Potchefstroomse Universiteit
vir Christelike Hoër Onderwys

Ontstaan 1869
50 jaar selfstandig 2001

WETENSKAPLIKE BYDRAES
REEKS H: INOUGURELE REDE NR. 172

IT: INSTRUMENT OF ORNAMENT?

Prof DB Jordaan

Inougurele rede gehou op 16 Augustus 2002

Publikasiebeheerkomitee
Potchefstroomse Universiteit vir Christelike Hoër Onderwys
Potchefstroom
2520

Navrae in verband met *Wetenskaplike Bydraes* moet gerig word aan:

Die Publikasiebeheerkomitee
Potchefstroomse Universiteit vir Christelike Hoër Onderwys
2520 POTCHEFSTROOM

ISBN 1-86822-417-1

1 Inleiding

Om die vraag "IT: Instrument of Ornament?" te beantwoord, word daar eerstens, aandag gegee aan inligtingstegnologie (IT) as 'n moontlike instrument en tweedens aan IT as 'n moontlike ornament.

Inligtingstegnologie is 'n baie breë konsep en weens die tydsbeperking, sal slegs die internet, en meer spesifiek die konsep "Transmission Control Protocol/Internet Protocol (TCP/IP)" oorsigtelik bespreek word, aangesien die internet waarskynlik die enkele faset van IT is wat deur die meeste persone hier teenwoordig op een of ander manier gebruik word. In die tweede gedeelte van hierdie bespreking word 'n aantal uitdagings aangeroer.

Weens die aard van die vakgebied, word die eerste gedeelte van hierdie aanbieding in Engels gedoen, aangesien daar baie rekenaarterme gebruik word wat makliker in Engels verstaan word. Die tweede helfte word in Afrikaans aangebied.

2 IT: Instrument

2.1 Understanding TCP/IP

2.1.1 A brief History of TCP/IP

The period of computer history spanning the 1950's and 1960's did not constitute a good time for networking. During this Dark Age of Computerdom, almost all computer systems were "technocentric," operating autonomously—they weren't designed to be connected to other systems. In that politically incorrect period of computer prejudice, hardware, operating systems, file formats, program interfaces, and other components, were all designed to work with only a particular type of computer system, exclusive of all others.

2.1.2 The Interest in Packet-Switched WANs

In the late 1960's, the United States Department of Defence (DOD) became interested in various academic research pieces concerning a Packet-switched Wide-area network, or WAN. The basic idea was to connect multiple, geographically dispersed networks,

and to allow for data, to be sent to the various locations within the WAN, in the form of packets.

The concept of packets can be explained like this: imagine that you have a really long letter to send—so long, it's impossible to fit it into one measly little #10 envelope. You've been given explicit instructions—you must use #10 envelopes. So, you begin to break up the letter into smaller sections, fitting each into an individual envelope. As you address each envelope, you number them sequentially so the recipient at its destination can successfully reassemble your letter. The letter we're talking about, is analogous to data that a user has created within an application and that he wishes to send to another user. The envelopes represent packets. In WAN's, information is transported by electronically inserting it into packets, which are addressed, sequenced, and then sent on their way.

The switched part of a Packet-switched Network refers to the routing of the packets towards a specific destination. As packets are addressed individually, they can be transmitted along different physical routes to their ultimate destination. This flexible transmission method is referred to as "packet-switching". The original reason the DOD was interested in this research, was because they wanted to create a fault-tolerant WAN that could carry, command, and control information in the event of a nuclear war. Because a network of this type would have multiple, geographically dispersed sites, and data would be sent in a packet-switched manner, there would be no single point of failure in the system.

2.1.3 The initial Research Issues behind the Internet

The research arm of the DOD was an agency called the Advanced Research Projects Agency (ARPA), now called the Defence Advanced Research Projects Agency (DARPA). The mission of this group was to fund basic research that could possibly contribute to the defence effort. It was this agency that funded and managed the project to create a packet-switched WAN. The scientists and engineers that were recruited for this project, came from major universities and the private firm of Bolt, Beranek, and Newman (BBN) in Cambridge, Massachusetts. The challenge they faced, related to two main areas: interconnectivity and interoperability.

Interconnectivity deals with transporting information. A software protocol was needed that could package and route information between multiple sites. Out of the concept of the packet-switched WAN evolved the protocol that eventually rose to meet this need: the Internet Protocol (IP).

With the problem of transmission resolved, the team moved on to tackle the next issue—communication. What good was transporting information from an application on a computer here, if the system's applications on the receiving end there couldn't understand it? Interoperability relates to application-to-application communication. Achieving interoperability was a real challenge.

Applications would be running on vastly disparate hardware platforms, with different operating systems, file formats, terminal types, and so on. For interoperability to be a reality, a way to bridge all these differences, was required.

The solution was to develop a series of standard application protocols that would enable application-to-application communication, independent of the extensive array of computer platforms. For instance, if a mainframe-based email program and a PC-based e-mail program were both using the same standard email protocol, they could exchange e-mail. This would be possible, despite the use of two totally different systems. This same principle was used to create standard protocols for file transfers, terminal emulation, printing, network management, and other applications.

2.1.4 From the ARPANET to the Internet

When the original team of researchers decided to conduct their first test of these ideas, they chose four universities as sites: the University of California at Los Angeles (UCLA), the Stanford Research Institute (SRI), the University of California at Santa Barbara (UCSB), and the University of Utah. In September of 1969, these four sites were connected by using 50Kbps (kilobits per second) – leased voice lines, and the resulting network was called the Advanced Research Projects Agency Network, or ARPANET.

Although the original aim of this research was a military one , it was soon used for other purposes. Researchers at the different sites utilized the ARPANET to log into distant sites and to communicate with each other by sending files and electronic mail.

Because the funding for this research was obtained from the U.S. government, and therefore, from U.S. taxpayers, the subsequent technology was considered owned by the U.S. public. And since the government hadn't classified the technology as top secret, it was considered to fall within the public domain. This meant that any individual, organization, or company could receive documentation of the protocols and write programs based on them. This is exactly what happened. Other universities and research and commercial organizations soon began to use this technology in order to create their own networks. Some of these networks were then connected to the ARPANET.

Another factor in the rapid growth of this technology, was the inclusion of the TCP/IP protocols in the Berkeley version of UNIX. The DOD funded two projects that lead to this. First, they had Bolt, Beranek, and Newman modify the TCP/IP protocols to work with the UNIX operating system. Then they had the University of California at Berkeley include them in their version of UNIX, called Berkeley UNIX or Berkeley Software Distribution UNIX (BSD UNIX). Things from Berkeley got around. Because 90 percent of all university science departments were using this version of UNIX, the TCP/IP protocols quickly gained wide usage, and more and more networks were created with them.

