Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Burkhard Stiller, Jan Gerke (Edt.)

PPS-Seminar: Grundlagen der Internet-Technologie 1

TIK-Report
Nr. 90, Juli 2000

Institut für Technische Informatik und Kommunikationsnetze
Computer Engineering and Networks Laboratory

Burkhard Stiller, Jan Gerke (Edt.):
PPS-Seminar: Grundlagen der Internet-Technologie 1
Juli 2000
Version 1
TIK-Report Nr. 90

Computer Engineering and Networks Laboratory,
Swiss Federal Institute of Technology (ETH) Zurich

Institut für Technische Informatik und Kommunikationsnetze TIK,
Eidgenössische Technische Hochschule Zürich

Gloriastrasse 35, ETH-Zentrum, CH-8092 Zürich, Switzerland

PPS-Seminar: Grundlagen der Internet-Technologie

Einleitung

Studierende des Departements für Elektrotechnik, die die Grundlagen und erste wichtige Begriffe eines weitverbreiteten Netzwerkes – dem Internet – erlernen möchten, wurden in diesem Seminar angesprochen.

Das Seminar vermittelte dabei wichtige und wesentliche Grundlagen für allgemeine Kommunikationstechnologien am Beispiel des Internet. Dabei wurden u.a. die folgenden Fragen aufgeworfen und Antworten hierzu gegeben: was ist ein Netzwerk, was bezweckt die Adressierung, wie funktioniert die E-Mail, welche Protokolle und Sprachen gibt es im Web, was ist IP-Telefonie, wie werden drahtlose Web-Zugriffe möglich? Ferner verarbeitete das Seminar diese Grundlagen weiterführende Details am gleichen Beispiel: was ist die Internet-Architektur, welche Protokolle gibt es, welche Rolle spielt die nächste Generation der Internet-Protokolle, welche Entwicklungstendenzen zeigen sich? Insbesonders wurden einige Themen behandelt, die mit dem Auftritt des Internet als Daten- und Informationspräsentationsmedium zusammenhängen, u.a. das HTTP-Protokoll, die Beschreibungssprache HTML sowie die Datenstrukturierungssprache XML.

Ablauf

Die Studierenden erarbeiteten zu den zwölf vorgegebenen Themen (siehe unten) eigene schriftliche Zusammenfassungen, die in diesem TIK-Report zusammengestellt sind. Diese Ausarbeitungen basieren auf teilweise bereitgestelltem Material sowie Literatur, die die Studierenden in eigener Arbeit ermittelt und erarbeitet haben. Neben dieser schriftlichen Arbeit hielt jeder Studierende einen Vortrag im Rahmen des Seminares, welcher zum Ziel hatte, in begrenzter Zeit das erarbeitete Wissen den Zuhörern nahezubringen, zu erläutern und zeigen zu können, daß selbständig erarbeitetes Wissen gut aufbereitet und verständlich präsentiert werden kann. Ein nachfolgende Diskussions- und Fragephase erlaubte das interaktive Behandeln von Unklarheiten, offenen Fragen sowie die Verküfung von Themen.

Vorträge und Inhalte

Vortrag 1: Grundlagen des Internet
Vortrag 2: Netzwerktechnologien für das Internet
Vortrag 3: IP, Adressierung und Routing im Internet
Vortrag 4: IPng – Die nächste Generation
Vortrag 5: Elektronische Post im Internet
Vortrag 6: Das HTTP-Protokoll des Web
Vortrag 7: Die Beschreibungssprache HTML
Vortrag 8: Sichere Kommunikation – SSL, SHTTP
Vortrag 9: Die Datenstrukturierungssprache XML
Vortrag 10: Server für HTTP
Vortrag 11: IP Telefonie
Vortrag 12: Drahtlose Kommunikation – WAP

Vortrag 1

Grundlagen des Internet

Sandro Ribolla

1. Einführung

Das Internet ist sicher eine der faszinierendsten Entwicklungen unserer Zeit. Anfänglich zu militärischen Zwecken verwendet und kaum beachtet, dann im akademischen Umfeld eingesetzt, ist es zwischenzeitlich zu einer Technologie angewachsen, welche aus unserem Alltag nicht mehr wegzudenken ist. Es ist die Realisierung des Wunsches nach einer einheitlichen Kommunikation. Das Internet ist eine Revolution. Man erhoffte von ihr, dass sie die sozialen Schranken aufhebe, die Standortvorteile der Zentren zunichte mache und die Drittweltländer zu den Industriestaaten nachrücken können. Die Realität war aber ernüchternd. Es war wie immer: die Reichen wurden noch reicher, die Armen noch ärmer, denn die modernen Kommunikationsmittel stehen in den USA und nicht in Afrika.
Das Internet bekam die Funktion eines Einkaufszentrum, eines Fliessbandersatzes, sowie einer unersättlichen Wissensstätte. Doch das Internet birgt auch Gefahren. Man spricht heute schon zum Teil von einem Synonym für Pornographie. Rund 80 Prozent des gesamten Datentransfers im Netz betreffen pornographische Inhalte, behaupten Fachleute. Auch Viren verursachen den Fachleuten Kopfschmerzen. 1983 entdeckte man in Amerika einen Virus, welcher schliesslich 2600 Computer infizierte, 10 Prozent aller damals am Internet angeschlossener Rechner.

2. Wie funktioniert das Internet?

Häufig werden auch die beiden Begriffe des *Internet* und des *Worl Wide Web* durcheinander gebracht. Um hier Klarheit zu schaffen, sind hier zwei Definitionen angebracht:

- **Internet** - Das Internet ist die Gesamtheit aller Computer, die das Paket der Internet-Protokolle als oberste Schicht ihrer Netzwerksysteme benutzen. Die Sammlung der Internet-Protokolle beinhaltet ein paketbasiertes Wide Area Netzwerk, das in der Lage ist, Netzwerke mit unterschiedlichen Protokollen und sehr verschiedenen Verbindungscharakteristika miteinander zu verbinden.

- **World Wide Web** - Das World Wide Web ist ein verteiltes Hypermedia-System, das auf einigen der Dienste, die das Internet bereitstellt, aufsetzt. Die wichtigsten sind der Benennungsdienst, den das *Domain Name System* (DNS) bereitstellt, und der recht verlässliche, verbindungsorientierte Übertragungsdienst des *Tansmission Control Protocol* (TCP).

2.1 Die Internet Umgebung

Das wichtigste, wenn man ins Internet will, ist zunächst eine Verbindung herzustellen. Es gibt prinzipiell zwei Arten, dafür aber eine grosse Anzahl technischer Lösungen. Wenn ein Computer ausschliesslich als Client fungiert verwendet man eine Einwählverbindung. Bei der Einwählverbindung benötigt man die Verbindung nur dann, wenn ein Server kontaktiert werden soll. Die einfachste und gängigste Lösung für diesen Ansatz ist die Modemverbindung. Wenn die Modemverbindung hergestellt ist, gibt es zwei verbreitete Methoden Daten auszutauschen. Die ältere Variante ist das *Serial Line Protocol* (SLIP) und die neuere Methode das *Point to Point Protocol* (PPP). Wenn man nun aber einen Server ans Netz schliessen will, braucht man eine permanente Verbindung, da man ja nicht weiss, wann der Client auf den Server zugreifen will. Diesem Zweck dienen die Standleitungen, welche bedeutend billiger sind als eine Telefonverbindung. Wenn man von einer Verbindung spricht, meint man, im Zusammenhang mit dem Internet, keine Verbindung von einem Computer zu einem anderen. Vielmehr spricht man von einem Netz von Netzen. Darum auch der Begriff Internet.
Um eine Verbindung zu erhalten, bedarf es aber auch noch einer Adresse. Adressen im Internet sind sogenannte *IP-Adressen* (Internet Protocol - Adressen). Eine IP-Adresse ist eine 32-Bit-Tahl, mit der man einen Computer global eindeutig bezeichnet. Namen im Internet sind sogenannte DNS-Namen. Das DNS bildet die Namen auf die IP-Adressen ab. (Telefonbuch) DNS-Namen sind in sogenannten Domains organisiert, die eine hierarchische Anordnung aufweisen. Die Domains der obersten Ebene der Hierarchieebene heissen *Top-Level Domains* (TLD), und jede TLD ist entweder eine *Country-Code TLD* (ccTLD), dargestellt durch einen aus zwei Buchstaben bestehenden Ländercode, oder eine *Generic TLD* (gTLD). Domain-Namen werden von rechts nach links notiert, so dass der oberste Level ganz rechts steht.
Um eine Web-Seite von einem Server zu laden, muss der Browser zunächst die IP-Adresse des Servers herausfinden. Dazu filtert er den DNS-Namen aus der URL und sendet eine Abfrage mit diesem DNS-Namen an den DNS Server. Als Antwort bekommt der Browser die IP-Adresse des Web-Servers und kann mit dieser Adresse eine Verbindung zum Server aufbauen, um die Web-Seite zu laden.

2.2 Grundbegriffe

Das Ziel der Web-Entwicklung war, basierend auf der allgemeinsten Stufe der Verteilung ein globales Hypermedia-System zu erzeugen, in dem Informationsressourcen auf einem Computer irgendwo in der Welt liegen können, solange dieser Computer vernetzt ist und die Standards für die Zugriffe auf Informationsressourcen versteht. Unter Hypermedia versteht man ein Konzept, welches Dokumente als miteinander verwobene Informationsstücke modelliert (z.B. ein Buch). Es gibt zwei Möglichkeiten, Hypermedia zu realisieren:

- Das externe Dokument automatisch an der Position des Links einzufügen
- eine Art Zeiger zu erzeugen, welcher auf das externe Dokument zeigt

Die Verteilung der Informationsressourcen ist auch ein interessanter Aspekt. Man unterscheidet drei Stufen der Verteilung:

- Innerhalb einer Datei
- Innerhalb eines Dateisystems oder Computers
- Innerhalb eines Netzwerkes, wobei die letzte Stufe die allgemeinste Stufe darstellt.

Dabei werden die drei Schlüsselkomponenten verwendet:

- **URL - Wie man ein Dokument benennt...**

Man braucht eine Möglichkeit, wie man eine Information im Netz identifiziert und auch wieder aufgefunden werden kann. Die Lösung dafür bietet die *Uniform Resource Locator* (URL).

- **HTTP - Wie man ein Dokument bekommt...**

Nachdem man eine Information benannt hat, benötigt man einen Mechanismus, um diese Information auch herbeischaffen zu können. Das *Hypertext Transfer Protocol* (HTTP) ist das Übertragungsprotokoll des Web.

- **HTML - Das Dokument für den Hypertext...**

Wenn man sich die Information beschaffen hat, muss man wissen, wie man diese zu behandeln hat. Das bedeutet, es muss Konventionen über das Dokumentformat geben. Dieses Format muss Hypermedia sowie auch Multimedia unterstützen können. Das vom Web bereitgestellte Konzept hierfür ist die *Hypertext Markup Language* (HTML).

Diese drei Grundkonzepte haben sich im Laufe der Jahre immer weiter zu wesentlich komplexeren und leistungsfähigeren Versionen weiterentwickelt. Doch die Schlüsselkonzepte sind immer noch die gleichen geblieben.

Es gibt vier weitere zentrale Begriffe, die im Zusammenhang mit dem Web immer wieder auftauchen:

- **Benutzer** - Der Benutzer ist ein menschliches Wesen, das mit Hilfe eines bestimmten Programms mit dem Web interagiert.
- **Browser** - Ein Browser ist ein Programm, das verwendet wird, um auf Web-Server zuzugreifen und von dort heruntergeladene Dokumente darzustellen. Die bekanntesten Browser sind der Netscape Navigator und der Microsoft Internet Explorer.
- **Client** - Ein Client ist im Web etwas allgemeineres als ein Browser. Während jeder Browser ein Client ist, ist nicht jeder Client ein Browser. Allgemein, ein Client ist ein Programm, welches auf Web Server zugreift (z.B. Suchmaschine).
- **Server** - Unter Server versteht man den Prozess, auf einem Rechner, der die Funktionalität bereitstellt, Anfragen von Clients zu beantworten. Ein Server muss nicht immer eine physikalische Maschine sein.

3. Datenübertragung

Da Netzwerke meistens höchst komplexe Objekte darstellen, braucht man eine angebrachte Organisation damit alles einwandfrei funktioniert. Diese Organisation wird mit Hilfe von sogenannten Protokollen aufgebaut. Zwei Dinge sind dabei speziell zu betrachten: Die Verpackung und die Hierarchie.
Wenn man Daten von einem Computer zu einem anderen senden will, braucht man eine Verpackung für diese Daten, da diese nichts als eine Reihe von digitalen Bits sind. Als erstes muss man den Transfer fehlerfrei machen. Es dürfen unterwegs keine Fehler auftreten. Hierfür werden dem Paket weitere Bits angehängt. Diese Bits enthalten redundante Informationen, welche allfällige Fehler entdecken helfen. Jetzt muss man noch sicher gehen können, dass das Paket auch an den richtigen Ort gelangt. Dies wird wiederum mit Codes gemacht, welche die oben genannten Adressen beschreiben und welche an das Paket „geheftet" werden. Das Paket wäre bereit. All diese Codesequenzen, welche man zugefügt hat, sind von einander unabhängig und isoliert. Sie werden Paketköpfe für Transfer Protokolle genannt. Diese Isolation gibt dem TCP/IP die benötigte Flexibilität.
Das ganze Netz ist hierarchisch gegliedert. Das ermöglicht einen effizienteren und schnelleren Transfer.

3.1 TCP/IP Protokolle

Dem TCP/IP Konzept liegt eine ganz spezielle Architektur zu Grunde. An oberster Stelle befindet sich die Applikation. Das sind die Protokolle der Anwendungen, welche die Informationen organisieren, welche das Netzwerk transportiert (z. B. Mail, News). Gleich unterhalb der Applikation steht der Transport. Transport Protokolle übernehmen von der Applikation die Informationen und bringen sie zum Ziel. Um dies zu garantieren, existiert das Internetwork. Es kennt die Zusammenhänge zwischen den Routern und den Hosts. Es kennt die Topologie des Netzes, welches durch den untersten Teil, der Netzwerk Technologie, aufgebaut ist. Unter der Netzwerk Technologie versteht man die physikalischen Mittel, welche die Vernetzung überhaupt ermöglichen (siehe Vortrag 2).
Ein Netzwerk kann aus vielen Subnetzwerken bestehen, welche untereinander durch Routers verbunden sind. Router übertragen die Daten durch das Netzwerk. Hosts empfangen oder senden Daten.

