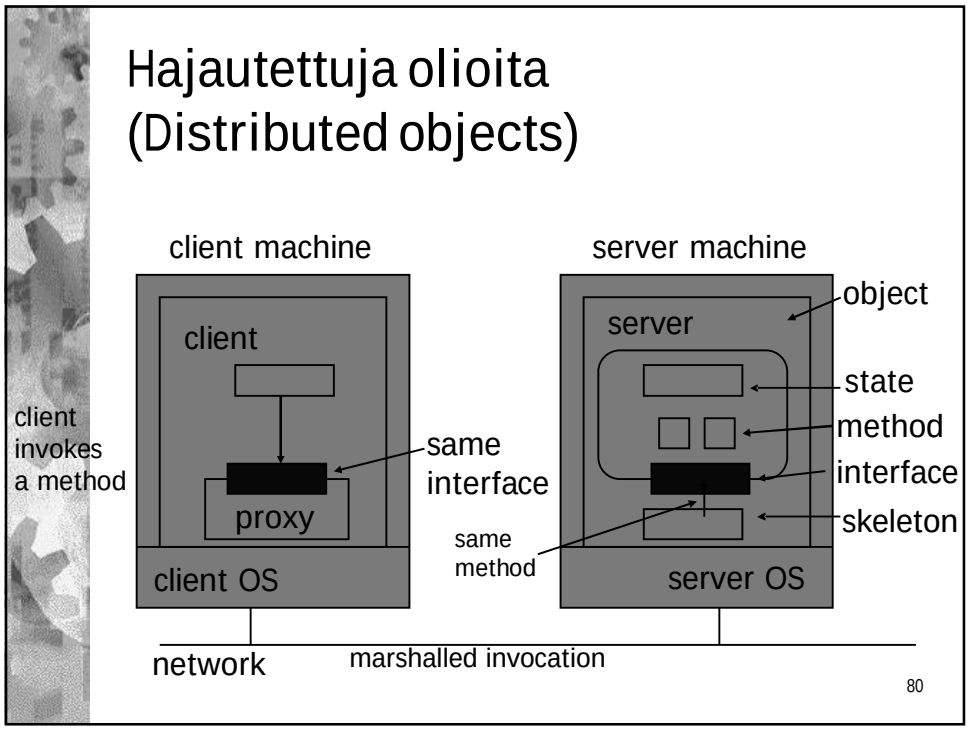
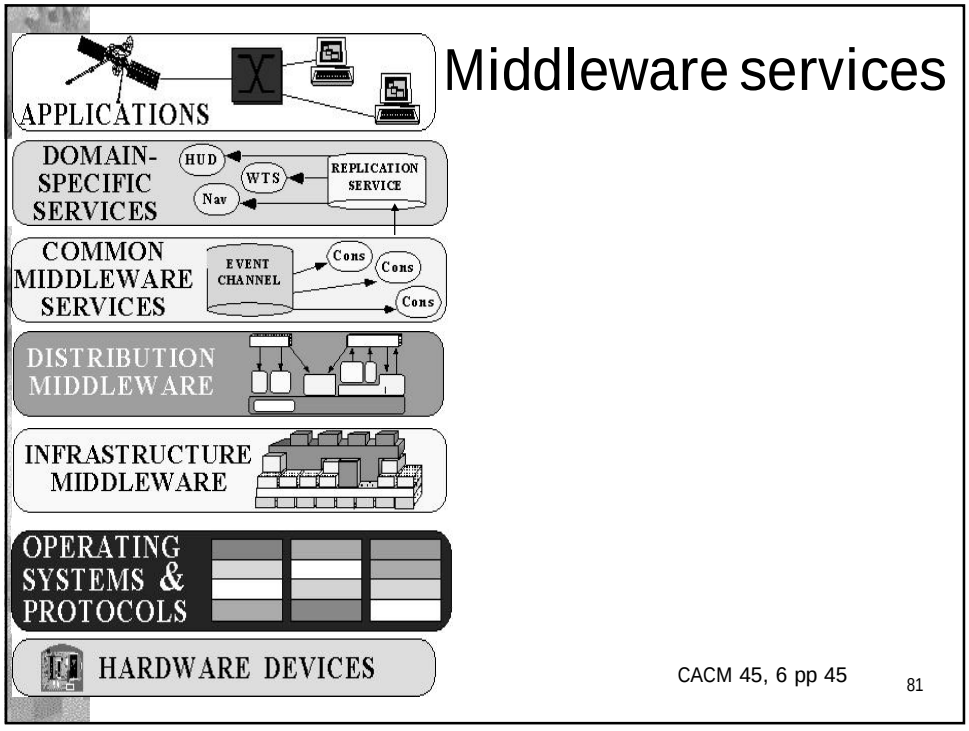