Mainframes, minicomputers, and microcomputers all became hardware platforms for TCP/IP protocols. Likewise, software environments from Digital Equipment Corporation (DEC), International Business Machines (IBM), Microsoft, and many others developed products that supported them. Over time, these networks began to connect to each other. Where there was originally only one, the ARPANET, soon there were many separate networks. Eventually, all these individual, interconnected TCP/IP networks were collectively referred to as the Internet, or more simply, the Net.

2.1.5 The Internet Today

Though the numbers increase everyday, the Internet connects about 40 million users worldwide. We commonly use the Internet for sending e-mail. The TCP/IP protocol that relates to this function, is SMTP. As mentioned earlier, this protocol allows people from all over the world, using disparate hardware and software platforms, to communicate with one another.

Another common application of the Internet, is to transfer files. Someone on a Macintosh computer in Iowa can download a file from a minicomputer in Norway. This type of file transfer is accomplished, in part, by the File Transfer Protocol (FTP) running on both machines.

A third frequently used application, is terminal emulation, sometimes called remote log-in. TCP/IP's Telnet protocol allows a user to log in to a remote computer. The computer logging-in acts as, or emulates, a terminal off the remote system; hence, the term "terminal emulation".

2.1.6 TCP/IP

TCP/IP (Transmission Control Protocol/Internet Protocol, or IP, for short) is the name given to a whole group of related protocols, which comprise the language of the Internet. Although there's nothing intrinsically better about TCP/IP relative to better-known LAN protocols, such as Novell's IPX/SPX or Microsoft's NetBEUI, it is rapidly becoming the de facto standard network protocol for one simple reason - the Internet.

IP has gone through multiple versions since its original development. Currently, version 4 is by far the most widely used. However, there are later revisions. Version 5 was never released, but its successor, once termed IPng (IP next generation) but now ratified as IPv6, will gradually replace the current version (IPv4). This is undoubtedly going to constitute a tremendous pain for everyone involved, as the changes are major, but essentially necessary. IPv4 uses 32-bit addresses, allowing for a theoretical maximum of 4,294,967,295 unique addresses.

Clearly, 32-bit addresses won't be enough for very much longer, and this is the driving reason for IPv6, which uses 128-bit addresses, allowing a startlingly vast range of addresses: approximately 3,402,824 x 10^{38}. Estimates vary, but this should be rather more than enough to allow every atom in the universe a unique IP address. Although the other changes between these versions are mostly minor and internal, the two protocols are not directly compatible; though they can share a network; IPv4 nodes and IPv6 nodes cannot directly communicate. Changing from one to the other is, therefore, a substantially duanting task.

2.1.7 IP and the LAN

As the Internet runs over IP, so do Internet-based applications. Proprietary e-mail systems, such as Microsoft Mail, use other protocol-independent means of communication (such as shared file systems). Internet-based e-mail programs communicate over IP, so client machines need an IP-based connection to the server.

For systems which require other protocols, such as older versions of Novell NetWare, it is possible to "tunnel" IP over other protocols, for example, by encapsulating IP packets inside IPX packets. If the client's machine's network stack hides this behind a standard API, such as Windows' WINSOCK, IP-based applications can run unmodified. As all major client and server operating systems today support IP natively, even alongside other protocols, there's little reason to do this, although it may be used for generating secure, encrypted connections over public networks.

2.1.8 How IP works

Building an IP network, requires significantly more planning than when using most other protocols. IP was developed in the 1960's for linking disparate networks - separate in both a geographical sense and in the sense of running different, incompatible systems. Protocols such as IPX and AppleTalk, intended for small LANs, are inherently simpler.

2.1.8.1 Addresses

Each device on an IP network requires a unique address. Unlike in other protocols, this is not automatically generated from the hardware (Media Access Control—MAC) address; it must be manually assigned. The word "device" here is important. It does not mean each computer; IP addresses go by network port. For example, a server with two Ethernet cards (such as a firewall) would need two addresses, one per interface.

Similarly, a machine with both a network card and a modem (or terminal adapter), requires addresses for both. To make matters even worse, it's possible to give one port multiple addresses, a technique called "multihoming". For instance, this allows a single machine to host several separate Web sites; each hostname points to a different address, but all refer to the same machine.

The address is divided into two parts: the network number and the host (or machine) number. All hosts on the same IP network must share the same network number, and no two hosts may share the same host number.

2.1.8.2 Subnet Masks

Alongside the address, each port requires a subnet mask. This value is used to split the complete address into network and host parts; in other words, to determine whether other IP addresses are on the local network or on a remote or foreign one. These two values are the absolute minimum. Using these, a machine will be able to communicate with others on the local network, if the other machine's IP address is known. Additional information is usually required to allow us to be able to access nodes on other networks, to access machines by name rather than number, and so on.

2.1.8.3 Gateways

For direct access to networks beyond the current one, each machine must be told the IP address of the router (or gateway) that connects the local network with the wider world.

2.1.8.4 Name Servers

For a small, server-based network with only one or two servers, access to them by their numeric IP address may be sufficient, but usually it would be desirable to use names instead. The most basic way of doing this is via a local configuration file called "hosts". As a minimum, this file contains a pair of entries per line, separated by spaces; first the address, then the corresponding name. However, for all but the most trivial of networks, keeping all the local files updated rapidly becomes a logistical nightmare, and it is desirable to set up a central server (name server) to resolve names to addresses.

For this, one or more name servers must be set up, and each client machine configured with the name servers' addresses. Name servers accept requests from the clients

containing the name of a machine, such as www.puk.ac.za, and return the matching IP address. The industry standard system for this is the Domain Name Service (DNS).

A DNS configuration is complex and the full functionality is not usually needed for a small LAN. Also, traditional DNS is static and does not cope gracefully with addresses that may change. For this reason, in Windows NT Server (both versions 3 and 4), Microsoft implemented its own proprietary system to deliver basic name-resolution services: the Windows Internet Name Service (WINS). WINS works only with Windows clients. It automatically builds a table of machine names using NetBIOS broadcasts and, with a simple GUI, allows static addresses—for instance, of Unix servers—to be added to the database. Versions of Windows since Windows NT, therefore, expect WINS. Windows for Workgroups pre-dated Windows NT, but the additional 32-bit IP stack for Windows for Workgroups 3.11, followed: this and subsequent versions (such as Windows 95, Windows 98 and Windows NT Workstation) have fields in the configuration dialog for WINS servers. Windows NT even complains if you click the OK button, and these fields are left blank.