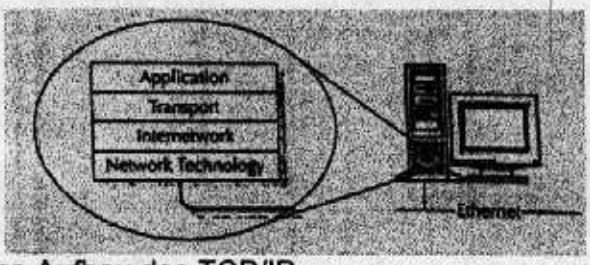


Abb.1 Der schichtweise Aufbau des TCP/IP

Es gibt zwei Arten, welche ein Protokoll unterstützt:

- **Connectionless Delivery** - Connectionless Delivery (Verbindungsloser Dienst) behandelt die Daten separat und unabhängig von den anderen. Vergleichbar ist das mit einer Postkarte, welche man, ohne mit dem Empfänger zu kommunizieren in den Briefkasten wirft und dann auf eine Antwort wartet.

- **Connection-Oriented Delivery** - Bei der Connection-Oriented Delivery (Verbindungsorientierter Dienst) nimmt das Protokoll zuerst Kontakt mit dem Empfänger der Daten auf und wartet auf eine Bestätigung, bevor das Paket oder weitere Pakete abgeschickt werden. Das ist sehr gut mit einem Fax vergleichbar.

Diese beiden Arten werden auch miteinander auf verschiedenen Ebenen kombiniert.

Beispiel:

Die Kommunikation beginnt sofort, nachdem die PC-Applikation das Transport Layer Protocol aufruft, eine Verbindung zur Workstation herzustellen. (1) Die Applikation benützt TCP als Transportprotokoll und TCP liefert einen connection-oriented service. Die erste Aktion ist deshalb die des TCP's, welches seine eigene Meldung zur Workstation sendet. Die Meldung wird vom TCP zum Internetwork layer protocol (IP) weitergegeben. (2) Das IP ist connectionless und gibt darum die Meldung gleich z.B. ans ATM weiter. (3) Wie TCP ist aber ATM ein connection - oriented service und hält darum das IP-Paket zurück und sendet zuerst seine eigene Meldung an die Workstation. (4) Diese Meldung erreicht die Workstation über die Verbindung und die Workstation sendet ihre Antwort zurück. (5) Nun fügt das ATM Netzwerk seine Informationen an die Meldung des IP's und sendet diese dann an die Workstation. (6) Die Meldung erreicht über die selben Stufen in umgekehrter Reihenfolge die Workstation(7),(8), welche nun weiss, dass der PC eine Transport Verbindung möchte. Sie bewilligt die Verbindung und sendet ihre eigene Meldung zurück. (9)Das IP fügt wieder seine Informationen hinzu und gibt die Meldung weiter ans ATM.(10) Da die Verbindung schon existiert gelangt die Meldung sofort zum PC (11-13).

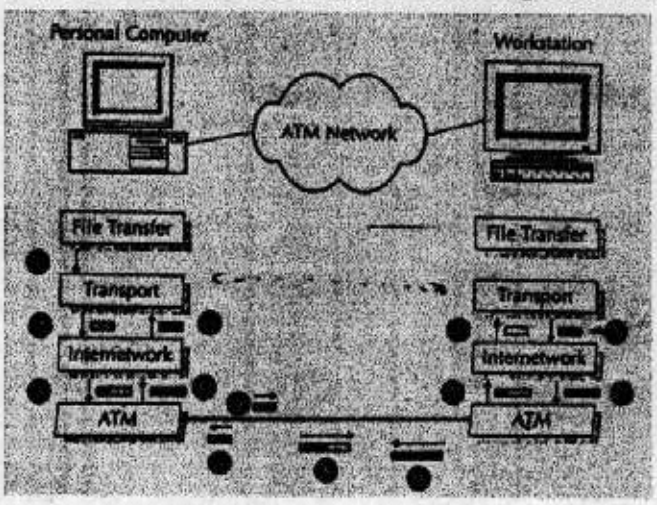


Abb. 2 Kombination von Connectionless und Connection Oriented Service.

4. Zusammenfassung und Schlussfolgerungen

Das Internet, ist ein Netz von Netzen, welches verschiedensten Applikationen ermöglicht, weltumspannende Datentransfers zu tätigen. Die am meist verbreitete Applikation ist sicher das World Wide Web. Wie man eine Verbindung ins Internet bekommt, hängt vom Benutzer ab. Die gängigste Variante ist sicher das Modem und

die Standleitung. Dabei verwendet das Internet Protokolle, welche einen zuverlässigen Datentransfer ermöglichen (TCP, IP). Man kann sich den Transfer als vielschichtiges Modell vorstellen. Die unterste Schicht bilden dabei die physikalischen Elemente, zuoberst steht die Applikation.
Das Internet wird mehr und mehr verflochtener, aufgrund seiner enormen Vielfältigkeit und Komplexität. Damit wird die Robustheit des Netzes auch garantiert. In Zukunft plant man den Einsatz des Internets auf Spielkonsolen, sowie über die Fernsehtechnik. Auch wird die Erforschung der Optoelektronik noch einen sehr grossen Einfluss auf das Internet ausüben.

Bibliographische Angaben

(1) E. Wilde: *World Wide Web – Technische Grundlagen*; Springer Verlag, Berlin, Deutschland, 1999

(2) S. Thomas: *Ipng and the TCP/IP Protocols*; John Wiley & Sons, Inc., New York, U.S.A, 1996

Netzwerktechnologien für das Internet

Andreas Egli

Vortrag Nr.2 / 2. Mai 2000

Netzwerktechnologien für das Internet

Das Internet besteht nicht aus einem einzigen, sondern aus vielen, miteinander verbundenen Netzwerken. Für diese unterschiedlichen Netze gibt es verschiedene Netzwerktechnologien. Die an diese gestellten Anforderungen, um sie für das Internet verwenden zu können, sind sehr gering. Die verschiedenen Netze müssen lediglich in der Lage sein, Pakete zu transportieren. Es bestehen jedoch keine Anforderungen bezüglich Zuverlässigkeit, Durchsatz, Verzögerung oder Reihenfolgeerhaltung. Die Netzwerke bilden lediglich die physikalische Verbindung, d.h. die auftretenden Fehler bzw. Probleme müssen von den aufgesetzten Protokollen erkannt und behoben werden.
Trotz der geringen Anforderungen eignen sich bestimmte Netze mehr oder weniger für das Internet.

1. Lokale Netze (Local Area Network, LAN)

Lokale Netze verbinden verschiedene Endsysteme im lokalen Umfeld (innerhalb von Gebäuden, Universitäten oder Firmengeländen). Die meisten Technologien für lokale Netzwerke beruhen auf einem gemeinsam benutzten Übertragungsmedium (in Bus- oder Ringstrukturen).

1.1 Ethernet

Die verschiedenen Stationen sind beim Ethernet an ein gemeinsames Koaxialkabel angeschlossen (Abb. 1). Will eine Station senden, so muss diese vorher überprüfen, ob das Kabel nicht belegt ist. Sobald das Übertragungsmedium frei ist, kann die Station mit dem Senden beginnen, dabei muss diese jedoch das Medium weiter überprüfen, ob nicht jemand zum selben Zeitpunkt mit dem Senden begonnen hat, was dann zu einer Kollision führen würde. Bei einer Kollisionserkennung müssen alle Sender ihre Übertragung abbrechen und dürfen erst wieder nach einer zufälligen Zeitperiode einen erneuten Sendeversuch starten. Dieses Problem hat zur Folge, dass die Stationen nicht beliebig weit entfernt sein dürfen und nur Pakete mit einer Mindestgrösse übertragen können, da sonst die Zeit nicht reicht um die Kollision zu erkennen. Je höher die Bitrate, umso höher muss die minimale Paketlänge oder umso kleiner muss die maximale Distanz zwischen den Stationen gewählt werden.

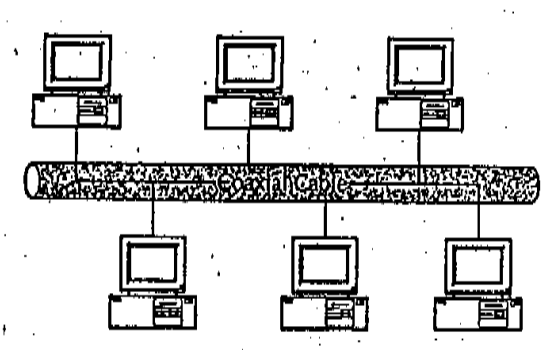


Abb. 1: Computer teilen sich ein gemeinsames Medium

Die gängigen Übertragungsraten des Ethernet sind 10Mbit/s und 100Mbit/s (*Fast Ethernet*), aber es gibt bereits solche mit 1Gbit/s (*Gigabit-Ethernet*).
Natürlich sind bei den heutigen Ethernets die Stationen nicht mehr direkt an einem Koaxialkabel, sondern via Twisted Pair-Kabel an einem HUB angeschlossen. Diese sind in der Lage, zwischen den einzelnen Stationen verschiedene Verbindungen zuschalten (*switched Ethernet*), so dass ein paralleles Benützen des Netzes möglich ist (Abb. 2).

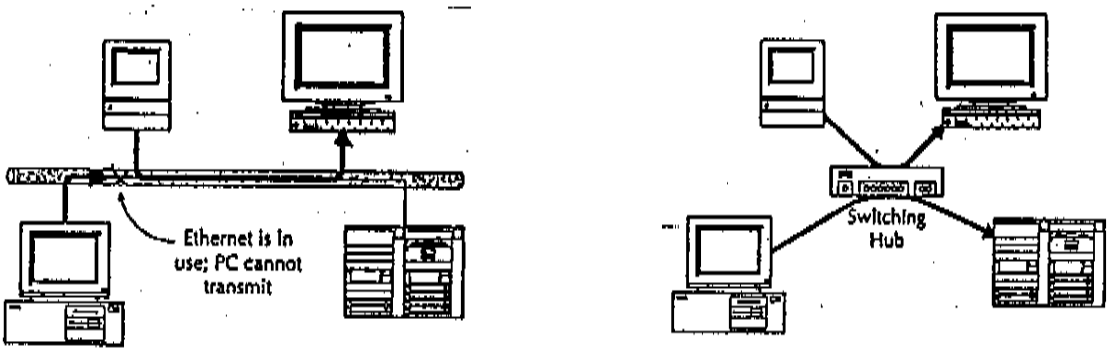


Abb. 2: normales Ethernet switched Ethernet

Obwohl das Ethernet die berühmteste und wohl auch weitverbreitetste Netzwerktechnologie für LAN ist, gibt es noch andere wichtige LAN-Technologien.

1.2 Token Ring

Bei der Token Ring-Technik wird das Senderecht mit Hilfe eines „Sendezeichens“ (*Token*) erteilt, d.h. eine sendewillige Station muss warten, bis diese das *Token*, welches von Station zu Station weitergereicht wird, bekommt. Sobald eine Station sendet und somit das *Token* vom Ring nimmt, kann keine andere mehr senden, bis das *Token* wieder auf dem Ring ist, dies ist spätestens nach einer maximalen Zeitperiode der Fall (Token Holding Time). So können im Gegensatz zum Ethernet keine Kollisionen entstehen und man kann eher abschätzen, zu welchem Zeitpunkt das Paket beim Empfänger ankommt. Vom physikalischem Aufbau her, ähnelt dieses Netz dem switched Ethernet, denn die Stationen werden auch via TP-Kabel an die Media Access Unit (MAU) angeschlossen (Abb. 3).

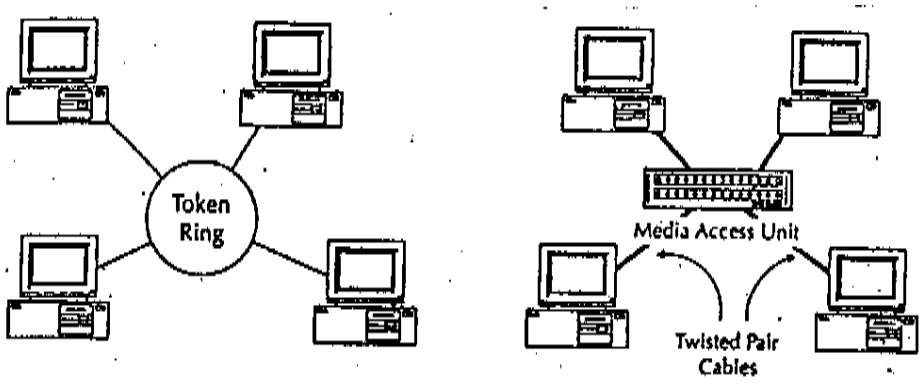


Abb. 3: Die verschiedenen Stationen sind über die MAU miteinander verbunden.

Die Weiterentwicklung für Glasfasernetze der Token Ring-Technik nennt sich *Fiber Distributed Data Interface* (FDDI), welche eine Bitrate von bis zu 100 Mbit/s bei einer

Reichweite von max. 200km haben können. Die wesentlichen Unterschiede bestehen darin, dass es FDDI erlaubt, mehrere Datenpakete von einer oder von verschiedenen Stationen zu übertragen und dass es grössere Sicherheit gewährleistet, da es aus zwei zueinander gegenläufigen Ringen besteht.

2. Weitverkehrsnetze (Wide Area Network, WAN)

Weitverkehrsnetze verbinden über grosse Distanzen verschiedene Netzwerke (LAN`s) oder Endsysteme miteinander. WAN`s sind meist verbindungsorientiert, d.h. vor dem Datenaustausch muss zuerst eine Verbindung hergestellt werden, wobei diese nach dem Datenaustausch wieder getrennt werden.