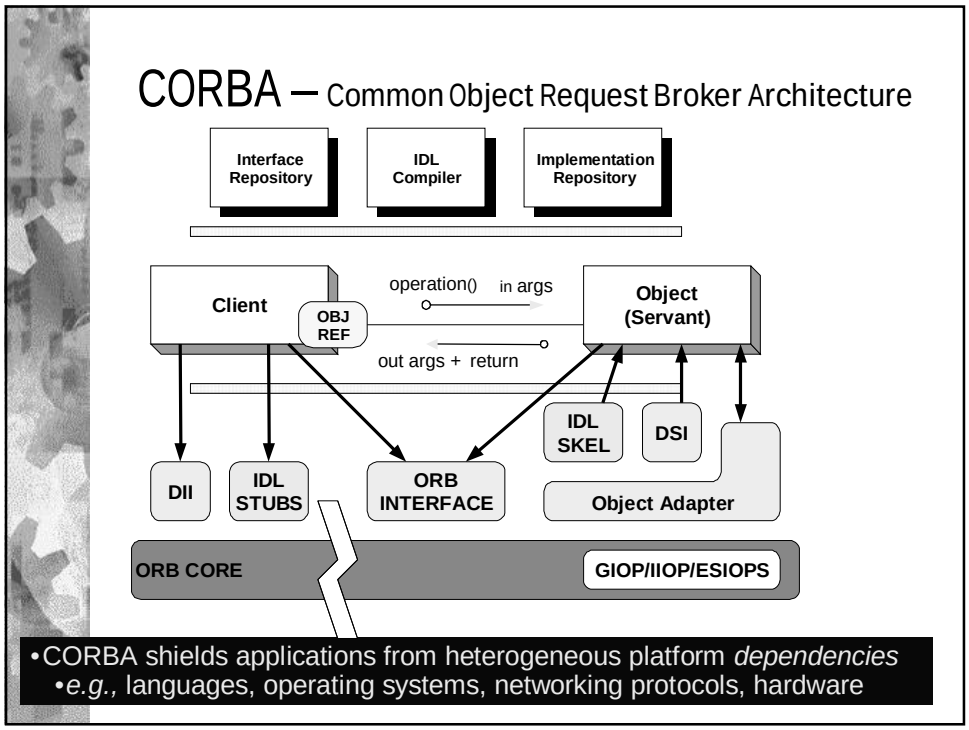
Figure 14.12 Remote Procedure Call Mechanism

78


Steps in implementation of the client and the server
DCE model
Uuidgen
Interface definition file
IDL compiler
Client code
Client stub
Header
Server stub
Server code
#include
#include
C compiler
C compiler
C compiler
C compiler
Client object file
Client stub object file
Server stub object file
Server object file
Linker
Runtime library
Runtime library
Linker
Client binary
Server binary


Hajautettuja olioita (Distributed objects)
client machine
server machine
client
server
object
state
method
interface
skeleton
client invokes a method
same interface
proxy
same method
client OS
server OS
network
marshalled invocation
80


Middleware services
APPLICATIONS
DOMAIN-SPECIFIC SERVICES
HUD
WTS
Nav
REPLICATION SERVICE
COMMON MIDDLEWARE SERVICES
EVENT CHANNEL
Cons
Cons
Cons
DISTRIBUTION MIDDLEWARE
INFRASTRUCTURE MIDDLEWARE
OPERATING SYSTEMS & PROTOCOLS
HARDWARE DEVICES
CACM 45, 6 pp 45
81


CORBA – Common Object Request Broker Architecture
Interface Repository
IDL Compiler
Implementation Repository
Client
OBJ REF
operation()
in args
out args + return
Object (Servant)
IDL SKEL
DSI
DII
IDL STUBS
ORB INTERFACE
Object Adapter
ORB CORE
GIOP/IIOP/ESIOPS
•CORBA shields applications from heterogeneous platform *dependencies*
•*e.g.,* languages, operating systems, networking protocols, hardware