2.1.9 Implementation: Address Ranges and Subnets

The first step in building an IP network—or adding IP to an existing system—is to determine the address range to be used. In IPv4, addresses consist of a set of four eight-bit values. As each individual bit can be significant, rather than the value represented by each set of eight, these are, strictly speaking, not bytes but octets. Nonetheless, the four octets of an address or mask are usually written as decimal values, separated by full stops—the dotted-quad notation, such as 193.54.7.18. These numbers are meaningful. Firstly, certain values are reserved and may not be used. The 0 refers to an entire network; for example, 192.168.24.0 means the range of addresses from 192.168.24.1 to 192.168.24.255, and 192.0.0.0 refers to the 192.0.0.1 to 192.255.255.255 range. A machine, therefore, may not be given an address ending in 0. Similarly, 255 is the "broadcast address": a packet sent to 192.169.24.255 will be picked up by all machines in the 192.168.24.0 network. Thus, 255 may not be used in the address.

Secondly, every port on every device on the Internet must have a unique number. Addresses are regulated, with blocks being allocated to organisations by controlling

authorities—the InterNIC's. It is, therefore, "illegal" to just pick numbers out of the air. You should apply to the NIC (or your ISP), giving them an estimate of the future size of your network, and they will allocate a block (or blocks) of addresses to you. These blocks come in three sizes: class A, class B and class C, in diminishing order of size.

Class A ranges use only the first octet to identify the network, and this lies in the range 1 to 127 (ie, 1.0.0.0 to 127.0.0.0); the matching subnet mask is 255.0.0.0 (see Figures 1 and 2). There are 2^{24} (16,777,216) addresses in a class A network. Note that the 127.0.0.0 range is reserved for loopback (the internal logical IP network via which any machine running IP may address itself). Class B ranges use the first two octets for the network number, and the first octet must be in the range 128 to 191; the subnet mask is 255.255.0.0. There are 2^{16} (65,536) addresses in a class B network. C ranges use the first three octets for the network number, and the first octet must be between 192 and 223. There are 2^8 (256) addresses in a class C range. There are also two special classes, which are not normally assigned. The class D range (between 224.0.0.0 and 239.0.0.0) is used for IP multicast, a form of broadcasting. Finally, class E (Experimental) reserves values from 240.0.0.0 to 255.0.0.0, which currently are not used.

Address Class	First Octet	Network Mask
A	1. to 127.	255.0.0.0
B	128. to 191.	255.255.0.0
C	192. to 233.	255.255.255.0
D	224. to 239.	None

Figure 1 – Summary of Internet address classes

	8	16	24	32
Class A	Network Number 1.-127.	Host Number 0.-255.	Host Number 0.-255.	Host Number 0.-255.
Class B	Network Number 128.-191.	Network Number 0.-255.	Host Number 0.-255.	Host Number 0.-255.
Class B	Network Number 192.-223.	Network Number 0.-255.	Network Number 0.-255.	Host Number 0.-255.

Figure 2 –Network and host numbers by class

The most common size is a class C address. This fixes the first three octets, leaving only the last mutable; for instance, 193.54.7.x. As the .0 and .255 host addresses are reserved, this allows for 254 addresses, from 193.54.7.1 to 193.54.7.254. The corresponding subnet mask is 255.255.255.0. Bitwise, it works as shown in Figure 3.

	Octet 1	Octet 2	Octet 3	Octet 4	
Minimum Address	11000001.	00110110.	00000111.	00000000	192.54.7.0
Maximum Address	11000001.	00110110.	00000111.	11111111	192.54.7.255
Subnet Mask	11111111.	11111111.	11111111.	00000000	255.255.255.0
Significant Bits (in the subnet)	00000000.	00000000.	00000000.	11111111	

Figure 3 – Bitwise representation of a class C address

The subnet mask "blanks out" the fixed part of the address (the network number), leaving just the local part (the host number). This, the simplest form of subnet mask, uses all ones or zeros within each octet; thus, subnet boundaries are also octet boundaries.

Applying a subnet mask to an IP address, allows one to identify the network and node parts of the address. The 1's in the mask represents the network bits, and the 0's represent the node bits. Performing a bitwise logical AND operation between the IP address and the subnet mask, results in the *Network Address.*

For example, using our test IP address and the default Class B subnet mask, we get:

10001100.10110011.11110000.11001000	140.179.240.200	Class B IP Address
11111111.11111111.00000000.00000000	255.255.000.000	Default Class B Subnet Mask
10001100.10110011.00000000.00000000	140.179.000.000	Network Address

A network can also be split into sub-units within an octet—so, for instance, dividing a single class C range into two parts. This is where subnet masks can become really useful. The example in Figure 4 translates to a subnet mask of 255.255.255.192 and two address ranges: 192.54.7.64 to 192.54.7.127, and 192.54.7.128 to 192.54.7.254.

	Octet 1	Octet 2	Octet 3	Octet 4
Subnet Mask	11111111.	11111111.	11111111.	11000000.
Subnet 1	11000000.	00110110.	00000111.	01xxxxxx.
Subnet 2	11000000.	00110110.	00000111.	1xxxxxxx.

Figure 4 – Dividing a single class C range into two parts.

Say you are assigned a Class C network number of say 200.133.175.0. You want to utilize this network across multiple small groups within an organization. You can do this by subnetting that network with a subnet address.

We shall break this network into 14 subnets of 14 nodes each. This will limit us to 196 nodes on the network instead of the 254 we would have without subnetting, but gives us the advantages of traffic isolation and security. To accomplish this, we need to use a subnet mask 4 bits long. Recall that the default Class C subnet mask is

255.255.255.0 (11111111.11111111.11111111.00000000)

Extending this by 4 bits yields a mask of

255.255.255.240 (11111111.11111111.11111111.11110000)

This gives us 16 possible network numbers, 2 of which cannot be used:

Subnet bits	Network Number	Node Addresses	Broadcast Address
0000	200.133.175.0	Reserved	None
0001	200.133.175.16	.17 thru .30	200.133.175.31
0010	200.133.175.32	.33 thru .46	200.133.175.47
0011	200.133.175.48	.49 thru .62	200.133.175.63
0100	200.133.175.64	.65 thru .78	200.133.175.79