2.1 X.25

X.25-Netze sind sogenannte Paketnetzwerke (Statistical Time Division Multiplexer, STDM), welche für die schnelle Datenübertragung wenig geeignet sind. Die Bandbreiten bei X.25 liegen typischerweise im Bereich unter 200 Kbit/s, in Ausnahmefällen bei maximal 2 Mbit/s; zudem ergeben sich in solchen Netzen erhebliche Laufzeiten, die im wesentlichen durch die Speicherung und Verarbeitung in den Netzknoten zustande kommen. Das Protokoll X.25 (Abb. 4) bietet jedoch die Möglichkeit auf einer physikalischen Leitung mehrere Kanäle gleichzeitig zu führen. Dadurch gelingt die Vollvermaschung eines Netzwerkes mit nur einem Interface pro Standort. Auch lasten Paketnetze die Leitungen zwischen den Netzknoten effizient aus, wodurch keine Bandbreiten verloren gehen.

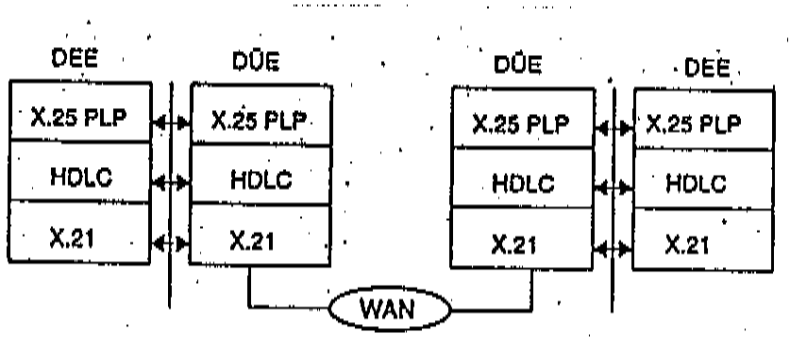


Abb. 4: X.25-ProtokollArcitektur

Legende:

X.21	physikalische Schnittstelle
HDLC	(High-Level Data Link Procedures) dient zur Fluss- und Fehlerkontrolle zwischen benachbarten Systemen (End- und Vermittlungseinrichtung)
X.25 PLP	(Packet Layer Protocol) dient zur Fluss- und Fehlerkontrolle zwischen zwei Endsystemen
DEE	Datenendeinrichtung
DÜE	Datenübertragungseinrichtung

Fazit: Obwohl X.25-Netze für die Datenkommunikation geschaffen worden sind, stossen sie bei Forderungen nach kurzen Laufzeiten und höheren Übertragungsraten jedoch schnell an ihre Grenzen.

2.2 Frame Relay

Frame Relay ist ein vergleichsweise junger Standard für die schnelle Datenübertragung, der eigentlich eine Weiterentwicklung von X.25 ist. Frame Relay wurde entwickelt, da die klassischen Technologien (z.B. X.25) nicht geeignet sind für die schnelle Datenkommunikation (mit typischen Bitraten von 2Mbit/s und mehr). Diese Übertragungsrate und somit kürzere Laufzeit erkaufte man sich durch den Verzicht auf eine gesicherte Übertragung, welche durch die besseren Telefonnetze gerechtfertigt wurde. Zu beachten ist, dass durch die nach wie vor paketorientierte Übertragung in einem Frame Relay-Netz Laufzeitschwankungen entstehen, die unter anderem von der Netzlast abhängig sind.
Anders als bei X.25 oder auch ATM (vgl . 2.4), definieren die Frame Relay-Standards keine physikalischen Schnittstellen. Es kann im Prinzip jedes Interface für die Bitübertragung eingesetzt werden. Da Frame Relay ein reines Softwareprotokoll ist, wird es oft auf den klassischen X.25-Produkten implementiert.

2.3 Integrated Services Digital Network (ISDN)

ISDN benützt die Telefonleitungen um schnelle, zuverlässige und flexible digitale Verbindungen herzustellen, welche das Telefonnetz attraktiver für die Datenübermittlung machen. Im Gegensatz zu X.25 und Frame Relay ist ISDN nicht paket-, sondern verbindungsorientiert, d.h. dass bei ISDN zwischen den Kommunikationsteilnehmer eine Verbindung mit fester Bitrate etabliert wird. Diese Verbindung wird auch B-Kanal genannt und entspricht einer Bitrate von 64 Kbit/s oder ein Vielfaches davon, abhängig von der Anzahl der B-Kanäle (max. 30 B- und 1 D-Kanal = 1.92 Mbit/s). Der Verbindungsaufbau und Abbau erfolgen über einen separaten Kanal, den D-Kanal.
Diese Technologie wird auch als N-ISDN (Narrowband ISDN) bezeichnet, welche zu unterscheiden ist von B-ISDN (Broadband ISDN). Das meist auf ATM (vgl. 2.4) basierende B-ISDN bringt Bitraten von 155 Mbit/s bis zu 622 Mbit/s.

2.4 Asynchronous Transfer Mode (ATM)

ATM hat sich zu einer wichtigen Netzwerktechnologie, vor allem im Weitverkehrsbereich, entwickelt. ATM Protokolle können problemlos mit zeitabhängigen Daten wie Video- und Audiosignalen umgehen. Auch hat sich ATM inzwischen im lokalen Bereich etablieren können, wo es aber selten eingesetzt wird, da es sich finanziell nicht lohnen würde. Ausgangspunkt für die Entwicklung von ATM waren die heutigen hohen Bitraten der Übertragungseinrichtungen und der Wunsch, nur diejenige Bandbreite den Anwendern zur Verfügung zu stellen, die sie auch benötigen.

Die wesentlichen Merkmale sind:

- Pakete (Cells) mit konstanter Länge (48 Nutzbytes, 5 Bytes Header)
- keine Übertragungssicherung der Nutzdaten
- für alle Nutzsignale (Audio, Video, Daten) und alle Betriebsarten (verbindungsorientiert, konstante Bitraten, variable Bitraten)

Die Daten werden am Netzzugang aufgeteilt und in entsprechend viele Zellen verwandelt. Wird die Bandbreite der Leitung nicht ausgeschöpft, so werden leere Zellen versandt, d.h. auf der Leitung werden ständig Zellen versandt. Je höher die Bandbreite am Netzzugang, desto mehr Zellen werden belegt und der Abstand zwischen den Zellen wird kleiner. Jede Zelle (Abb. 5) besteht aus 5 Byte Header (Kontrollinformationen) und 48 Byte Nutzdaten. Bei der Festlegung der Zellgrösse hat man einen Kompromiss zwischen grossen Zellen (64 Byte) für optimale Datenübertragung (Wunsch in USA) und kleinen Zellen (32 Byte) für Sprachübertragung (Wunsch in Europa) getroffen. Je grösser die Zellen, desto kleiner das Verhältnis zwischen Nutz- und Systemdaten (Overhead), aber desto höher die Übertragungsverzögerung (schlecht für isochrone Signale).

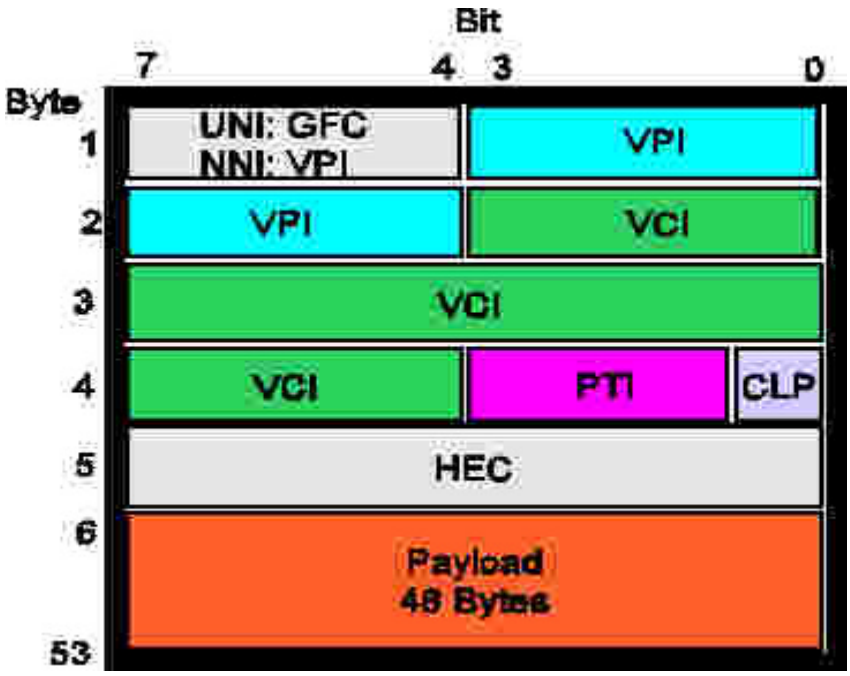


Abb. 5: ATM-Zelle

Legende:

GFC	Generic Flow Control	Datenflusssteuerung
VPI	Virtual Path Identifier	Adress-Kennzeichnung im Netz
VCI	Virtual Channel Identifier	Adress-Kennzeichnung im Netz (Nummerierung innerhalb VPI)
PTI	Payload Type Identifier	Kennzeichnung des Zellinhaltes
CLP	Cell Loss Priority	wenn 1 kann Zelle bei Bedarf entfernt werden
HEC	Header Error Checksum	Sicherung des Headers durch Prüfsumme
UNI	User Network Interface	Netzzugang User ins Netz
NNI	Network Node Interface	innerhalb des Netzes

3. Zusammenfassung

Die hier erläuterten Netzwerktechnologien:

- Ethernet
- Token Ring
- FDDI
- X.25
- Frame Relay
- ISDN
- ATM

sind mehr oder weniger oberflächlich betrachtet worden. Ich habe hier nicht versucht jede Technologie bis ins kleinste Detail zu erklären, sondern nur die wesentlichen Dinge darzulegen, so dass man im nachhinein die Vor- und Nachteile sowie die Unterschiede der verschiedenen Technologien kennen sollte.

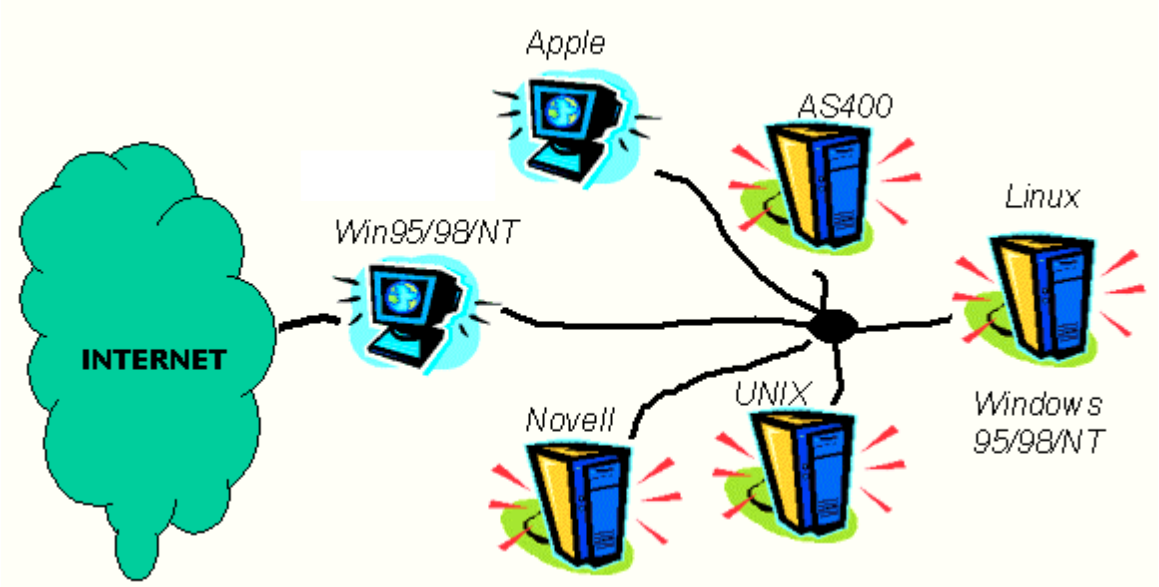
4. Referenzen

- IPng and the TCP/IP Protocols (S. Thomas)
- IPng – Neue Internet-Dienste und virtuelle Netze (T. Braun)
- Einführung in die Telekommunikationstechnik ATM (T. Jaeckel)

Eidgenössische Technische Hochschule Zürich

PPS Grundlagen des Internet

Vortrag 3



IP, Adressierung und Routing im Internet

Matthias Ammann

Einleitung

Dieser Beitrag behandelt folgende Punkte:

- Was ist ein Internet-Protokoll-Paket (IP-Paket)
- Wie funktioniert Adressierung im Internet
- Wie gelangen Daten von einem Rechner zu einem anderen.
- Was hat es mit IP-Adressen und Subnet-Masks auf sich.
- Wozu sind DHCP-, WINS- und DNS-Server gut.

Das Internet-Protokoll (IPv4)

Das Internet Protokoll umfasst im wesentlichen die folgenden Funktionen:

- Wegewahl,
- Lebenszeitkontrolle,
- Segmentieren und Reassemblieren,
- Fehlererkennung und
- Adressierung.

Eine der integralen Bestandteile von Protokollen der Vermittlungsschicht stellt die Wegewahl im Netz dar. Jedes System (End- oder Zwischensystem) untersucht die Zieladresse eines ankommenden Paketes daraufhin, ob sie identisch mit der eigenen Adresse ist oder ob sie eine Adresse eines direkt angeschlossenen Netzwerkes darstellt. Ist beides nicht der Fall, so wird das Paket an ein nachfolgendes Zwischensystem weitergeleitet. Handelt es sich um eine nicht bekannte Adresse, so wird eine Fehlernachricht über das ICMP (Internet Control Message Protocol) an den Sender des Pakets verschickt.

