# Practicum opgaven Week 5 - Yuri Hoogeweg

## What are the advantages of the variation of linked allocation that uses a FAT to chain together the blocks of a file?

The advantage is that the lookup to find a disk block is a lot more efficient using a FAT. The location of a block can be found by using the pointers in the FAT instead of checking each block in a file. Especially if FAT is cached this is a lot faster because it means that most of the time, only memory accesses are necessary to determine the pointer.

## Could a RAID Level 1 organization achieve better performance for read requests than a RAID Level 0 organization (with nonredundant striping of data)? If so, how?

Yes. A Raid Level 1 organization **could** achieve better performance. If the type of disks used still use disk heads. A raid level 1 organization offers two copies of the same data, so if one of the two disks already has the disk head in the correct or very close position, this would reduce the time needed for a disk head to 'travel' to the appropriate location. Contrary to Raid level 0 organizations which would need to move every disk head to find the correct bit on every disk.

## What are the advantages and disadvantages of supporting memory-mapped I/O to device-control registers?

The advantages are that if the I/O instructions are mapped to memory, there is no need to have special I/O instructions in the instruction set, because the CPU can just use read/write instructions to the device-control registers this way.

One of the disadvantages is that making the device-control registers accessible through memory is dangerous. Because if this memory space is not properly protected, any user program *could* gain access to the device-control registers.

## Describe three circumstances under which blocking I/O should be used. Describe three circumstances under which nonblocking I/O should be used. Why not just implement nonblocking I/O and have processes busy-wait until their device is ready?

1. I/O-blocking when a process is waiting for a specific event. For example a disk, keyboard or memory read.
2. Non-blocking I/O when a process is not waiting for a specific event but just 'any' event from multiple possible sources. For example (web) servers (don't know what request is going to come from where), programs accepting multiple inputs simultaneously, and any other process that doesn't know what kind of request is going to come from where. (Copy-operation?)

It is not efficient to just implement nonblocking I/O everywhere because it is less efficient than blocking I/O. Busy waiting requires a lot of CPU cycles so the overall performance of applications would decrease drastically.

## Explain why doubling the speed of the systems on an Ethernet segment may result in decreased network performance. What changes could help solve this problem

When all systems on a network double in speed, they will be able to send more network traffic (packets) more rapidly than before. This means that the infrastructure might not be able to handle this amount of traffic, resulting in decreased network performance or even packet loss. Changes that could help solve these problems are increasing the performance of systems in the rest of the network infrastructure (routers, cabling etc) or splitting up the network in smaller sub-networks so that there's only a small amount of high-performance systems in each network.

## The original HTTP protocol used TCP/IP as the underlying network protocol. For each page, graphic, or applet, a separate TCP session was constructed, used, and torn down. Because of the overhead of building and destroying TCP/IP connections, performance problems resulted from this implementation method. Would using UDP rather than TCP be a good alternative? What other changes could you make to improve HTTP performance?

I don't think UDP is a good alternative to TCP. It may have less overhead but UDP is a lot less reliable. UDP offers no guarantee whatsoever that the packages sent will 1. All arrive 2. Arrive without being corrupt.

A better solution would be to allow TCP connections to 'stay opened' and use one connection to receive all the data through instead of separate connections.