**Computernetwerken**

Hoofdstuk 1

* 1. Protocollen

Wanneer computers met elkaar willen communiceren moeten er vaste afspraken worden gelegd.

= **PROTOCOL**:

* Welke berichten er kunnen verstuurd worden
* Wat elk bericht betekent
* Wat de onderlinge verhoudingen zijn tussen de berichten, o.a. hoe op elk bericht kan gereageerd worden

Per aspect wordt een protocol opgesteld.

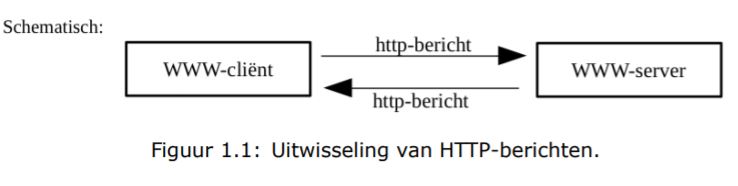
Per aspect zal er een set van afspraken zijn: een protocol. Verschillende aspecten worden vastgelegd is verschillende protocollen die samen een protocolstack (= hoop protocollen die op elkaar voortbouwen) vormen.

Eén computer neemt het initiatief om contact op te nemen met de andere. De andere luistert.

Dit heet het serverprogramma.

Het programma dat het initiatief neemt, richt meestal een verzoek aan het serverprogramma. Het initiatie nemend programma heet clientprogramma.

World Wide Web – protocol

HTTP = HyperText Transfer Protocol

Bij verschillende webbrowser mogen de interface, programmeertaal, enz. allemaal verschillen maar de bericht die naar een serverprogramma gestuurd worden MOETEN hetzelfde zijn.

* 1. Protocollagen

Het verschil tussen een protocol en een programma is dat een protocol een set is van afspraken en een programma een één van de mogelijke implementaties is van deze afspraken.

Een computernetwerk kan worden gezien aks een infrastructuur die **gedistribueerde applicaties** voorziet van diensten. Als een programma (bv. WWW-client) een bericht stuurt naar ene programma op ene andere machine (bv. WWW-server) dan wordt er hierbij een netwerk gebruikt.

Een netwerk heeft verschillende mogelijke technologieën:

* ethernet,
* token ring,
* ATM

Zelfde browsers kunnen het zelfde netwerk gebruiken.  
Dit komt door:

* de browser moet niet zelf het HTTP-bericht omvormen tot een bitrij en moet zelf de bitrij niet op het netwerk zetten
* de browser zal geen bericht afleveren aan programmatuur op een lager niveau die ervoor zorgt dat het HTTP-bericht omgevormd wordt in een bitrij en op het netwerk geplaatst wordt
* hetzelfde geld voor de berichten die de browser ontvangt

Dus, vooraleer een bericht van een applicatie op het netwerk kan gezet worden, moet de netwerkprogrammatuur bewerkt worden.

De netwerkprogrammatuur heeft verschillende lagen:

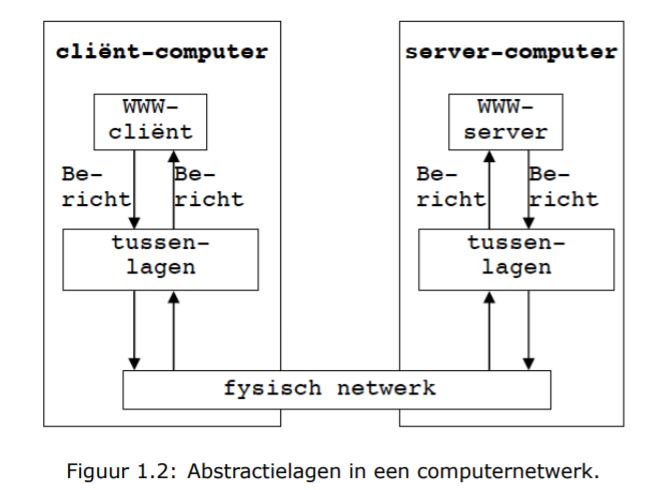
* (hoogste niveau) applicatielaag of toepassingslaag:
  + Op deze laag woorden communicatieboodschappen uitgewisseld tussen toepassingen
* (tussen lagen)
  + Zie verder…
* (laagste niveau) fysieke laag
  + Deze laag verstuurt de bitrij op het netwerk en zet ontvangen signalen v/h fysieke communicatiemedium om in bitrij