<table>
<tr><td>Bit
1 | 2 | 3 | 4</td><td>5 | 6 | 7 | 8</td><td>9 | 10 | 11 | 12</td><td>13 | 14 | 15 | 16</td><td>17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32</td></tr>
<tr><td>Version</td><td>Header-length</td><td>Precedence</td><td>Type of service</td><td>Total length</td></tr>
<tr><td colspan="3">Identification</td><td>Flags</td><td>Fragment offset</td></tr>
<tr><td>Time to live</td><td colspan="2">Protocol</td><td colspan="2">Header checksum</td></tr>
<tr><td colspan="5">Source address</td></tr>
<tr><td colspan="5">Destination address</td></tr>
<tr><td colspan="4">Options</td><td>Options padding</td></tr>
<tr><td colspan="5">Data</td></tr>
</table>

Abbildung 1- IP-Paket

Durch Fehlfunktionen im Netzwerk kann es vorkommen, dass Pakete nicht an die Zieladresse ausgeliefert werden, sondern im Netzwerk zirkulieren. Da diese Daten unnötig Netzwerkkapazität beanspruchen, müssen sie durch spezielle Maßnahmen irgendwann gelöscht werden. Dies ist die Aufgabe der Lebenszeitkontrolle. Im Feld TTL (time-to-live) wird ein Wert eingetragen, der die Lebenszeit des Pakets im Netz limitiert. Ursprünglich sollte die Maßeinheit dabei die im Netz verbrachten Sekunden darstellen. Praktisch durchgesetzt hat sich allerdings der sogenannte Hop-Count. Er legt fest, wie viele Knoten ein Paket im Netz durchlaufen darf, bevor sie gelöscht wird. In jedem durchlaufenen Knoten (End- oder Zwischensystem) wird die Lebenszeit um Eins erniedrigt. Erreicht sie den Wert Null und ist noch nicht am Ziel angekommen, so wird das Paket gelöscht und eine Fehlernachricht mittels ICMP an die sendende Station geschickt.
In einem Netzwerk besteht grundsätzlich eine Oberschranke für die Länge von IP-Paketen. Diese kann mehrere Ursachen haben: Hard- oder Softwarebeschränkungen, Beschränkungen aufgrund eines verwendeten Standards oder als eine Maßnahme zur Reduzierung der Fehlerquote bei der Übertragung. Aufgrund dieser Einschränkungen muss IP über eine Funktion verfügen, welche Pakete beim sendenden System segmentiert und im Endsystem wieder reassembliert. Im

Internet-Sprachgebrauch hat sich die Bezeichnung Fragmentieren anstelle von Segmentieren etabliert. Reassembliert wird lediglich in Endsystemen. Zwischensysteme reassemblieren zum einen wegen der dadurch steigenden Verweilzeit der Pakete im Netz nicht. Zum anderen können die Daten unterschiedlichen Wegen im Netz folgen und so eine Reassemblierung in den Zwischensystemen unmöglich machen.

Feld	Aufgabe
Version	Versionsnummer des Protokolls
Header length	Länge des IP-Kopfes in 32-Bit-Worten
precedence	Prioritätsinformationen, Routing-Protocol-Daten
Type of service	Angabe eines Diensttyps (Gesichtspunkte zum Weiterleitein etc.)
Total length	Gesamtlänge in Byte inklusive Kopf und Daten
Identification	Identifikationswert von Paketen
Flags	z.B. Anzeige. ob Fragmentierung zugelassen
Fragment offset	Position im reassemblierten Teil des gesamten Pakets
Time to live	Maximaler Hop-Count des Pakets
protocol	Kennung des Schicht-4-Protokolls z.8. 7 für TCP
Header checksum	Prüfsumme über den Kopf eines IP-Pakets
Source adress	Internet-Adresse des Herkunftssystems
Destination address	Internet-Adresse des Zielsystems
Options	Enthält Optionen wie Weginformationen etc.
Options Padding	Füllt die durch die Options angegebenen Daten auf ein Vielfaches von 32 auf
Daten	Beinhaltet die Nutzdaten

Tabelle 2 - Bedeutung der Felder eines IP-Packets

Die Fehlererkennung basiert auf einer Prüfsumme über den Kopf des Pakets, das heißt es können auch nur Fehler im Kopf erkannt werden. Der Datenteil ist auf dieser Schicht komplett ungesichert. Durch die jeweilige Anpassung der Lebenszeit eines Pakets muss die Prüfsumme in jedem System (Zwischen- oder Endsystem) neu berechnet werden, da sich der Kopf des Pakets ändert.

Adressierung: Hausnummer IP-Adresse

Die Grundlage für das Verständnis von Netzwerken bildet die Adressierung. Jeder vernetzte Rechner hat eine Adresse, ähnlich einer Hausnummer, über die er ansprechbar ist. Bei IP-Netzwerken ist das die IP-Adresse. Genaugenommen wird jeder Netzwerkkarte eine IP-Adresse zugeordnet, so dass ein Rechner mit mehreren Karten auch mehrere Adressen besitzen kann.

IP-Adressen sind eindeutig, 32 Bit breit und werden üblicherweise im Dezimalsystem notiert:

`172.89.34.12`

Jede IP-Adresse besteht aus zwei Teilen:

- Die **Netzwerk-ID** gibt das Netz an, in dem sich der PC befindet,
- und die **Rechner-ID** die Rechneradresse.

Um zwischen großen und kleinen Netzen zu unterscheiden, gibt es fünf verschiedene Klassen:

- Bei **Klasse-A-Netzen** gibt das erste Byte (die erste Zahl) das Netz an. A-Netze werden nur für große Firmen, meist amerikanische Organisationen, verwendet. Das erste Byte der Adresse stellt die Netzwerk-ID dar, wobei das erste Bit immer auf Null gesetzt ist. Mit den Verbleibenden 7 Bits lassen sich also 2^7 = 128 Netzwerke adressieren. *Zwei Kombinationen werden jedoch nie (auch in den Klasse-B und –C Netzwerken nicht) zur Adressierung verwendet: Alle Bits sind auf 0 oder alle auf 1 gesetzt.* Sie dienen speziellen Zwecken. Die Adresse 127 stellt außerdem noch einen Sonderfall dar: die Loopback-Adresse, die für rechnerinterne Testzwecke reserviert ist. Es verbleiben also 126 Klasse-A-Netzwerke. Weltweit kann es nicht mehr geben. Die verbleibenden drei Byte, bzw. drei Zahlen, geben den Host-Anteil an: etwa x.27.333.45. Das sind 2^{24} -2

=16777214 Kombinationen. In jedem Klasse-A-Netz können sich über 16 Millionen Rechner befinden.

- Bei **Klasse-B-Netzen** werden die ersten beiden Bytes für das Netzwerk verwendet, wobei die ersten beiden Bits immer 0 und 1 sind. Das bedeutet 16384 Netze mit je 65534 Hosts.
- Bei **Klasse-C-Netzen** geben die ersten 3 Bytes das Netz an. C-Netze erlauben 2097152 Netze mit je 254 Hosts.
- **Klasse-D-Netze** werden fürs Multicasting verwendet. Der Netzwerk-Anteil wird durch das erste Byte repräsentiert, wobei die ersten 3 Bits immer gesetzt sind. Multicast-Adressen beginnen mit 224.
- **Klasse-E-Netze** dienen experimentellen Zwecken. Bei ihnen sind die ersten 4 Bits immer gesetzt.

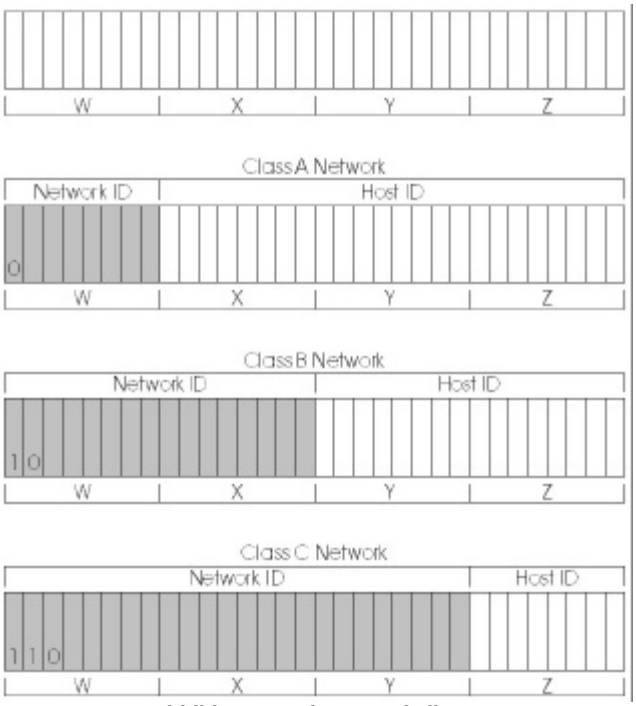


Abbildung 1 - Klassenaufteilung

Welcher Klasse eine IP angehört, erkennen Sie leicht am ersten Oktett. In der Tabelle "IP-Klassen" sind in der vorletzten und letzten Spalte jeweils der niedrigste und höchste Wert für das erste Oktett jeder Klasse angegeben.

Klasse	*Anzahl der möglichen Netze*	*Anzahl Hosts pro Netz*	*Anfang*	*Ende*
A	*126*	*16777214*	*1*	*126*
B	*16384*	*65534*	*128*	*191*
C	*20097152*	*254*	*192*	*223*
D	*--*	*--*	*224*	*224*
E	*--*	*--*	*240*	*--*

Tabelle 3 - IP-Klassen

Für Ihr Netz daheim können Sie folgende Adressen einsetzen:
•10.0.0.0 bis 10.255.255.255
•172.16.0.0 bis 172.16.255.255
•192.168.0.0 bis 192.168.255.255
Sie sind für den internen Gebrauch reserviert und werden offiziell nicht vergeben.

Subnet-Mask

Neben der IP-Adresse ist die Subnet-Mask der zweite wichtige Wert für die Adressierung. Mit der Subnet-Mask wird das Teilnetz bestimmt, in dem sich ein Host befindet. In der Subnet-Mask sind alle Bits auf 1 gesetzt, die den Netzwerkanteil festlegen. Ein Klasse-C-Netz verwendet zum Beispiel die ersten drei Bytes für den Netzwerkanteil, die Subnet-Mask lautet dann:

`255.255.255.0`

Wenn die Subnet-Mask mit der IP-Adresse logisch mit UND verknüpft wird, ist das Ergebnis der reine Netzwerkanteil. Der Host-Anteil entfällt, das in der Subnetmask alle Bits des Host-Anteils auf 0 gesetzt sind.

Adress-klasse	Binäre Subnet-Mask	Dezimale Subnet-Mask
A	11111111 00000000 00000000 00000000	255.0.0.0
B	11111111 11111111 00000000 00000000	255.255.0.0
C	11111111 11111111 11111111 00000000	255.255.255.0

Tabelle 4 - Subnet-Masks

Die Masks für die verschiedenen Klassen entnehmen Sie der Tabelle "Subnet-Masks". Der Host-Anteil der Subnet-Mask muss nicht immer gleich 0 sein, es gibt auch Masken wie 255.255.248.0 für ein B-Netz mit Subnetzen. Werte ungleich 255 bedeuten, dass das Netz in Teilnetze aufgeteilt ist, dieses Verfahren heisst Subnetting

Subnetting

Bei großen Netzen wie den Klasse-A oder B-Netzen tritt das Problem des Broadcasts auf. Broadcasts sind Rundsendungen im Netzwerk, die an alle Rechner gerichtet sind. Jeder Rechner muss eine solche Rundsendung untersuchen und darauf reagieren. Ein typischer Broadcast entsteht zum Beispiel, wenn ein Rechner versucht, einen anderen zu finden. Er sendet dazu ein Datenpaket an jeden Rechner im Netz. Alle Rechner müssen dann das Paket untersuchen, um festzustellen, ob es für sie bestimmt ist. Es wird also Rechenzeit verschwendet und das Netz belastet. Dieses Problem kann sehr groß werden - es entstehen Broadcast-Stürme, die das Netzwerk lahm legen können.
Die Lösung besteht in der Aufteilung des Netzes in Subnetze. Dabei werden Bits aus dem Host-Anteil für den Netzwerkanteil verwendet - es entsteht eine angepasste Subnet-Mask.

Anzahl	Bits binäres Oktett	Dezimal
1	Ungültig	Ungültig
2	11000000	192
3	11100000	224
4	11110000	240
5	11111000	248
6	11111100	252
7	11111110	254
8	11111111	255

Tabelle 5 - Angepasste Subnet-Masks

Sind zum Beispiel 12 Teilnetze erforderlich, so benötigen Sie dafür 4 Bits. Für ein Klasse-B-Netz sieht die Subnet-Mask damit so aus:

```
11111111 11111111 11110000 00000000
```

oder dezimal:

```
255.255.240.0
```

Mit 4 Bits lassen sich 16 Zustände darstellen, also auch 16 Teilnetze beschreiben. Wie bei dem Host- und dem Netzwerk-Anteil der IP-Adresse sind allerdings auch hier zwei Kombinationen abzuziehen (alle Bits 0 und alle Bits 1, so dass mit den 4 Bits 14 Teilnetze darstellbar sind).
In diesem Beispiel verbleiben 12 Bits für den Host-Anteil. Damit gibt es pro Teilnetz 2^{12} - 2 = 4094 Hosts. Auch hier sind die beiden Sonderfälle wieder abzuziehen.

Teilnetz-ID binär	Dezimal	Netzwerk - Maske
00000000	0	ungültig
00100000	32	x.y.32.0
01000000	64	x.y.64.0
01100000	96	x.y.96.0
10000000	128	x.y.128.0
10100000	160	x.y.160.0
11000000	192	x.y.192.0
11100000	224	ungültig

Tabelle 6 - Angepasste Subnets

Die IDs der Teilnetze lassen sich berechnen, indem die zusätzlichen Bits von 0 bis zum Maximalwert hochgezählt werden. Für eine Teilnetz-ID von zum Beispiel 3 Bit ergeben sich für eine Klasse-B-Netz die Teilnetz-IDs der Tabelle 6.