Cluster: classification

Clustering Method	Description	Benefits	Limitations
Passive Standby	A secondary server takes over in case of primary server failure.	Easy to implement.	High cost because the secondary server is unavailable for other processing tasks.
Active Secondary:	The secondary server is also used for processing tasks.	Reduced cost because secondary servers can be used for processing.	Increased complexity.
- Separate Servers	Separate servers have their own disks. Data is continuously copied from primary to secondary server.	High availability.	High network and server overhead due to copying operations.
- Servers Connected to Disks	Servers are cabled to the same disks, but each server owns its disks. If one server fails, its disks are taken over by the other server.	Reduced network and server overhead due to elimination of copying operations.	Usually requires disk mirroring or RAID technology to compensate for risk of disk failure.
- Servers Share Disks	Multiple servers simultaneously share access to disks.	Low network and server overhead. Reduced risk of downtime caused by disk failure.	Requires lock manager software. Usually used with disk mirroring or RAID technology.

Cluster Computer Architecture

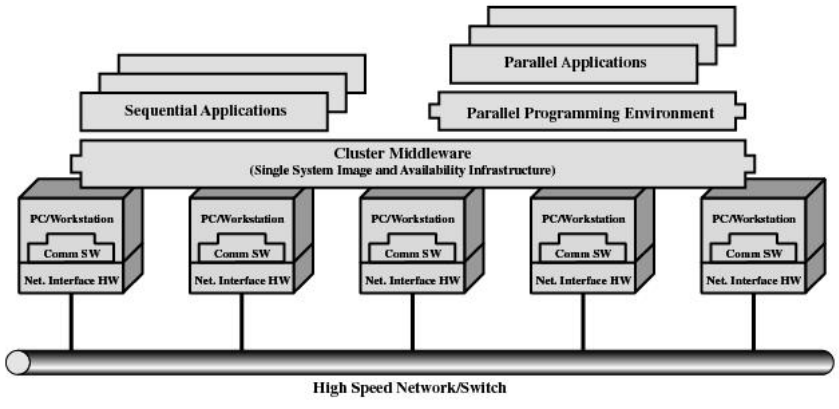
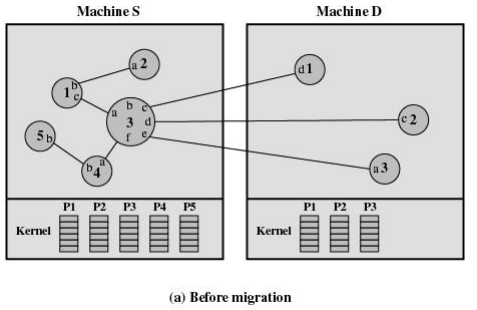


Figure 14.14 Cluster Computer Architecture [BUYY99a]

84

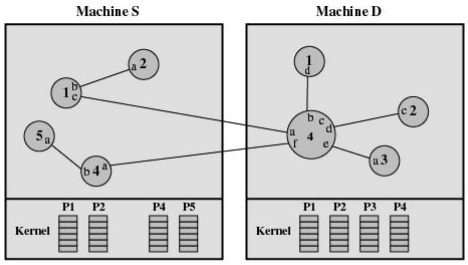
Process Migration



(a) Before migration

Prosessi 3 (solmu S)
Prosessiksi 4 (solmu D)