De afspraken i.v.m. netwerken zijn ingedeeld in (abstractie)lagen. Een laag op het hoogste niveau hoeft geen kennis te hebben van het type netwerk of type communicatiemedium dat gebruikt wordt om bv. HTTP-berichten te versturen of te ontvangen.

Het HTTP-bericht wordt in onderliggende tussenlagen vertaald in een bitrij.

Indeling in lagen heeft twee belangrijke voordelen:

* De complexiteit kan verdeeld worden d.m.v. abstractie
* Meerdere toepassingen kunnen gebruik maken van dezelfde programmatuur van de onderliggende laag

Door de opdeling in lagen kan men bij het ontwerpen van cliënt- en serverprogramma’s veronderstellen dat deze programma’s berichten met mekaar uitwisselen. Zonder dat er rekening moet worden gehouden met het fysieke netwerk (figuur 1.2)

De communicatie in figuur 1.1 in is bijgevolg **abstract** of **virtueel.**

In werkelijkheid worden de berichten afgeleverd aan ene tussenlaag of ontvangen van een tussenlaag. Elke tussenlaag bewerkt het bericht en levert een bericht  
(of meerdere berichten) af aan de laag eronder. Op de fysieke laag wordt het bericht overgebracht over het communicatiemedium.

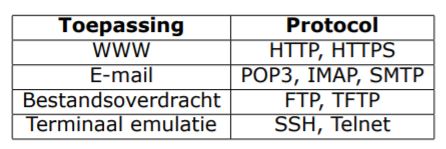
In computernetwerken worden(lange) berichten van de toepassingslaag meestal in kleinere stukken verstuurd (een stuk = een segment). Er is een protocol dat alle afspraken voor opdeling en wedersamenstelling van deze segmenten omvat. Zoals:

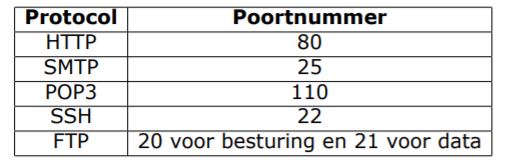
* Hoe groot kan een segment zijn
* Wat als een segment niet correct is
* …

Voordelen om grote berichten in segmenten om te delen:

* Kans op een fout in een segment is kleiner, als het allemaal in één keer doorgestuurd zou worden en er een fout is. Dan zou het HELE bestand opnieuw verzonden moeten worden
* Het netwerk kan segmenten van verschillende communicaties door mekaar versturen. Doordoor kunnen meerdere applicaties of toepassingen het netwerk op het zelfde moment gebruiken
  1. TCP/IP

Een cliëntprogramma kan niet zomaar om het even welk bericht sturen naar ene serverprogramma. Er moet aan een aantal afspraken worden voldaan. Voor elke toepassing is er een apart protocol.