Routing

Routing ist der Schlüssel zum Verständnis eines Netzwerks. Wenn Sie verstehen wollen, wie das Internet funktioniert, müssen Sie verstehen, wie Routing funktioniert. Router verbinden Teilnetze miteinander und müssen daher mindestens über zwei Netzwerkkarten verfügen, für jedes Teilnetzwerk eine. Der Router leitet nun Datenpakete von einem Teilnetz in ein anderes. Anders ausgedrückt: Er leitet Datenpakete, die an der einen Netzwerkkarte ankommen, an die andere weiter. Dabei hat er verschiedene Manipulationsmöglichkeiten, die Administratoren für Proxies oder Firewalls ausnutzen (Stichwort Paketfilterung, Network address translation).
Woher weiß ein Router, für welches Teilnetz ein Datenpaket bestimmt ist? Ganz einfach: Er vergleicht die Subnet-Masks der Quelle mit der des Ziels. Beide Werte sind in einem Datenpaket enthalten. Nur wenn diese Werte unterschiedlich sind, schickt er die Pakete an das Teilnetz, das durch die Subnet-Mask des Ziels beschrieben ist.
Ist dem Router die Subnet-Mask des Ziels unbekannt, wählt er das Standard-Gateway: Es erhält automatisch alle Pakete, die der Router nicht zuordnen kann.
Das Standard-Gateway funktioniert ebenfalls wie ein Router, besitzt also auch ein Standard-Gateway. Auf diese Weise werden die Pakete von Gateway zu Gateway geleitet, bis sie das richtige Teilnetz erreichen.
Die Tabelle der Zielnetzwerke die der Router kennt, heißt Routing-Tabelle. Eine einfache Routing-Tabelle enthalt als wichtigsten Eintrag den Pfad zum Standard-Gateway.
Der Aufbau einer Routing-Tabelle ist einfach. Sie besteht aus Spalten:

- der Ziel-IP mit zugehöriger Subnet-Mask, dem zugeordneten Gateway,
- der IP der eigenen Netzwerkkarte, die das Paket erhalten hat,
- und der Metrik. Die Metrik beschreibt die Anzahl Rechner („Hops"), die bis zum Ziel zurückzulegen sind.

Die Routing-Tabelle lassen Sie mit
`route print` (Windows)
anzeigen.

DHCP

Die TCP/IP-Client-Konfiguration von Hand ist mühselig und fehlerträchtig. Ein einfacher Weg, die Einstellungen zu automatisieren, ist der DHCP-Dienst (**D**ynamic **H**ost **C**onfiguration **P**rotocol). Der DHCP-Server weist neuen Clients automatisch IP-Daten (seine Adresse, den Standard-Gateway, den DNS-Server) zu. Dieser Prozess verläuft in vier Schritten:

1. Der Client sendet einen Broadcast *„Wer ist hier DHCP-Server?"*
2. Alle vorhandenen DHCP-Server bieten ihre Dienste an
3. Der Client sucht sich den nächstgelegenen Server aus und fordert einen Datensatz an.
4. Der betroffene DHCP-Server schickt eine Empfangsbestätigung und zeigt damit insbesondere auch den anderen DHCP-Servern, dass er den Client erfolgreich registriert und bedient hat.

Die Vergabe einer IP-Adresse per DHCP wird Lease genannt, denn der Client leiht sich eine Adresse für einen bestimmten Zeitraum. Vor Ablauf dieser Frist kann er den weiteren Bedarf anmelden, und so die Lease verlängern. Reagiert er innerhalb dieser Frist nicht, wird seine Adresse für andere Hosts freigegeben.

Namensauflösung

Oft wird in Netzwerken der IP-Adresse ein lesbarer Name wie *www.pc-magazin.de* zugewiesen. Geben Sie einen solchen Namen zum Beispiel im Internet Explorer ein, verwendet der Browser spezielle Dienste, die die IP-Adresse des Namens ermitteln und zur Verfügung stellen.
In kleinen Netzen kommt eine statische Textdatei für die Zuordnung der Namen zu IP-Adressen zum Einsatz (Für Windows-NetBIOS-Namen heißt diese Datei *lmhosts,* für Internet-Domänen *hosts).* Sie ersparen sich damit den Einsatz spezieller Rechner, wie WINS oder DNS-Server. Diese Dateien können zentral auf einem Rechner liegen und von jedem Client via UNC-Namen abgeholt werden, zum Beispiel *\\meinrechner\lmhosts. So* müssen Sie nicht mehrere Dateien ändern, um einen neuen Host einzutragen.

Die Wartung erfolgt grundsätzlich von Hand, was bei größeren Netzen schnell zu umständlich und fehleranfällig wird.
Adressen von WINS- und DNS-Servern können auch per DHCP übergeben werden.

In Microsoft-Netzen werden zur Identifizierung von Netzwerkgeräten (Computer, Drucker etc.) NetBIOS-Namen verwendet. Für die Auflösung der NetBIOS-Namen in die zugeordneten IP-Adressen sind die NetBIOS-Namensserver (WINS-Server, Windows Internet Name Service) zuständig.
Aus Redundanzgründen sind meistens mehrere WINS-Server aktiv. Der große Vorteil der WINS-Server besteht in der Verringerung des Netzwerkverkehrs, da sich ein Client direkt an den WINS-Server wendet und nicht mehr per Broadcast erfragen muss: *„Heißt hier jemand pcmagazin?"* In gerouteten Netzwerken werden Broadcasts nicht weitergeleitet, so dass diese Methode sowieso entfällt.
Bei mehreren Teilnetzen wird entweder ein WINS-Server pro Teilnetz installiert, oder auf dem Verbindungsrechner (Gateway) läuft der sogenannte WINS-Proxy, der Namensanfragen an die WINS-Server weiterleitet. WINS-Server duplizieren ihre Datenbanken gegenseitig, so dass auf allen Servern immer die gleichen Daten vorhanden sind. Im WINS-Konfigurationsregister finden Sie das Texteingabefeld Bereichs-ID. Das ist ein Zusatz für NetBIOS-Namen, mit dem Sie Namen gruppenweise adressieren. Der NetBIOS-Name *pcmagazin* kann zum Beispiel die Bereichs-ID *.apps* erhalten, wodurch der neue NetBIOS-Name *pcmagazin.apps* lautet. Über die Adressierung **.apps* können nun gezielt alle PCs mit der ID *apps* angesprochen werden. Die Bereichs-ID fügt dem flachen NetBIOS-Namensraum eine zweite Ebene hinzu.
Für eine flexible Namensgebung reichen diese zwei Ebenen allerdings kaum aus. Ein besseres Namenskonzept stellt das DNS (Domain Name System) dar, das Namen mit drei oder mehr Ebenen wie *www.pc-magazin.de* einsetzt, sogenannte Fully Qualified Domain Names (FQDNs), die Sie aus dem Internet kennen.

DNS

Das **D**omain **N**ame **S**ystem benutzt einen hierarchischen Namensraum, der als ein auf dem Kopf stehender Baum dargestellt werden kann.
Der DNS-Baum besteht aus dem Root, den Top-Level Domains (TLDs) wie *de, com, org* etc. und den Domänen, manchmal auch Domänen der zweiten Hierarchieebene genannt. Root und TLDs werden von Organisationen wie InterNic oder ICANN verwaltet (In der Schweiz SWITCH). Diese besitzen große DNS-Server, die alle Anfragen an TLDs beantworten.
Die nächsttieferen Ebenen stellen die Domänen dar, die zum Beispiel von Firmen, aber auch von Privatleuten gemietet werden können. Die TLD *de* ergibt zum Beispiel mit der Domäne *pc-magazin* und dem Web-Server www den FQDN *www.pc-magazin.de.* Dieser FQDN besteht aus drei Ebenen. Mit Sub-Domains können weitere Ebenen entstehen, zum Beispiel *news.service.pcmagazin.de* für eine Domäne mit News-Servern. Sub-Domains werden in der Regel im eigenen DNS-System aufgelöst und nicht an die DNS-Server des Providers durchgereicht.
In jeder Zone gibt es einen Master-DNS-Server für die Beantwortung der Anfragen, den sogenannten primären DNS-Server. Daneben gibt es oft beliebig viele weitere sekundäre Server. DNS-Anfragen werden von Server zu Server nach oben im Baum weitergereicht. Kann der lokale DNS-Server zum Beispiel eine Anfrage nicht bearbeiten, reicht er sie weiter, etwa an den DNS-Server des Providers. Wird die Domäne dort nicht gefunden, gelangt die Abfrage zu einem der TLD-Server. Erst wenn diese keinen Rat mehr wissen, kann die Anfrage nicht bearbeitet und der Host nicht gefunden werden.

DNS-Domänen sind nicht mit den NT-spezifischen Domänen zu verwechseln, die zwar auch der Organisation dienen, aber einen Microsoft-spezifischen Mechanismus darstellen. DNS-Domänen sind dagegen unabhängig vom Betriebssystem.
Erst in Windows 2000 wird das Konzept der flachen Microsoft-Domänen zugunsten des Internet-DNS aufgegeben, so dass in Zukunft der Namensraum innerhalb von großen Organisationen viel feiner und komplexer unterteilt werden kann.

Zu den verschiedenen Methoden der NetBIOS-Namensauflösung (Broadcast, WINS- und DNS-Server, LMHOSTS und HOSTS) gesellt sich noch der NetBIOS-Namens-Cache auf dem Client hinzu. Dort wird eine erfolgreich durchgeführte Abfrage eine Zeit lang gespeichert. Bei einer erneuten Abfrage mit dem gleichen Namen sieht der Client erst in seinem Cache nach - das spart Zeit.

Schlusswort

Das Internet-Protokoll stellt ein relativ einfaches und übersichtliches System zur Kommunikation in Netzwerken zur Verfügung, das sehr starke Verbreitung gefunden hat. Mit zunehmendem Wachstum des Internets wird aber der jetzt schon knappe Adressraum immer enger. Mit Sicherheit werden aber in absehbarer Zeit auch Techniken zur Einsparung von IP-Adressen (Stichwort NAT, DHCP) nicht mehr genügen um die geforderte Anzahl Adressen bereitstellen zu können.
Deshalb steht schon einige Zeit eine verbesserte Version des Internet-Protocol in den Startlöchern – IP Version 6. Es soll den knappen Adressraum vergrössern und auch bessere Techniken für den Netzwerkverkehr bringen. Doch darüber berichtet der nächste Artikel...

Glossar

DHCP
Das Dynamic Host Configuration Protocol vergibt automatisch IP-Adressen an Hosts. Damit entfällt die umständliche Konfiguration von Hand, und es passieren weniger Fehler. DHCP übergibt auch andere Parameter wie zum Beispiel Adressen von WINS- oder DNS-Servern.

DNS
Das Domain Name System löst Host-Namen in IP-Adressen auf. Dazu erhält ein DNS-Server eine Anfrage von einem Host oder einem anderen DNS-Server in Form eines FQDN. Der DNS-Server liefert entweder selbst die zugehörige IP-Adresse zurück oder reicht die Anfrage an andere DNS-Server zur Bearbeitung weiter.

Domäne
Organisatorische Einheit für die Verwaltung von Hosts in größeren Netzwerken. Zu unterscheiden sind die DNS-Domänen des Internet von den Microsoft-Domänen.

FQDN
Der Fully Qualified Domain Name ist ein Name, der den Pfad zu einem Host beschreibt, zum Beispiel www.pcmagazin.de. www ist der Rechner, pc-magazin.de Domäne und de die Top Level Domain.
Host
Ein Host ist ein Rechner in einem Netzwerk. Dies kann sowohl ein Server als auch ein Client sein. Sobald Sie sich zum Beispiel mit Ihrem Internet-Provider verbunden haben, ist Ihr PC ein Host im Internet.

HOSTS
Statische Textdatei zur Auflösung von FQDNs in IP-Adressen. Wird nur in sehr kleinen Netzen verwendet, da die Verwaltung per Hand durchgeführt werden muss. Außerdem können sich Hosts nicht dynamisch registrieren. Ersetzt oder ergänzt DNS-Server.

LMHOSTS
Statische Textdatei zur Auflösung von NetBIOS-Namen in IP-Adressen. Kann in kleinen Netzen anstelle eines WINS-Servers eingesetzt werden. Wie HOSTS-Dateien muß auch die LMHOSTS-Datei von Hand gepflegt werden.

Namensauflösung
Die Namensauflösung beschreibt den Vorgang des Findens einer IP-Adresse anhand eines Names. In Windows-Netzen gibt es zwei Typen von Namen: NetBIOS- und Host-Namen.

NetBIOS
Einfache Programmierschnittstelle zur Kommunikation zwischen Anwendungen in einem Windows-Netzwerk. Wichtig sind die NetBIOS-Namen, wie zum Beispiel der Rechnername. NetBIOS-Namen dürfen maximal 15 Zeichen lang sein. Dies ist für größere und komplexe Netze oft zu wenig, weswegen man besser DNS-Namen verwendet. Diese sind auch besser strukturierbar.

Teilnetze (Subnetze, Segmente)
Wenn das Datenaufkommen in einem Netzwerk zu groß wird, wird es in Teilnetze aufgeteilt, die ihrerseits mit Verbindungsgeräten wie Routern verbunden sind.

Top Level Domains (TLD)
Oberste Hierarchie-Ebene (nach dem Root) des DNS. Wird an den Endungen der FQDNs erkannt, etwa .de, .com oder .org. TLD-Server werden von großen Organisationen gepflegt, die man für diese Dienste bezahlen muss (machen die Provider).

Verbindungsgeräte (Brücken, Router, Switches)
Verbindungsgeräte stellen Verbindungen zwischen Teilnetzen her. Es sind meistens ziemlich teure mehr oder weniger intelligente Geräte, die Datenpakete von einer Quelle über mehrere Teilnetze hinweg zum Ziel leiten.

WINS
Das Windows Internet Name System ist Microsofts System zur Namensauflösung. Funktioniert im Prinzip wie DNS, ist aber dynamisch: Geht ein neuer Host ans Netz, registriert er seinen Namen automatisch beim WINS-Server, sodass ein manueller Eingriff wie bei LMHOSTS- oder HOSTSDateien nicht nötig ist.

Quellenangaben:

[1] **T. Braun: Ipng – Neue Internet-Dienste und virtuelle Netze**; dpunkt Verlag, Heidelberg, Deutschland, 1999, Seiten 27-44.

[2] **S. Thomas: Ipng and the TCP/IP Protocols**; John Wiley & Sons, Inc., New York, U.S.A, 1996, Seiten 27-41.

[3] **M. Zitterbart, T. Braun: Hochleistungskommunikation 2**; Oldenbourg Verlag, München, Deutschland, 1996, Seiten 46-51.