0101	200.133.175.80	.81 thru .94	200.133.175.95
0110	200.133.175.96	.97 thru .110	200.133.175.111
0111	200.133.175.112	.113 thru .126	200.133.175.127
1000	200.133.175.128	.129 thru .142	200.133.175.143
1001	200.133.175.144	.145 thru .158	200.133.175.159
1010	200.133.175.160	.161 thru .174	200.133.175.175
1011	200.133.175.176	.177 thru .190	200.133.175.191
1100	200.133.175.192	.193 thru .206	200.133.175.207
1101	200.133.175.208	.209 thru .222	200.133.175.223
1110	200.133.175.224	.225 thru .238	200.133.175.239
1111	200.133.175.240	Reserved	None

2.1.9.1 Private Ranges

The next step, is to choose the range of addresses you will use. The "official" way to do this, as mentioned earlier, is to apply to a NIC for a range. In practice, it's now more common for you to be allocated one by your ISP, which has already purchased a whole set of ranges. Unfortunately, many people implementing IP don't know this and just make up a range, such as 100.100.100.0. This will work as long as the network isn't directly connected to the Internet. However, if—or when—it is, a working configuration suddenly goes wrong. As this range isn't private, there may be real hosts out there somewhere on the Internet using these addresses, and a local server address of 100.100.100.54 suddenly also points to another machine somewhere else in the world. Depending on how the Internet connection works, things start to fail. At best, when the link is open, machines on the internal network can no longer access that server—an intermittent fault, and those are always the hardest to trace. At worst, the server itself may detect a clash of IP addresses and fail.

Happily, it is not strictly necessary to reserve a range. The designers of IP anticipated this problem and set aside blocks of addresses for internal networks—the private ranges. There are three private ranges: one class A, one class B and one class C (see Figure 5). All you need to do, is to choose the one of appropriate size for your network. For most small LANs of fewer than 255 machines, the private class C range is the best, even though the private class A range of I0.x.x.x is easier to remember. As these

addresses are reserved as private, no hosts on the Internet will ever use addresses in any of these ranges. Similarly, the main routers on the Internet backbone will not pass packets with such addresses. There will be many other networks using the same ranges, but they can never clash with one another.

If an illegal range is used, it's not necessarily the end of the world. There are ways around it—either avoiding a routed connection between the network and the Internet, or using a smart router, which can translate on-the-fly between illegal internal addresses and legal external ones, a technique called Network Address Translation (NAT). Use of NAT is actually commonplace, although usually for security reasons, rather than to repair earlier mistakes.

Today, intermediate networks (ones of between a few hundred to a few thousand hosts) are being allocated multiple class C (256-address) ranges rather than single class B (65,536-address) ones. This is because the total IPv4 address space is rapidly filling up. In the early days, companies were readily assigned class A ranges—in other words, their own first octet. Although there are fewer than 255 possible class A ranges, there probably aren't that many companies in existence, which really require sixteen million machines visible on the Internet! Thus, vast ranges of potential addresses were effectively wasted, and efforts are afoot to make best use of the remaining space. Similarly, if your network is unlikely to ever exceed 255 machines, don't use the private class A or class B ranges unnecessarily. If you need to link up multiple networks into a WAN and you are using private ranges, you don't need a single big range to embrace them all, unless there are more than 255 of them. It's preferable to use multiple private class C ranges and alter the third octet—for instance, the Johannesburg office might use 192.168.1.0 and Pretoria 192.168.2.0.

It's not usually a good idea to link private company LAN's over the public Internet, for obvious reasons. For simple point-to-point links, either over ISDN or permanent by leased lines, it doesn't matter what ranges you are using. However, if you wish to make a routed connection between a private network and the Internet, you will need to use routers that support NAT. For security and performance, in any case, it's generally preferable to use proxy servers, firewalls, or both.

Class	Start of Range	End of Range	Subnet Mask
A	10.0.0.0	10.255.255.255	255.0.0.0
B	172.16.0.0	172.31.255.255	255.255.0.0
C	192.168.0.0	192.168.255.255	255.255.255.0

Figure 5 – The private address ranges

2.1.10 Address Allocation

Once you have chosen the address range (or ranges) that you will use, the next task is allocating addresses to individual machines. The simplest way to perform this, is just to go to each machine and configure it with its address—which is fine if there are only a handful of machines to set up. However, most server-centric networks are larger than this, with only a few machines that are accessed by all the rest. For such purposes, the addresses of the servers must be known to all machines, but those of individual workstations are irrelevant, as other machines will not routinely be connecting to them. This means, the servers need to have static addresses (ones which are permanent) but workstations need not: their addresses can be given to them when they boot-up by a program running on a server. When a workstation shuts down or reboots, its address can then be released back into a pool of available addresses, and may later be given out to another machine when it boots.

This system relieves a great deal of the administrative burden. Rather than maintaining a list of all the addresses on the network and visiting each machine to set its address, you need only set a few fixed addresses, and then set up a server to dynamically allocate addresses to workstations from a predefined range.

2.1.10.1 Allocation Protocols

For many years, Unix used a basic system for allocating IP addresses at system startup: the Boot Protocol (BOOTP). Like the simplified Windows-only name service, WINS, there's a simpler system, the Dynamic Host Configuration Protocol (DHCP), originally included with Windows NT Server. Unlike WINS, though, this isn't a Microsoft-only standard, and it is being widely adopted by other PC operating systems such as NetWare, MacOS, Linux and BeOS. DHCP is a superset of the older BOOTP system, which in time it will probably replace.

With DHCP, configuration is reduced to more or less the minimum level currently possible. The server needs to be told only the address range or ranges to put in the pool, and the client that it only should use DHCP to find its address. There's no need to tell the server the MAC addresses of the clients it will handle, or to tell the clients the address of the server: everything else happens automatically, DHCP doesn't allocate addresses and subnet masks: it can also be used to inform clients of the location of name servers (both WINS and DNS) and gateways. DHCP servers are included with Windows NT Server, Linux and recent versions of NetWare. However, Windows NT Workstation, Windows 95 and 98 and MacOS do not, though third-party ones are available.

2.1.11 Packets

An IP packet consists of the IP header and data. The header includes a 4-bit protocol version number, a header length, a 16-bit total length, some control fields, a header checksum and the 32-bit source and destination IP addresses. This totals 20 bytes in all.

The detail of all the IP control fields will not be discussed in detail. However, the protocol field is important. It identifies which higher-level TCP/IP protocol has sent the data. When data arrive at the destination (either the packet's destination address equals the host's own IP address, or it is a broadcast address), this field tells IP which protocol module to pass it on to.