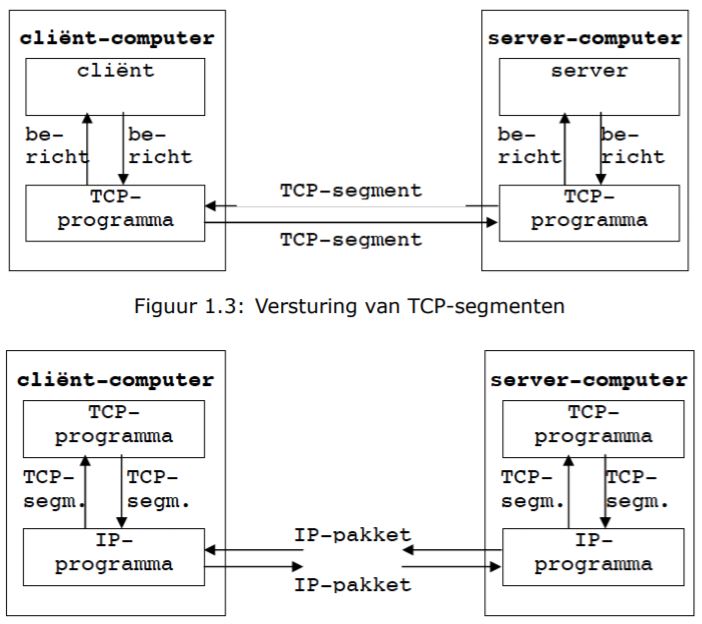
Een cliënt die een dienst wil, moet het IP-adres en het gepaste poortnummer v/h serverprogramma opgeven.

Wanneer ene serverbericht wordt terug gestuurd. Dan moet de server het IP-adres en poortnummer v/h cliëntproces kennen. Als een cliënt een verbinding start met het serverprogramma, dan wordt er een willekeurig nummer (boven 1024) als poortnummer gekozen. Dit server-poortnummer blijft voor diezelfde applicatie hetzelfde.

**Internet RFC’s (= Request for comments)**Cliënt en server voeren protocollen uit die in een computernetwerk het volgende regelen:

* Boodschappen en de volgorde ervan die worden verstuurd
* Boodschappen en de volgorde ervan
* De structuur van de boodschappen

Om iedereen de mogelijkheid te gen om cliënts en/of servers te ontwikkelen zijn er publieke en gestandaardiseerde protocollen nodig. Internetstandaarden worden ontwikkeld door de IETF.

Protocollen zoals HTPP, SSH, FTP, SMTP, … bevatten afspraken over de berichten die de toepassingsprogramma’s (bv. Een browser, programma voor terminaal-emulatie, … of verschillende serverprogramma’s) kunnen uitwisselen.

* 1. TCP/IP-model van de tussenlagen

Toepassingsprogramma’s communiceren niet rechtstreek met mekaar. Ze maken gebruik van programmatuur op een lager niveau. Meestal is dit het TCP-programma. Dit is een implementatie van het TCP ( = Transmission Control Protocol). Het verstuurt/ontvangt berichten die aan het TCP beantwoorden. Deze berichten worden TCP-segmenten genoemd (figuur 1.3).

De TCP-programmatuur wordt gebruikt door de toepassingsprogrammatuur en staat dus op een lager niveau.

Protocollen op:

* Toepassingslaag : HTTPP, FTP, …
* Transportlaag : UDP, TCP

De TCP-programma’s op verschillende machines wisseleen niet rechtstreek maar virtueel TCP- (of UDP-)segmenten uit. Er wordt gebruik gemaakt van programmatuur op een lagere laag: IP-programma.

Dit is een implementatie van het IP (= internet protocol). Berichten die beantwoorden aan het IP worden IP-pakketten genoemd. De laag waarop IP functioneert (figuur 1.4) wordt de netwerklaag genoemd.

IP-programma’s gebruiker een lagere laag: datalinklaag.  
Voorbeeld van programmatuur in deze laag: besturingsprogramma of driver voor de netwerkkaart.  
Berichten die in de datalinklaag uitgewisseld worden, zijn bv. Ethernet-frames.

Frames worden doormiddel van de diensten v/d fysieke laag van de ene naar de andere machine gestuurd.

De fysieke laag bevat afspraken i.v.m. de voorstelling van 0 en 1 (bv. Welke spanningsniveaus).  
De apparatuur (netwerkkaart) genereert spanningen op de kabel en verstuurt aldus bits. De apparatuur in de andere machine herkent deze spanningen en zet deze terug om in bits.

* 1. OSI-referentiemodel van de tussenlagen

P 12