- Load balancing, availability, move closer to a resource



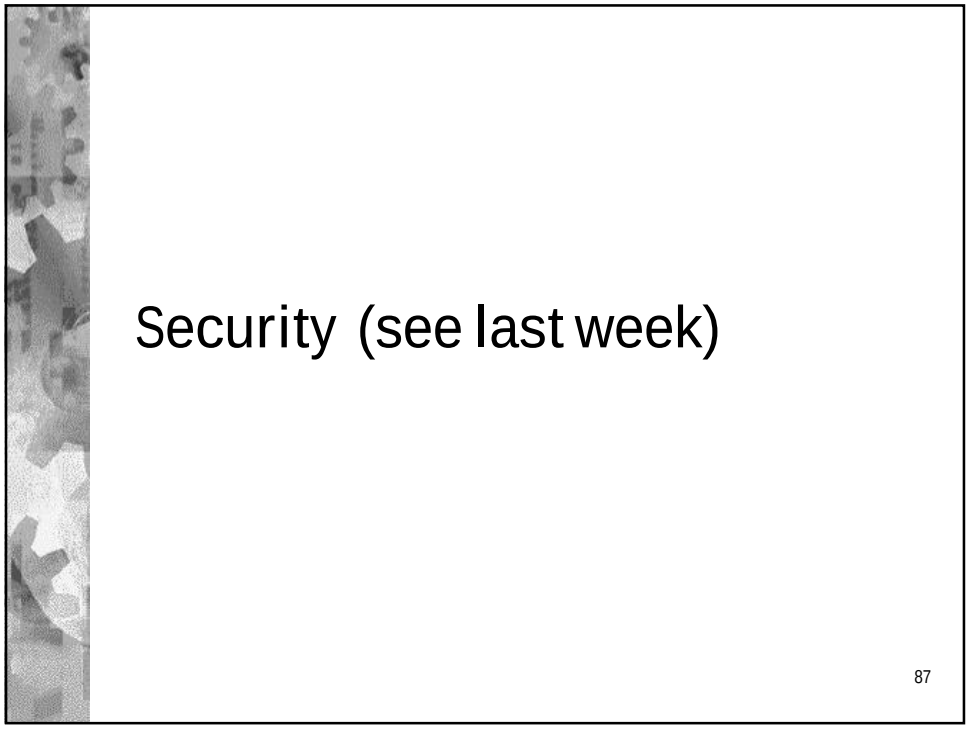
(b) After migration

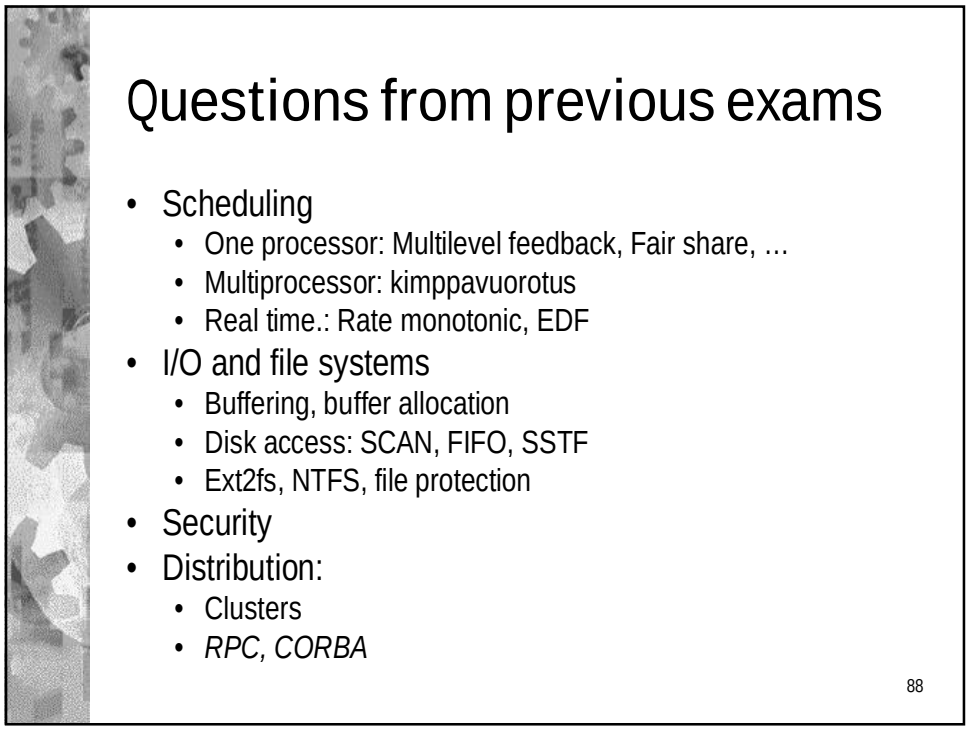
85

Siirtopolitiikkoja

- Eager (all):
  - Siirrä kaikki (muistialueet+muut)
- Precopy:
  - Prosessia vielä suoritetaan, kun muistialueita jo kopioidaan
- Eager (dirty):
  - Siirrä vain se osa muistiavaruutta, joka on keskusmuistissa ja jota on muutettu
- Copy-on-reference:
  - Siirrä sivua vain viittattaessa
- Flushing:
  - Kopioi prosessin muuttuneet sivut levylle

86


Security (see last week)
87


Questions from previous exams
• Scheduling
• One processor: Multilevel feedback, Fair share, …
• Multiprocessor: kimppavuorotus
• Real time.: Rate monotonic, EDF
• I/O and file systems
• Buffering, buffer allocation
• Disk access: SCAN, FIFO, SSTF
• Ext2fs, NTFS, file protection
• Security
• Distribution:
• Clusters
• RPC, CORBA
88

-- END --

Käyttöjärjestelmät

89