One control field, the Time-to-live (TTL) field, is interesting. It is initialised by the sender to a particular value, usually 64, and decremented by one (or the number of seconds it is held on to) by every router that the packet passes through. When it reaches zero, the packet is discarded and the sender notified by using the Internet Control Message Protocol (ICMP), a network layer protocol for sending network-related messages. The TTL field is a safety mechanism, which prevents packets from travelling the Internet forever in routing loops.

Although the total field length in the IP protocol header is 16 bits, IP packets are usually much smaller than the 64 KB maximum this implies. For one thing, the link layer will have to split this into smaller chunks anyway, so most of the efficiency advantages of sending data in large blocks, is lost. For another, IP standards did not historically require a host to accept a packet of more than 576 bytes in length. Many TCP/IP applications limit themselves to using 512-byte blocks for this reason, though today most implementations of the protocol aren't so much restricted.

2.1.12 IP Routing

So how does an IP packet addressed to a computer on the other side of the world, find its way to its destination? The basic mechanism is very simple.

On a LAN, every host sees every packet that is sent by every other host on that LAN. Normally, it will do something with that packet only if it is addressed to itself, or if the destination is a broadcast address.

A router is different. A router examines every packet, and compares the destination address with a table of addresses that it holds in memory. If it finds an exact match, it forwards the packet to an address associated with that entry in the table. This associated address may be the address of another network in a point-to-point link, or it may be the address of the next-hop router.

If the router doesn't find a match, it runs through the table again, this time looking for a match on only the network ID part of the address. Again, if a match is found, the packet is sent on to the address associated with that entry. If a match still isn't found, the router looks to see if a default next-hop address is present. If so, the packet is sent there. If no default address is present, the router sends an ICMP "host unreachable" or "network unreachable" message back to the sender. If this message appears, it usually indicates a router failure at some point in the network.

The difficult part of a router's job, is not how it routes packets, but how it builds up its table. In the simplest case, the router table is static: it is read in from a file at start-up.

This is adequate for simple networks. You don't even need a dedicated piece of kit for this, because routing functionality is built into IP.

Dynamic routing is more complicated. A router builds up its table by broadcasting ICMP router solicitation messages, to which other routers respond. Routing protocols are used to discover the shortest path to a location. Routes are updated periodically in response to traffic conditions and availability of a route.

2.1.13 Domain Names

2.1.13.1 Introduction

The domain name system is a global network of servers that translate host names like www.pukke.co.za into numerical IP addresses, like 204.62.131.129, which computers on the Net use to communicate with each other. Without DNS, we'd all be memorizing long numbers instead of intuitive URL's or e-mail addresses; and that wouldn't be much fun, would it?

2.1.13.2 History of DNS

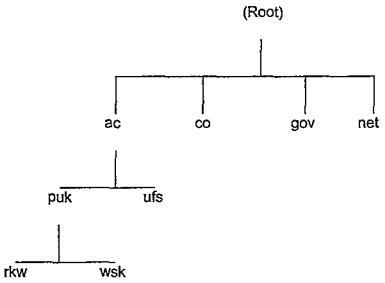
Paul Mockapetris designed DNS in 1984, to solve escalating problems with the old name-to-address mapping system. The old system consisted of a single file, known as the host table, maintained by the **S**tanford **R**esearch **I**nstitute's **N**etwork **I**nformation **C**enter (SRI-NIC). As new host names were requested, SRI-NIC would add them to the table—a couple times a week. Systems administrators would request the newest version (via FTP) and update their domain name servers.

But as the Internet grew, the host table became unwieldy. Though it worked fine for name-to-address mapping, it wasn't the most practical or effective way to update and distribute the information. And since the stability of the rapidly growing Internet was at stake, Mockapetris and others decided to find a better way.

No single organization is responsible for updating the domain name system. It's what's known as a "distributed database"; it exists on many different name servers around the world, with no one server storing all the information. Because of this, DNS allows for almost unlimited growth.

2.1.13.3 The domain name space

In order to understand how a DNS server works, one should be familiar with what is called the domain name space. It sounds a little ominous, but it's quite simple. In fact, you've probably seen it at one time or another represented by an inverted tree that looks something like this:



Each node on the tree represents a domain. Everything below a node, falls into its domain. One domain can be part of another domain. For example, the machine *rkw* is part of the *.puk* domain as well as the *.ac* domain. You'll see why this is important in just a minute.

2.1.13.4 How it works

A DNS server is just a computer that's running DNS software. DNS software is generally made up of two elements: the actual name server, and something called a resolver. The name server responds to browser requests by supplying name-to-address conversions. When it doesn't know the answer, the resolver will ask another name server for the information.

To see how it works, let's go back to the domain-name-space inverted tree. When you type in a URL, your browser sends a request to the closest name server. If that server has ever fielded a request for the same host name (within a time period set by the administrator to prevent passing old information), it will locate the information in its cache and then reply.

If the name server is unfamiliar with the domain name, the resolver will attempt to "solve" the problem by asking a server farther up the tree. If that doesn't work, the second server will ask yet another—until it finds one that knows. (When a server can supply an answer without asking another, it's known as an "authoritative server".)

Once the information is located, it's passed back to your browser. Usually this process occurs quickly, but occasionally it can take an excruciatingly long time (like 15 seconds). In the worst cases, you'll get a dialog box that says the domain name doesn't exist—even though you know it does. This happens because the authoritative server is slow in replying to the first, and your computer times-out (drops the connection). But if you try again, there's a good chance it will work, because the authoritative server has had enough time to reply, and your name server has stored the information in its cache.

2.1.14 Setting up Name Servers

There are currently two main standards for IP name resolution: DNS, which is cross-platform, and WINS, which is Windows-only. However, this changes with the advent of Windows 2000, which subsumes WINS into an enhanced dynamic DNS-compatible system—something which may prove to be a significant driver towards adoption of Windows 2000. On Windows NT Server versions 3 and 4 WINS integrate closely with DHCP, and DNS is peripheral. Configuring a WINS server is almost as simple as configuring a DHCP one: all that needs to be done, is to tell the server the domain name, add entries for any fixed addresses, and the server does the rest, automatically building a database by "scavenging" traffic for machine node names and their associated addresses.

DNS servers are complex and difficult, and describing the set-up and configuration of them needs an article—or possibly book—to itself. There is only space here for the bare essentials. On Unix systems, DNS is usually implemented by using the open source BIND program, but others are available, including DNS servers for NetWare and NT. The most basic kind of DNS server, is a DNS proxy. This simply takes DNS requests from the local network and forwards them onto the ISP's name server; it maintains no database of its own whatsoever, so repeated requests for the same address will generate repeated look-ups, including bringing up the connection if it is not already open. Proxy servers such as Wingate (www.wingate.net) and Mailgate

(www.mailgate.com), often used to provide external web access for a LAN, frequently include a simple DNS proxy.