[4] **B. Müller: Internet Basics**, PC-Magazin, Ausgabe März 2000, WEKA Computerzeitschriften Verlag GmbH, Poing, Deutschland, 2000, Seiten 228 - 232

IPng
Die nächste Generation des Internet Protokolls

Stefan Wildhaber

Vortrag Nr.4 / 16. Mai 2000

IPng - Die nächste Generation des Internet-Protokolls

1 Einführung

Die Popularität des Internet nimmt seit Jahren stark zu. Neben privaten Nutzern von Online-Angeboten, steigt auch die Zahl der Firmen, die das Internet kommerziell nutzen, d.h. ihre Dienste auf dem Netz anbieten indem sie sich und ihr Subnetz dem Internet anschliessen. Die Anzahl der Internet-Knoten nimmt aber auch wegen dem immer breiter werdenden Einsatzgebiet des Netzes zu. Mit dem Boom mobiler Endgeräte soll in Zukunft einfach „alles“ vernetzt werden. Es ist unschwer vorauszusehen, dass sich Engpässe im zum Verfügung stehenden Internet-Adressraum ergeben.
Aus diesem Grund hat sich schon 1992 die IEFT (Internet Engineering Task Force) entschieden, eine neue Version des Internet-Protokolls (IP) zu entwickeln. Mit den Erfahrungen die man mit TCP/IP gemacht hat soll mit IPv6 ein Nachfolger von IPv4 entworfen werden der die wachsenden Ansprüche befriedigen kann.
Die wichtigsten Neuerungen des neuen IPv6 sind in den Bereichen der Daten- und Adressierungsformate, der automatischen Systemkonfiguration und der Sicherheit zu finden. Trotzdem hat IPv6 eine einfache Architektur und ist „kompatibel“ mit schon bestehenden Netzwerkprotokollen.

2 Datenformate

2.1 Allgemeines Paketformat

Im Vergleich zu IPv4 wurden bei IPv6 gravierende Änderungen beim Paketformat vorgenommen (siehe Abb.1). Insgesamt fällt auf dass das minimale Paketformat von IPv6 weniger Felder enthält als das minimale IPv4-Paketformat. Verschiedene Felder wurden eliminiert (dunkle Felder), andere IPv4-Felder wanderten in verschiede Erweiterungs-Header (hellere Felder).
Durch die neue Struktur sollen die Verarbeitungskosten der Mehrheit der IP-Dateneineheiten, jene ohne optionale Informationen, möglichst gering gehalten werden.

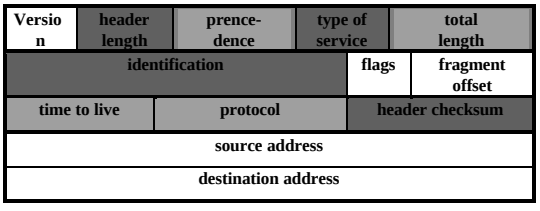


Abb.1/ „alter“IPv4-Header

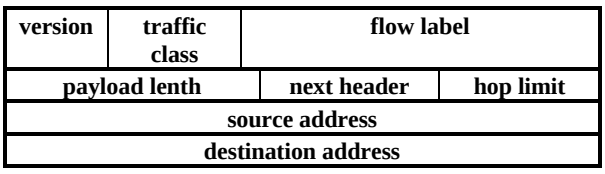


Abb.2/ IPv6-Header

2.2 Erweiterungs-Header

In IPv6 werden optionale Informationen in separaten Erweiterungs-Headern kodiert. Diese werden gerade anschliessend nach dem IPv6-Header plaziert (siehe Abb.3). Header-Erweiterungen werden bis auf wenige Ausnahmen nicht in den Routern entlang eines Pfades bearbeitet, sondern erst in den Endsystemen ausgewertet. Eine Ausnahme bildet die Hop-by-Hop-Option, die in den Zwischensystemen ausgewertet werden und unmittelbar dem IPv6 Kopf folgen muss. Die Erweiterungs-Header enthalten jeweils einen Verweis auf den Typ des nachfolgenden -Header

Header-Art	Grösse
IPv6-Header	40 Bytes
Hop-by-Hop-Option	variabel
Destination-Options-Header	variabel
Routing-Header	variabel
Fragment-Header	8 Bytes
Authentication-Header	variabel
Encapsulation-Security-Payload-Header	variabel
Destination-Options-Header	variabel
TCP-Header	20 Bytes
Anwendungsdaten	variabel

Abb.3/ Erweiterungs-Header

2.2.1 Der Routing-Header

Der Routing-Header wird verwendet, um einen oder mehrere Router auszuwählen, die auf dem Weg eines Pakets zum endgültigen Zielknoten passiert werden sollen. Mit z.B. einer Anycast-Adresse im Routing-Header kann der Weg eines Pakets zu seinem Ziel beeinflusst werden, um geringere Kosten oder eine bessere Dienstqualität zu erreichen.

2.2.2 Die Hop-by-Hop-Option

Die Hop-by-Hop-Optionen werden dazu verwendet, um optionale Informationen auszutauschen, die in jedem Knoten eines Pfades ausgewertet werden müssen. Jede Option wird durch einen Optionstyp und durch eine Längenangabe gekennzeichnet. Danach folgen die Optionsdaten und zuletzt der Fall, was geschehen soll, falls ein Knoten die Option nicht erkennt - je nachdem wird dann eine Fehlermeldung gesendet und/oder das Paket gelöscht. Ein Beispiel für eine Hop-by-Hop-Option ist die Router-Alert-Option, die verwendet wird, um einem Router anzuzeigen, dass das IP-Paket durch den Router besondere Beachtung erfahren soll. Eine andere Option, die Jumbo-Payload-Option ermöglicht die Erweiterung der IP-Paketlänge, die durch das 16 Bit lange Feld „packet length“ auf 65535 Bytes begrenzt ist.

2.3 Datenaufteilung und -Zusammensetzung

Jedes Subnetz hat eine maximale Grösse für die Dateneinheiten (IP-Pakete), die übertragen werden können. Diese „obere Paketgrösse“ ist abhängig von den physikalischen Eigenschaften des Subnetzes und wird Maximum Transmission Unit (MTU) genannt.
Der Fragment-Header ermöglicht es, grössere Nutzlastpakete zu senden, als dass die Pfad-MTU aufnimmt. Im Gegensatz zu IPv4 werden Datenpakete in IPv6 nur in Quellknoten segmentiert und nur im Zielknoten wieder reassembliert, d.h. Router führen keine Aufteilungs oder Zusammensetzungsfunktionen durch. Es sei denn, der Datenfluss kommt an eine Stelle, an der die MTU kleiner als das Minimum von ca.1300 Oktetts ist. In diesem Fall müssen auf der Link-Ebene Segmentier- und Reassemblierfunktionen angeboten werden.
Im Fragment-Header wird unter anderem die relative Position eines Fragments innerhalb des originalen Datenpakets beschrieben, und ob es das letzte Fragment darstellt oder nicht. Zudem wird im Fragment-Header der Sender eindeutig identifiziert, um beim Reassemblieren nur Fragmente des gleichen Senders berücksichtigen zu können.

2.4 Pfad-MTU-Erkennung

Um von Anfang an feststellen zu können, wie hoch die Pfad-MTU einer Verbindung ist, sendet der Sender das erste Paket an eine gegebene Zieladresse mit der Grösse der Link-MTU des ersten Links (siehe Abb.4). Falls auf dem Weg die MTU kleiner ist als die Paketgrösse, so dass das Paket nicht weitergeleitet werden kann, sendet der

betreffende Router die ICMP-Nachricht „message too big“ mit Angabe der Link-MTU an den Sender zurück. Der Sender passt infolge die Grösse des Datenpakets an der neuen MTU an.

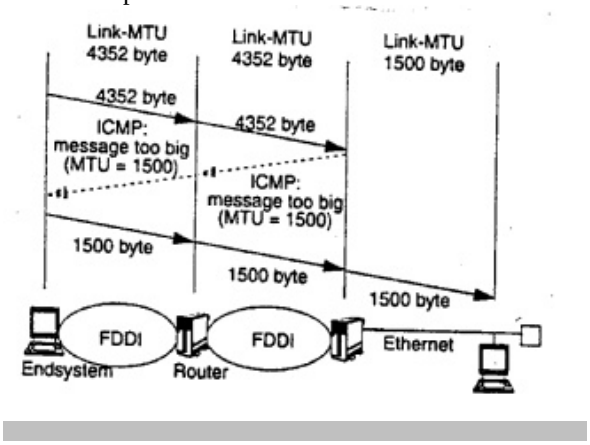


Abb.4/ Pfad-MTU-Erkennung

2.5 Flussmarken

Eines der interessantesten neuen Merkmale an IPv6 sind die sogenannten Flussmarken(flow labels). Mit der Flussmarke werden Pakete gekennzeichnet, die eine besondere Behandlung durch IPv6-Router benötigen. Beispielsweise können damit Pakete gekennzeichnet werden, die Realzeitdaten enthalten oder Verbindungen mit bestimmten Dienstgüteanforderungen verlangen. Flussmarken erlauben eine gleichzeitige Existenz von mehreren Flüssen zwischen einem Quell- und einem Zielknoten. Die Gültigkeitsdauer der Flussmarken wird durch Kontrollprotkolle bzw. durch die Angaben in den Optionen festgelegt. Es ist aber zu beachten, dass nicht sämtliche IP-Pakete mit verschiedenen Flussmarken versehen werden. Ansonsten könnte die Anzahl der zu unterstützenden Flüsse in den Routern zu gross werden, was bei zu kleinen Cache-Speichern in zu langen Suchzeiten nach den Flussmarken ergeben kann.

2.6 Internet Control Message Protocol (ICMP)

Mit dem ICMPv4 wurden hauptsächlich Fehler- und Echo-Nachrichten ausgetauscht. Das ICMP der neuen Generation (ICMPv6) wird auch für die Gruppenverwaltung eingesetzt, was zuvor von IGMP (Internet Group Management Protocol) übernommen wurde. Die ICMP-Nachrichten lassen sich in verschiedene Klassen einteilen:

- Fehlernachrichten
- Informationsnachrichten
- Nachbar Identifikationen (Neighbor Discovery)

Fehlernachrichten können die folgenden Angaben enthalten:

- Ziel unerreichbar
- Paket zu gross
- Zeitüberschreitung
- Parameterproblem

2.6 Automatische Systemkonfiguration

Die Untergruppe Neighbor Discovery (ND) der ICMPv6-Nachrichten hat vielseitige Verwendung und basiert wie andere IPv6-Kontrollmechanismen auf der IP-Multicast-Kommunikation. ND wird eingesetzt zur:

- Router-Erkennung:
  ND erkennt alle angeschlossenen Router, da diese periodisch Router-Advertisement-Pakete an alle umliegenden Hosts senden. Sie können aber auch explizit angefordert werden. Durch diese gegenseitige Kommunikation kann auch der nächste Knoten oder der Router für den kürzesten Weg bestimmt werden. Dank dem periodischen Absenden der Router-Advertisement-Pakete sind die Knoten immer up-to-date und erkennen stets, ob der Nachbarknoten noch erreichbar ist.

- Präfix-Erkennung:
  Die Router-Advertisement-Pakete enthalten Präfix-Listen für den Link, auf dem das Paket gesendet wurde. Sie befinden sich mit gemeinsamen Präfixen am gleichen Link. Knoten mit anderen Präfixen können also nur via Router erreicht werden.

– Parameter-Erkennung:
  Durch Analyse der Router-Advertisement-Pakete erkennen die Knoten link-spezifische Grössen wie die MTU oder den maximalen Hop-Limit-Wert. Die Parameter-Erkennung erleichtert die Konfiguration von IP-Knoten sehr stark, werden gar automatisch konfiguriert.
– Zustandslose automatische Konfiguration:
  Die IP-Adresse setzt sich bei der Benutzung dieses Mechanismus aus dem Link-spezifischen Präfix und der individuellen Interface-Kennung zusammen. Diese beiden Parameter können automatisch erkennt werden, sobald ein Knoten ein periodisch gesendetes Router-Advertisement-Paket empfängt.

3 Adressierung

Aufgrund der enorm wachsenden Nachfrage an IP-Adressen ist abzusehen, dass zwischen den Jahren 2005 und 2010 der derzeitig verfügbare IPv4-Adressraum den Bedarf nicht mehr abdecken kann. Eine Lösung besteht darin, einen grösseren Adressraum mit Adressierungshierarchien zu schaffen.
Die Adressen in IPv6 wurden um 86 Bits auf 128 Bits erweitert, was theoretisch pro Quadratmeter der Erdoberfläche ca. $6.65 \leftarrow 10^{23}$ Adressen erlaubt. Mit den führenden Bits, dem Formatpräfix wird grundsätzlich zwischen Anycast/Unicast und Multicast unterschieden.

3.1 Unicast-Adressen

Eine Unicast-Adresse identifiziert ein einziges Interface eines Routers oder Endsystems. Man spricht in diesem Fall von 1:1 Kommunikation (siehe Abb.5). Mit einem Präfix werden Unicast-Adressen in mehrere Typen und Hierarchiestufen unterteilt.
Es gibt verschieden Formen des neuen Adressformat(siehe Abb.6):

- globale/öffentliche Unicast-Adressen
- link lokale Adressen
- standortlokale Adressen

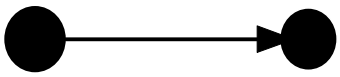


Abb.5/ 1:1-Kommunikation

	3 bit	13 bit	8 bit	24 bit	16 bit	64 bit
aggregierbar global	010	TLA ID	res.	NLA ID	SLA ID	Interface -ID
Standortl - lokal	1111111011 00...0				SLA ID	Interface -ID
Link- lokal	1111111011 00...0					Interface -ID
IPv4- kompatibel	0...0					IPv4 Adresse
IPv4- mapped	0...011111111 11111111					IPv4 Adresse
	96 bit					32 bit

Abb. 6/ IPv6 Unicast-Adressen

Die aggregierbaren globalen Unicast-Adressen, der allgemeinste Fall, werden ihrerseits wieder in einen globalen-, einen lokalen-Teil und einen Endsystem-Identifikator unterteilt. Der globale Teil wird benutzt, um Pakete über das globale Internet zur Lokation weiterzuleiten. Von dort aus führt der lokale Teil der Adresse die Pakete durch die Subnetz-Struktur innerhalb der Lokation, und schliesslich beschreibt die 64 Bit lange Interface-ID eindeutig den Typ des IP-Systems.
Das neue Adressformat ist aber auch kompatibel mit den schon bestehenden IPv4-Adressen, die durch einen Präfix in IPv6-Form gebracht werden.
Bei der Kommunikation innerhalb eines link-lokalen Netzes, also ohne fremde Router, werden entsprechend auch link-lokale Adressen verwendet, deren globaler Teil durch ein Präfix ersetzt wird.
Falls eine Organisation noch nicht an das Internet angeschlossen ist, kommen Standortlokale Adressen zum Einsatz. Deren Präfix kann bei einem späteren Anschluss ans Internet nur durch ein Provider-basiertes Präfix ersetzt werden, der Rest kann übernommen werden.