Since even requests for local nodes will cause a DNS proxy to query the ISP's servers, it is often desirable to run a more capable DNS server as well, to handle internal requests. A basic, but nonetheless useful DNS server for Windows, is SimpleDNS by Jesper Hoy (www.jhsoft.com). This looks up client requests in the local hosts file, which reduces administration to maintaining a single version; clients need be told only the address of the machine running the DNS server, either via local configuration or DHCP. If the proxy server itself uses the ISP to resolve external addresses, this is all that's needed, and such a server can significantly reduce the number of connections to the ISP. Better performance can be achieved by running a DNS caching proxy. This has no local database, but when an address is resolved by using the ISP's servers, the name and address are kept in memory. After a period, all commonly used addresses can be supplied locally without recourse to the ISP, improving response time and reducing the number of calls. A low-specification machine running Linux and BIND is ideal for this.

After this, DNS configuration becomes more complex, as servers maintain part of a database and also refer to higher-level servers—up to the top-level master servers maintained by Network Solutions in the USA, which control the top-level domains (TLDs) such as .com

2.1.15 Gateways

It is not always necessary to provide a routed connection between a LAN and the outside world. For single-site networks, a proxy and e-mail server can provide Web access, ftp access and e-mail forwarding, without routing. Here, the proxy is the only machine connected to the Internet, and IP packets never travel between the LAN and the Internet. However, for multi-site WANs or direct Internet access, a gateway machine must be set up to route packets from the LAN to elsewhere. This may be a dedicated router, or a machine with a server OS (such as NT Server, NetWare or Linux) running a software router as a process. In the example shown in Figure 6 there are two separate sub-networks, 192.168.1.0 and 192.168.2.0. The gateway (gateway.puk.ac) has two network connections, attaching it to both networks; in the 192.168.1.0 subnet, it appears as 192.168.1.254, and in the 192.168.2.0 subnet, it

appears as 192.168.2.254. Some choices are arbitrary, or purely matters of convenience. For instance, in this example the subnets are not given separate names, although they could be; one might be vaal.puk.ac and the other potch.puk.ac. The hostnames would then be as shown in Figure 7.

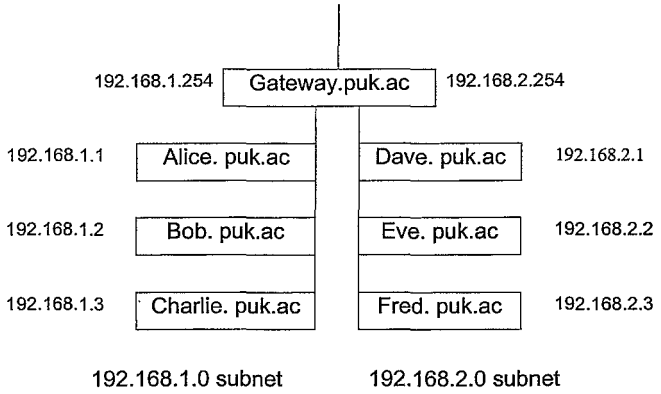


Figure 6 – A gateway linking two subnets

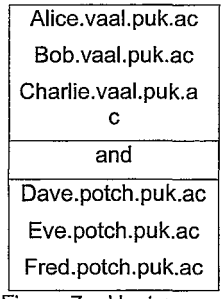


Figure 7 – Host names

Gateways are often (but by no means always) given address 0.0.0.254 in each of their networks. They may be given hostnames, but as they are usually referred to by address, this is not necessary.

3 IT: Ornament?

3.1 Inleiding

In die vorige afdeling is aandag gegee aan konsepte rakende die Internet, met die doel om aan te toon dat inligtingstegnologie 'n instrument in die hande van die meeste gebruikers is.

3.2 Uitdagings

Daar bestaan verskeie uitdagings wat tans as probleme geidentifiseer kan word. Om bloot maar net 'n gebruiker van IT te wees en te bly impliseer dat die universiteit en dus die land nooit 'n leier op die gebied van IT sal wees nie. In hierdie konteks sal IT altyd as 'n ornament gesien en gebruik word. Enkele uitdagings word vervolgens bespreek.

3.2.1 IT as 'n professionele dissipline

Volgens Denning (2001:1) verwag gebruikers, dat professionele IT – mense hulle sal bystaan in hulle behoeftes aangaande die ontwerp, verkryging, onttrekking, gebruike, konfigurering, programmering, onderhoud en die verstaan van rekenaars, netwerke, toepassings en digitale objekte. Studente verwag dat IT-kurrikulums voorsiening sal maak vir uitgebreide dekking van tegniese, navorsing en leierskapbeginsels en praktyke om sodoende effektiewe professionele mense van hulle te maak. Hulle verwag van die universiteit om hulle te voorsien van 'n uitgebreide siening (view) van die vinnig veranderende, gefragmenteerde wêreld, vir bystand om belangrike vrae te begrens en te beantwoord en vir opleiding in standaard professionele praktyke. Professionele mense verwag van hulle verenigings om hulle identiteit as professionele mense te ondersteun, om lewenslange voortgesette professionele leer aan te moedig, en om in die openbaar standpunt in te neem oor sake rakende IT.

Wat gebruikers, studente en professionele mense in werklikheid ondervind, voldoen beslis nie aan hulle verwagtinge nie. Hulle word gedurig gekonfronteer met swak-ontwerpte pakette, komplekse en verwarrende stelsels, stelsels wat onverklaarbaar reageer, stelsels sonder waarborge, swak tegniese ondersteuning, miskenning van privaatheid, ensovoorts.

Organisasies ondervind probleme om gekwalifiseerde IT persone te kry en hulle op die hoogte te hou t.o.v. 'n fenomenaal snel veranderende omgewing. Studente vind IT

kirrikulums wat meer fokus op programmering as op stelsels, op teorie eerder as eksperimentering, en op konsepte eerder as op praktyke. Professionele mense vind min ondersteuning vir lewenslange voortgesette leer of beroepsbevordering en baie konflikterende stemme van professionele groepe. Dertouzos is van mening, dat gebruikers, wat die grootste groep vorm, moeg is vir hierdie probleme. Hulle verwag dat IT-mense hulleself meer effektief moet organiseer om probleme aan te spreek en 'n beter diens aan hulle kliënte te lewer. Waarom gebeur dit nie? Is ons nie besig om IT as 'n ornament te beskou nie?

3.2.1.1 Wat karakteriseer 'n professionele dissipline?

Denning is van mening, dat die volgende eienskappe 'n professie kenmerk.

- A durable domain of human concerns
- A codified body of principles (conceptual knowledge)
- A codified body of practices (embodied knowledge)
- Standards for performance
- Standards for ethics and responsibility.

3.2.1.2 Waar staan IT in terme van bogenoemde kenmerke ?

- Durability. This criterion is clearly met.
- Body of principles. This criterion is clearly met.
- Body of practices. This criterion is not met.
- Standards of performance. This criterion is not met.
- Ethics and responsibility. This criterion is partially met.

3.2.2 Studente

Denning (2001:1a) stel dit, dat vir 'n geruime tyd "... IT professionals have been grappling with four dilemmas". Hierdie vier probleme word soos volg beskryf.

- "The IT skills dilemma"

  Werknemers is van mening dat dat IT-gegradueerders nie oor die nodige vaardighede beskik wat in die werksplek benodig word nie. Die belangrikste hiervan, is kennis t.o.v. die nuutste inligtingstegnologieë en sogenaamde

"sagte" vaardighede, soos aanbiedingsvaardighede, kliëntverhoudings, leierskap en spanwerk. Andersyds is werknemers tevrede met die waarde van algemene, beginsel-gebaseerde opleiding wat universiteite bied en elke gegradueerde word opgeraap deur die indistrieë. Is dit nodig dat universiete hulle kurrikilums moet verander?

"The breath versus depth dilemma".

Dit wil voorkom asof die mark gegradueerdes benodig wat oor diepgaande kennis beskik in 'n tegniese spesialiteitsveld, maar wat terselfdertyd ook 'n breë kennis het van die IT-veld. Daar is so baie spesialiteitsvelde, dat een enkele departement net een of twee sulke velde kan dek. Die breedte-versus-diepte probleem is ook baie organisasie- (selfs net 'n individuele keuse) gebonde. In watter area word groter diepte benodig? Oor watter spektrum soek 'n persoon 'n breër opleiding?

"The design dilemma".

Bestaande sagteware ontwerpprosesse lewer stelsels met foute oor 'n breë spektrum. Hierdie stelsels is dan baie onbetroubaar en soms moeilik om te gebruik. Die gebruiker kan nie teen hierdie kompleksiteite beskerm word sonder 'n fundamentele verandering in die ontwerpproses nie. Daar is verskeie outeurs wat aanbeveel dat daar beweeg moet word van 'n tegnologie-gesentreerde na 'n mens-gesentreerde ontwerp-benadering. Ons onvermoë om sagteware te ontwikkel wat die gebruiker se spesifikasies en tot bevrediging van die gebruikers is, het gelei tot ontevredenheid met kommersiële pakette wat van die rak af gekoop kan word. Hierdie stelsels is soms baie kompleks, ingewikkeld en oorlaai met eienskappe wat nie noodwendig nuttig is nie, terwyl ander belangrike en nuttige eienskappe nie daarin voorkom nie. Verder is die tegniese rugsteuning nie na wense nie.

"The licensing dilemma".

Sagteware ingenieurs is nie eenstemmig oor die waarde van die lisensiëring van sagteware ingenieurs wat kritieke stelsels ontwikkel nie. Daar is veskeie voor- en nadele hieraan verbonde.

Die eerste twee probleme wat Denning uitwys, handel oor vaardighede in die werksplek, asook die spektrum van kennis waaroor die student uiteindelik moet beskik.

"*In general, most scientists with a mathematics-based training are trained problem solvers. If they are also willing to learn the computer basic, they too can become comprensively computer literate*" Venter (2001:9). 'n Ondersoek wat oor die afgelope 4 jaar gedoen is, toon duidelik, dat daar 'n probleem is met die gehalte van onderrig wat studente op skool ontvang. 'n Toets wat uit 25 probleme bestaan, is aan eerste- en tweedejaar rekenaarwetenskapstudente by drie verskillende universiteite, naamlik die Vaaldriehoekkampus van die PU vir CHO, QwaQwakampus van die universiteit van die Noorde en Medunsa, gegee om te voltooi. Daar is geen tydsbeperking op die toets geplaas nie. Tabel 1 toon die uitslag van die ondersoek.

Wanneer 'n mens die vrae in die toets bestudeer, is dit duidelik dat die student baie logies moet kan redeneer, drie-dimensioneel moet kan waarneem en patrone moet kan herken. Hierdie is tipiese eienskappe wat van IT-personeel verwag word, aangesien hulle probleme moet kan identifiseer, ontleed en oplossings daarvoor bied. Probleemoplossing en die logiese, sistematiese uiteensetting van moontlike oplossings (en die uiteindelike keuse van die "beste" oplossing), is belangrikste aspekte in die mondering van die professionele IT-persoon.

Die skokkende van hierdie ondersoek is, dat die probleme uit die graad 6 Wiskunde Olimpiade geneem is! Soos reeds genoem, is die studente almal ingeskrewe rekenaarwetenskapstudente. In rekenaarwetenskap moet probleme opgelos word wat logiese denke, drie-demensionele waarneming en patroonherkenning verg. Indien studente probleme ondervind met hierdie konsepte, sal hulle probleme ondervind met probleemoplossing.

Hierdie probleem van studente wat nie op standaard is nie, is 'n kritieke verskynsel wat dringend aandag sal moet kry. Indien daar nie 'n voldoende stroom van "sterk" IT-studente na universiteite vloei nie, sal Suid-Afrika nooit aan sy eie behoeftes vir professionele IT persone kan voldoen nie. Nagraadse studie word hierdeur baie beperk en dit lei weer tot stremming op navorsing. Dit alles sal meebring dat Suid Afrika 'n IT-

gebruiker sal bly en nooit 'n wêreldleier op hierdie gebied sal word nie, ten spyte van die feit dat die land oor die nodige kundiges beskik.