3.2 Anycast-Adressenensysteme identifizieren.

Eine Anycast-Adresse kennzeichnet eine Menge von Endgeräten, die typischerweise zu verschiedenen Zwischensystemen gehören (siehe Abb.7). Ein an eine Anycast-Adresse gesendetes Paket wird dabei genau an ein Interface dieser Menge ausgeliefert, in der Regel dem Nächstgelegenen gemäss Routing-Metrik. Eine typische Anwendung von Anycast-Adressen besteht im Wunsch, Pakete über ein Subnetz eines bestimmten Netzanbieters zu transportieren. Anycast-Adressen sollen hauptsächlich Zwischensysteme identifizieren.

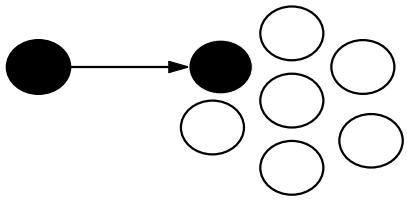


Abb.7/ Anycast-Adressen

3.3 Multicast-Adressen

Multicast-Adressen sind für die Gruppenkommunikation im Internet notwendig (siehe Abb.8). Ein an eine Multicast-Adresse gesendetes Paket wird von allen Gruppenmitgliedern empfangen. Eine Multicast-Adresse enthält ein Flag-Feld, ein Scope-Feld und die Gruppenkennung. Das Flag-Feld zeigt an, ob es sich um eine permanente Gruppe handelt, d.h. eine bekannte Gruppe mit registrierten Adressen, oder nicht. Das Scope-Feld kodiert die Reichweite, bzw. den Gültigkeitsbereich der Gruppe.
Bestimmte Multicast-Adressen sind vordefiniert. Dazu gehören die Adresse der Gruppe aller IPv6-Knoten, aller IPv6Router, aller „DHCP“-Server sowie die sogenannte Solicited-Nodes-Multicast-Adresse. Dieser link-lokalen Solicited-Nodes-Multicast-Adresse gehört automatisch jeder Knoten der verbundenen Gruppe an.
Multicast-Kommunikation bildet damit eine wichtige Basis für mehrere grundlegende Kontrollmechanismen im IPv6-Umfeld.

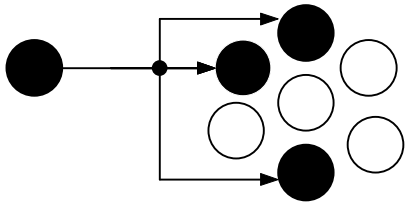


Abb.8/ Multicast-Adressen

3.4 Textform von IPv6-Adressen

Die 128 Bit der IPv6-Adresse werden in acht jeweils 16 Bit lange Integer-Werte aufgeteilt, die jeweils durch vier hexadezimale Zeichen dargestellt werden.
Zur Vereinfachung können Adressen mit vielen „0“-Zeichen abgekürzt werden.
Die bestehenden IPv4-Adressen werden kompatibel gemacht, indem sie vorne einfach mit Nullen aufgefüllt werden, denn ihre Adresslänge ist ja bekanntlich mit 36 Bit um 92 Bit kürzer als jene der IPv6-Adressen.

z.B.:	kann	1080:0000:0000:0000:0008:0800:200C:4711
	reduziert werden auf:	1080::8:800:200C:4711

4 Sicherheitsfunktionen

Sicherheitsfunktionen, die bei IPv4 noch optional sind, sind bei IPv6 Standard und müssen eine Basismenge von Funktionalitäten unterstützen. Authentifizierungen stellen sicher, dass die gesendeten Daten unverfälscht beim Empfänger ankommen während Verschlüsselungen der Pakete gewährleisten, dass nur der beabsichtigte Empfänger die Daten in eine lesbare Form umwandeln kann. Zur Durchführung von Sicherheitsfunktionen müssen sich Sender und Empfänger auf eine Menge von Parametern einigen. Dies sind z.B. Schlüssel für Authentifizierung,

Algorithmen zur Verschlüsselung, Lebenszeit der Schlüssel, oder Sicherheitsstufe der Kommunikation. Diese Parameter müssen von beiden Stellen vor der eigentlichen Datenübermittlung vereinbart werden.

4.1 Authentifizierung

Die Überprüfung der Echtheit der gesendeten Daten erfolgt durch Authentifizierungs-Header. Bei der Verwendung des MD5-Algorithmus wird vor dem Senden der Daten mit dem Empfänger ein Algorithmus und eine 128-Bit Kennung vereinbart und in die Authentifizierungsdaten eingetragen. Der Empfänger berechnet dann mit dem abgemachten Algorithmus die Daten. Beim Vergleich der Kennung erkennt der Empfänger ob die Übertragung ohne Verfälschung und oder Manipulationen Dritter verlief.

4.2 Verschlüsselung

Um einen vertraulichen, abhörsicheren Datenaustausch zu ermöglichen, wird der Verschlüsselungsalgorithmus DES-CBC (Data Encryption Standart - Cipher Block Chaining) verwendet. Der IPv6 Verschlüsselungskopf enthält in diesem Fall zusätzlich eine Sequenznummer und einen Initialisierungsvektor, der durch eine Zufallszahl erzeugt wird.
Die Verschlüsselung kann vor oder nach der Authentifizierung erfolgen. Die Daten können mit dem Tunnelmodus oder dem Transportmodus verschlüsselt übertragen werden. Der Unterschied ist, dass beim Tunnelmodus das gesamte IP-Paket verschlüsselt wird, und ein neuer unverschlüsselter IP-Header erzeugt wird, wobei im Transportmodus der IP-Header nicht verschlüsselt wird.

5 Schlusswort

Eine neue Generation des Internet-Protokolls wird fällig, und somit unausweichlich die Zukunft bestimmen. Mit den Erfahrungen des bisheriger IPv4 bringt man viele Ideen und Neuerungen in IPv6 ein. Es soll auch auf weite Sicht eine Basis zur Internet-Kommunikation bilden. Das neue Internet-Protokoll beeinhaltet Neuerungen im Bereich der Adressierung, der Datenformate und auch der Sicherheit.
Aber vielleicht hindern gerade diese Neuerungen, die z.T. Unsicherheitsfaktoren bergen, die Nutzer der Netzwerke auf IPv6 umzustellen, denn es sind erst wenige Firmen auf die neue Generation umgestiegen.
IPng ist bis heute fast auschliesllich erst auf Versuchsnetzen in Betrieb und lässt die Frage der vollständigen Kompatibilität mit dem bestehenden IPv4 noch offen.

Quellenangaben:

- T.Braun: *Ipng-Neue Intenet-Dienste und virtuelle Netze*
- S.Thomas:*Ipng and the TCP/IP Protocols*; John Wiley & Sons, Inc.,New York U.S.A, 1996

Elektronische Post im Internet

Seminar: Grundlagen der Internet Technologie
Vortrag Nr.5

Weber Michaël

1. Einleitung

Heutzutage kann sich kaum noch einer an die „schönen alten Zeiten“ erinnern, als alles mit rechten Dingen geschah. Wollte einer einen Brief an seinen Verwandten in Übersee schreiben, so musste er mühsam von Hand und wenn möglich in Schönschrift schreiben, denn Brief teuer frankieren und bei der nächstbesten Gelegenheit einwerfen. Mit ein bisschen Glück kam dieser Brief dann eine Woche später am Zielort an, ausser – naja, es kam halt auch vor, dass Briefe irgendwo auf dem Weg verlorengingen. Und das war gar nicht schön.
Zudem war eine richtige Kommunikation durch Briefverkehr kaum möglich, da, als der Brief angekommen war, sein Inhalt womöglich schon „veraltet“ war.
Aber es gibt einen Grund, wieso man heute über den herkömmlichen Briefverkehr meckern kann: Es gibt etwas Besseres!

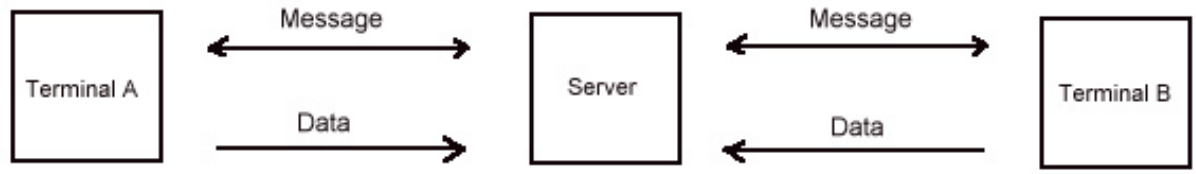
Es war eigentlich eine grosse Revolution in der Kommunikationstechnik, die aber kaum für all zu grosses Aufsehen sorgte. Eigentlich ging es blitzschnell. Jeder wollte es, jeder hatte es. Wer sich dagegen sträubte, hatte keine Chance. Wir reden hier von der elektronischen Post Im Internet, vom Email.

Leider kamen mit den überwiegenden Vorteilen auch Nachteile zum Vorschein. Zum Beispiel die Überschwemmung von meist absenderlosen Werbemails für einschlägige Seiten im Web (Spam), den Handel mit Email Adressen, um potentielle Kunden zu erreichen, und natürlich die viel einfachere, schnellere, globale und dadurch viel gefährlichere Verbreitung von Viren (Melissa, ILoveyou).

Aber nichtsdestotrotz ist heute ein Dasein ohne Email kaum noch vorstellbar.

2. Geschichtliches

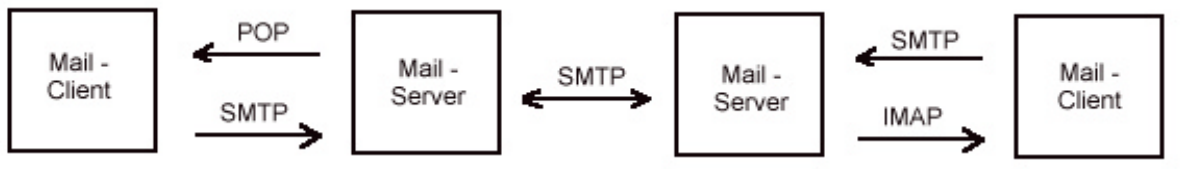
Alles begann eigentlich in den 80er Jahren in den grösseren Betrieben, vor allem in den wissenschaftlichen, mit eigenen Netzwerksystemen. Dort konnten die Mitarbeiter Nachrichten von einem Terminal zu einem anderen senden, die aber alle an einen gemeinsamen Server gebunden waren. Attachments mitzusenden war noch nicht notwendig, denn mit der entsprechenden Zugangsberechtigung durften Mitarbeiter auf gemeinsame Daten auf dem Server zugreifen.



Für den Anfang war dieses System ausreichend. Doch die Anforderungen an das System stiegen immer weiter an.

3. Email

Die Entwicklung ging also stetig weiter. Server verschiedener Unternehmungen fingen an, Daten untereinander auszutauschen und die Clients (Anwender) konnten sich von einem beliebigen Computer mit einem Modem in den Mail Server einloggen und Daten verschicken, beziehungsweise Daten empfangen. Dafür mussten aber entsprechende Protokolle entwickelt, angepasst und standardisiert werden.



Heute sind grundsätzlich drei Protokolle im Einsatz. Eines um Nachrichten zu senden (SMTP) und zwei um Nachrichten vom Mail-Server abzuholen (POP, IMAP).

4. SMTP (Simple Mail Transfer Protocol)

SMTP ist das Internet Standard-Protokoll zum Senden von Email-Nachrichten. Der Absender (Email-Programm) einer solchen Nachricht öffnet zum Empfänger der Nachricht eine Verbindung und sendet die Daten mit wenigen SMTP Befehlen.

Ein Beispiel: (RFC 821)

```
R: 220 smtp.ee.ethz.ch Simple Mail Transfer Service Ready
S: HELO smtp.datacomm.ch                                  // Anmeldung
R: 250 smtp.ee.ethz.ch

S: MAIL FROM:<Hansi@datacomm.ch>                          // Absender
R: 250 OK
S: RCPT TO:<webemich@ee.ethz.ch>                          // Empfänger 1
R: 250 OK
S: RCPT TO:<Misch@ee.ethz.ch>                             // Empfänger 2
R: 550 No such user here
S: DATA                                                   // Daten sende
R: 354 Start mail input; end with <CRLF>.<CRLF>
S: ... input .... etc.
S: ...etc. etc. etc.
S: <CRLF>.<CRLF>
R: 250 OK                                                 // Daten erhalten

S: QUIT                                                   // Abmelden
R: 221 smtp.ee.ethz.ch Service closing transmission channel
```

Der einfache SMTP-Standard erlaubt aber nur das Verwenden von ASCII-Zeichen, was nicht gerade benutzerfreundlich ist.
Denn zum Beispiel sind unsere Umlaute ä, ö, ü nicht in ASCII enthalten.

5. ESMTP (Extended Simple Mail Transfer Protocol)

Deswegen erweist sich das einfache SMTP schnell einmal als ungenügend. So wurde eine erweiterte Form des Protokolls geschrieben, das ESMTP. Dabei ist aber zu beachten, dass das ESMTP nur Sinn macht, wenn beide Seiten, der Sender wie auch der Empfänger, dieses Protokoll verwenden.
Hier sind einige Änderungen:

NACHRICHTEN MIT 8-BIT ZEICHENSÄTZEN

SMTP erkennt leider nur ASCII Zeichen, so dass es ziemlich unsicher sein kann, Nicht-ASCII Zeichen in einer Email zu verwenden. Mit ESMTP ist es auch möglich, nicht ASCII-Zeichen, wie zum Beispiel unsere Umlaute, zu verwenden.