Vraag	QwaQwa 1 1998	Medunsa 1 2001	VDK 1 1998	QwaQwa 2 2000	Medunsa 2 2000
1	68	93	93	100	100
2	68	81	68	50	100
3	25	68	65	60	33
4	25	25	34	70	83
5	25	31	51	10	0
6	62	31	62	20	66
7	50	50	62	20	83
8	37	50	55	90	66
9	25	6	31	10	16
10	37	43	44	60	33
11	6	12	41	10	0
12	12	12	20	10	33
13	81	87	86	70	100
14	0	18	62	30	50
15	0	18	20	0	50
16	93	87	79	90	100
17	81	93	82	100	100
18	37	53	58	50	83
19	18	0	37	40	16
20	18	6	6	40	16
21	18	18	17	10	50
22	18	43	44	30	66
23	37	50	51	50	83
24	68	62	68	80	83
25	31	31	41	30	33
Gemiddeld	**38**	**42**	**51**	**45**	**58**

Tabel 1 – Gemiddeldes per vraag

3.2.3 Personeel

In teenstelling met Steyn (2000:12) se bevinding aan die Potchefstroomkampus is die personeelomset by die Vaaldriehoekkampus baie hoog. In die agtien jaar vanaf 1983 tot 2001 was twaalf dosente en twee laboratoriumassistente in permanente poste aan die Departement Rekenaarwetenskap betrokke. Baie tyd en energie word aan

personeel spandeer om hulle kwalifikasies te verbeter, net om hulle te verloor vir beter salarisse, gespesialiseerde werksomstandighede, ensovoorts.

Tabel 2 gee 'n oorsig van die doseerlas aan by die Vaaldriehoekkampus. Let op dat daar 'n vakante pos is en dat een van die twee statistiek-lede aan die einde van 2002 aftree. Verder is een van die IT-personeellede ook die Direkteur van die skool, wat uit die aard van sy verpligtinge nie 'n volledige doseerlas kan hanteer nie. Verder word die volledige IT-program voltyds sowel as deeltyds aangebied, wat die doseerlas verder verhoog. Tabel 2 sluit nagraadse verpligtinge uit.

Personeel	Vak	Modules	Semester 1	Semester 2
2	Statistiek	5	1	4
4 (+1 Lab. Ass.)	IT RINL111	19 1	10 1	9
2	Wiskunde	9	5	4
1 Vakant				

Tabel 2 – Doseerlas van personeel in die skool vir Modelleringswetenskappe

3.2.4 Navorsing

Navorsing op die gebied van IT in die skool vir Modelleringswetenskappe en in die vakgroep Rekenaarwetenskap en Inligtingstelsels, is gladnie wat dit behoort te wees nie. Verskeie redes kan hiervoor aangevoer word, soos byvoorbeeld die hoë personeel-omset, hoë doseerlas, ensovoorts. Die uitdaging lê beslis nie in verskonings nie, maar in oplossings wat hierdie probleem daadwerklik sal aanspreek.

Personeeltekorte en redes daarvoor, is reeds bespreek. 'n Ander knelpunt wat dringend aandag moet geniet, is die navorsingsomgewing. Deur die jare is 'n groot agterstand opgebou, deurdat apparatuur en programmatuur so vinnig verander het, en steeds vinniger verander, maar dat befondsing vir navorsing het nie hiermee tred gehou nie.

3.2.4.1 Strategiese posisionering

Om die navorsingsprobleem aan te spreek, moet daar opnuut na die plek en rol van IT binne die universiteit kyk word. Daar kan byvoorbeeld gekyk word na raakvlakke tussen

die strategiese doelwitte van bestuur en doelwitte wat akademici ten opsigte van IT stel. Verder moet besin word oor strategiese raakvlakke tussen ondersteuningsdepartemente soos ITB en akademiese departemente.

4 Samevatting

Volgens die HAT is 'n instrument 'n konkrete hulpmiddel met behulp waarvan die een of ander taak verrig word. Vir die mense wat hierdie hulpmiddel nodig het om hulle taak beter te vervul, geniet IT en gebruik dit tot sy volle potensiaal

'n Ornament, volgens die HAT, is 'n versiering, 'n versiersel of 'n sieraad. Kom ons ondersteun en motiveer die mense wat verantwoordelik is vir die daarstelling van bogenoemde instrument, om hulle in staat te stel om hulle taak ten volle te kan uitvoer – anders kan IT dalk net 'n ornament wees en bly.

5 Bedankings

Die Raad en Bestuur: My dank aan die raad en bestuur van die PU vir CHO. Ek onderneem om my taak na die beste van my vermoë uit te voer.

Almal teenwoordig: Baie dankie vir u teenwoordigheid, dit word opreg waardeer.

Kollegas: Baie dankie vir julle ondersteuning oor baie jare. Ons kan 'n verskil maak.

Werkers: Almal wat meegewerk het om hierdie aand 'n sukses te maak.

My moeder: Dankie vir al die ondersteuning, opofferings en geduld. Dit bly 'n voorreg om u kind te wees.

My gesin: Dankie vir die ondersteuning en opofferings wat julle maak. Ek is baie lief vir julle.

Ons Hemelse Vader: Aan U kom al die dank, lof en eer toe. In U lig sien ons die lig!

6 Verwysings

1. Denning P.J. September 26, 2001. *The Core of a Third-Wave Professional.* CACM Column: IT Profession. Internetdokument (www.denning.com)
2. Denning P.J. August 2001. *When IT becomes a Profession.* The Invisible future. McGraw Hill.
3. Denning P.J. August 1998. *Computing the Profession.* (This essay has been prepared for the book Computer Science and Engineering Education, Toby

Greening, Editor, and for Educom Review, John Gehl, Editor). Internetdokument (www.denning.com)

4. Denning P.J. July 1999. Computer Science: The Discipline. Internetdokument (www.denning.com)
5. Lammle T. 1998. *TCP/IP for NT Server 4.* Sybex Computer Books INC., USA.
6. Morton D. May 1997. *Understanding Ipv6.* PC Network Advisor, 83, 17-22.
7. Moss J. September 1997. *Understanding TCP/IP.* PC Network Advisor, 87, 3-6.
8. Parsons J.J. and Oja D. 1998. *Computer Concepts*, 3rd edition. Thompson Publishing Company.
9. Proven L. January 2001. *Understanding TCP/IP.* PC Network Advisor, 126, 9-16.
10. Steyn T. 2000. Rekenaarwetenskap & Inligtingstelsels: 'n Situasie-Analise. PU vir CHO. Potchefstroom. Wetenskaplike Bydraes. Reeks H: Inougurele Rede nr. 158.
11. Denning P.J. August 2001. *When IT becomes a profession.* The Invisible future. McGraw Hill.
12. Venter L.M. 19 February 2002. *Are all Computer Scientists computer literate?* Inaugural Lecture presented at UNISA.
13. Washburn K, Evans J.T. 1995. *TCP/IP Running a Successful Network.* Addison-Wesley.