ANKÜNDIGEN DER NACHRICHTENGRÖSSE

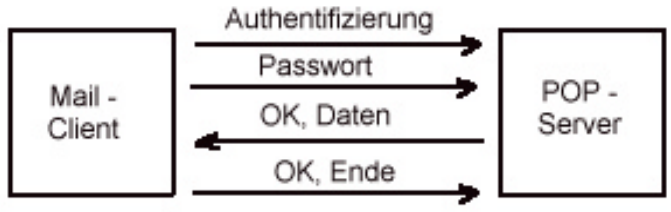
In letzter Zeit wurde es auch Mode, Mails mit ziemlich grossen Attachments zu senden. Da der Empfänger aber seine Leitung nicht stundenlang blockieren will, kann es sehr nützlich sein, im Voraus zu erfahren, wie gross die zu empfangende Nachricht ist. Mit ESMTP kennt der Empfänger im Voraus den Namen der Datei, deren Format und deren Grösse.

UMFANGREICHE UND BINÄRE NACHRICHTEN

Es kann sinnvoll sein, umfangreiche Nachrichten in mehrere Stücke zu zerlegen. So muss mit einem entsprechenden Mechanismus bei einer fehlerhafter Übertragung nur das defekte Teilstück neu übertragen werden.

6. POP (Post Office Protocol)

Alle empfangen Mails sind in einem Postfach unter dem User-Namen auf dem Mail-Server abgespeichert.
Der Client braucht nur noch eine Verbindung zum POP-Server herzustellen und mit einfachen POP-Befehlen zu prüfen, ob das Postfach nicht leer ist, und wenn ja, die neuen Nachrichten „abzuholen“.
Der POP-Server muss aber wissen, wer sein Postfach leeren will. Deswegen muss sich der Client zuerst mit seinem User-Namen und seinem Passwort identifizieren.

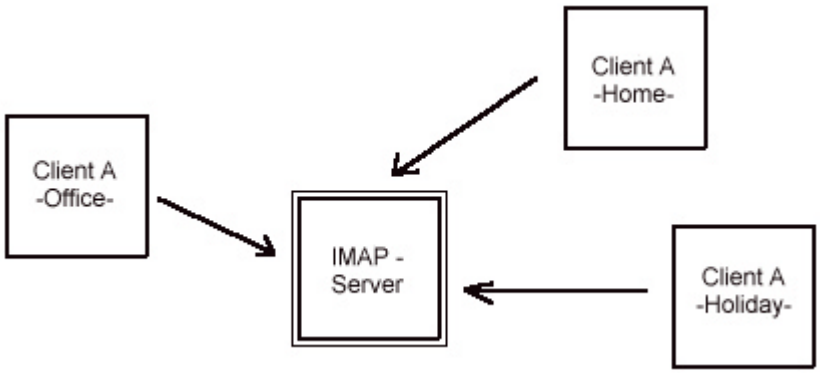


Der Client arbeitet mit dem POP eigentlich nur Off-Line, denn die einzigen Befehle beim Zugriff auf das Postfach dienen zum Abholen oder Löschen einer Nachricht.

7. IMAP (Internet Message Access Protocol)

Da POP quasi nur denn Offline-Betrieb unterstützt, wurde ein weiteres Protokoll entwickelt, welches eine Obermenge von POP ist. Dieses IMAP Protokoll wurde zuerst als *Interactive Mail Access Protocol* benannt, doch die heutige Bezeichnung lautet eben *Internet Message Access Protocol*.

Mit IMAP werden die Daten direkt auf dem Mail-Server verwaltet. Dadurch hat der Client immer Zugriff auf sein Mail-Konto. Ob er nun im Büro, Zuhause oder in Übersee ist, es spielt keine Rolle.



Hier sind die grössten technischen Vorteile von IMAP gegenüber POP:

UNTERSTÜTZUNG VERSCHIEDENER ORDNER
IMAP bietet die Möglichkeit, neben dem Ordner für erhaltene Nachrichten über das Netzwerk weitere Ordner anzulegen und zu bearbeiten (Auflisten, Erstellen, Kopieren, Löschen, Umbenennen, usw.).
Die Ordner auf dem IMAP-Server sind also genauso organisiert wie die lokalen Ordner.

ORDNERBEARBEITUNG ÜBER DAS NETZWERK
Nachrichten können von einem Ordner zu einem anderen Ordner verschoben und als gelesen oder ungelesen markiert werden.
Nachrichten können zudem so aktualisiert werden, dass andere Benutzer von dieser Aktualisierung in Kenntnis gesetzt werden.

OPTIMIERTE ONLINE-PERFORMANCE
Da Nachrichten sehr umfangreiche Teilstücke wie Bilder oder Videos enthalten können, erlaubt IMAP das einzelne Abholen solcher Teilstücke. Um dies zu ermöglichen unterstützt IMAP das Erkennen der Nachrichtenstruktur, ohne die gesamte Nachricht herunterladen zu müssen.

So klar die Vorteile von IMAP ersichtlich sind, so komplexer und schwieriger ist es, diese Protokoll zu implementieren.
Dennoch, die Zukunft wird wohl dem IMAP gehören.

Ein Beispiel zu IMAP: (RFC 2683)
(löschen eines Verzeichnisses „ETH“)

```
C: 008 CLOSE
S: 008 OK done
C: 009 DELETE ETH
S: 009 NO Delete failed; mailbox is not empty.
C: 010 SELECT ETH
S: * ... untagged SELECT responses
S: 010 OK done
C: 011 STORE 1:* +FLAGS.SILENT \DELETED
S: 011 OK done
C: 012 CLOSE
S: 012 OK done
C: 013 DELETE ETH
S: 013 OK done
```

8. Verschlüsselung und digitale Signaturen

Normale Nachrichten können auf dem Weg vom Absender zum Empfänger sehr leicht von Drittpersonen geöffnet und manipuliert werden. Deswegen sollten Nachrichten mit wichtigem Inhalt verschlüsselt oder digital signiert werden. Ganz klar, ohne den passenden Schlüssel ist es heutzutage unmöglich, verschlüsselte Nachrichten zu knacken. Die digitale Signatur weist ihren Urheber aus und verhindert somit Manipulationen.
Doch leider gibt es mehrere Standards solcher Verschlüsselungstechniken, was zu Kompatibilitätsprobleme führen kann.

Die wohl meist gebrauchten Standards sind S/MIME (Secure Multipurpose Internet Mail Extensions) und PGP (Pretty good privacy). S/MIME und PGP sind Spezifikationen für sichere Email-Kommunikation. Sie beschreiben Mail-Zusätze für kryptographische Dienste, die Authentifizierung, Verbindlichkeit, Integrität und Vertraulichkeit von Nachrichten sicherstellen können.
Diese Spezifikationen sollten im Allgemeinen ein sicheres Senden und Empfangen von verschlüsselten und digital signierten Emails zwischen unterschiedliche Mail-Clients ermöglichen. Doch leider lassen S/MIME und PGP gewisse Interpretationsmöglichkeiten offen, was zu unterschiedlich kompatiblen Softwareprodukte führen kann, die ein fehlerfreies zusammenarbeiten nicht gewährleisten können.

9. Schlussfolgerungen

Das Email ist zu einer sehr starken Kommunikationsmöglichkeit in unserer Gesellschaft geworden; Betriebe wie Privatpersonen schätzen die Schnelligkeit und Einfachheit des Emailsystems.

Doch mit jeder neuen Technik kommen auch neue Gefahren hinzu. Denn es wird immer Menschen geben, die Schwachstellen im System suchen um diese dann gnadenlos auszunutzen. Es geht hierbei um die Sicherheit des Email-Inhalts, aber auch um die gefährlichen Viren, die Milliardenschäden auf der ganzen Welt verursachen können. Wohlgemerkt, bei der letzten Virusattacke „ILoveYou“ war die Schwachstelle nicht das System selber, sondern die Neugierde der Mail-Clients.

Ganz allgemein sollte mehr für die Sicherheit vom Email getan werden. Vor allem für die Verschlüsselung wichtiger Nachrichten sollte endlich Standards entwickelt werden, die eindeutig kompatibel sind.

A. Quellenangaben

E.Wilde: World Wide Web - Technische Grundlagen ;
Springer Verlag, Berlin, Deutschland, 1999; Seiten 497 - 505

G.Spiegel: Gesicherter Umschlagplatz ;
c’t, Heft 26, 1999; Seiten 160 - 169

Internet: http://www.cis.ohio-state.edu/htbin/rfc/

HTTP

Hypertext Transfer Protokoll

Marco Somaini

Grundlagen der Internettechnologie

Vortrag 6 / 23. Mai 2000

Einführung

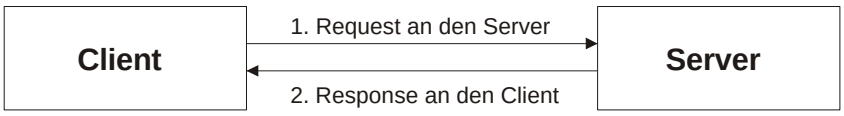
Immer wieder rufen wir Webs und andere Informationen im Internet auf, ohne genau zu wissen wieso wir gerade das auf den Bildschirm sehen, was wir sehen möchten. Denn all diese Daten sind nicht lokal gespeichert, sondern auf einem Server im Netz. Damit der Server den wir um eine Antwort bitte auch genau weiss, was wir sehen möchten, müssen wir ihm mitteilen, was wir von seiner Antwort erwarten. So wurde ein Verfahren entworfen, das es möglich macht auf einfache Weise die nötigen Informationen an den Server zu schicken. Somit erhalten wir als Client diese gewünschten Informationen. Dieses Verfahren für den Zugriff auf nicht lokal gespeicherte Informationen ist das *Hypertext Transfer Protocol (HTTP)*. Dieses Protokoll basiert auf einer Client/Server-Architektur, bei welcher der Client Informationen aus dem Web abrufen möchte und zu diesem Zweck mit einem Web-Server Kontakt aufnimmt.

Wie HTTP funktioniert

Um Informationen aus dem Internet abzurufen wird immer ein Client und ein Server benötigt. Auf dieser Grundlage basiert auch das HTTP-Protokoll. Will ein Client eine Information eines Servers abrufen, so sendet der Client dem Server eine Anfrage (Request). Auf diese Anfrage hin wird der Server mit einer Antwort (Response) dem Client die gewünschten Daten schicken. Das HTTP-Protokoll ist nun der Transportdienst, dar eine solche Kommunikation ermöglicht.

- *Client*
  Ein Client ist ein Programm, das Verbindungen aufbaut, um Requests zu senden. Normalerweise handelt es sich bei einem Client um einen WWW-Browser, aber er kann auch eine Suchmaschine oder eine andere Art von Programm darstellen.

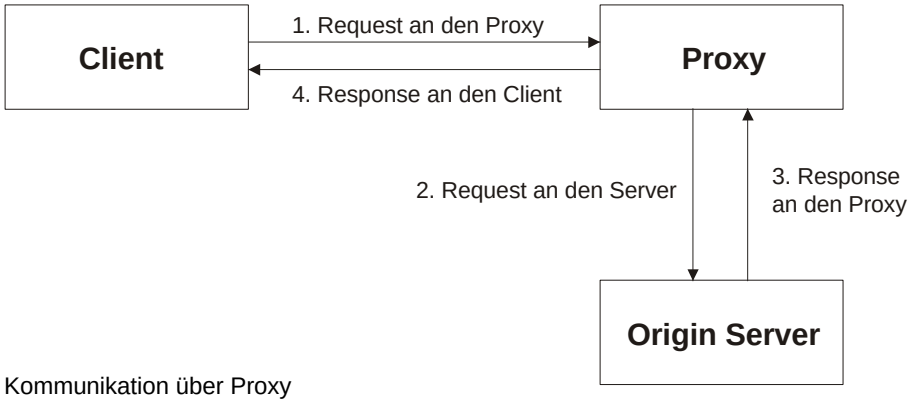
- *Server*
  Jedes Programm, das Verbindungen zulässt, um Requests durch die Rücksendung von Responses zu bedienen, wird Server genannt. Ein Server muss den Inhalt eines Request interpretieren und verstehen können.



Direkte Kommunikation

Ein solches Request/Response Protokoll kann direkt zwischen Client und Server stattfinden oder über einige Zwischenstationen geleitet werden. Die wichtigsten Zwischenstationen sind Proxies, Gateways und Tunnels.

- Proxy
  Ein Proxy übernimmt sowohl die Funktion eines Client als auch die Funktion eines Servers. Das heisst er nimmt vom Client ankommende Requests entgegen, verschickt aber auch selber Requests an einen Server. Der Proxy ist daher ein Programm, das den Origin Server unterstützt. So zum Beispiel kann der Proxy häufig besuchte Webs zwischenspeichern und sie direkt an einen Client weitersenden. So wird der Origin Server entlastet und lange wartezweiten verhindert. Wichtig dabei ist, dass sich der Client direkt an den Proxy wendet und seine Requests nicht an den Origin Server adressiert.



Kommunikation über Proxy

- *Gateway*
  Bei einem Gateway handelt es sich ebenfalls um ein Programm, welches anderen Servern als Zwischenstation dient und das aus diesem Grund einem Proxy ähnlich ist. Ein Client, der einen Request an ein Gateway sendet, weiss jedoch nicht, dass er nicht mit dem Origin Server kommuniziert, während er sich bei der Kommunikation mit einem Proxy dessen explizit bewusst ist.

- *Tunnel*
  Anders als bei einem Proxy oder Gateway handelt es sich bei einem Tunnel um ein Programm, dass bei der HTTP-Kommunikation als blinde Zwischenstation dient. Aus diesem Grunde wird die Nachricht weder interpretiert noch bearbeitet. Der Tunnel befördert eine ankommende Nachricht nur weiter an den gewünschten Server

Nachrichten

Der Aufbau einer HTTP-Nachricht ist recht einfach, da es sich bei der Nachricht nur um eine Textnachricht handelt, die im Internet leicht zu kontrollieren sind. Aus diesem Grunde sind auch die Unterschiede zwischen Request und Response nur sehr gering. Beide bestehen aus einer `start-line`, `message-header` -Feldern (die man auch als Header bezeichnen kann), einer Leerzeile sowie dem optionalen `message-body` (der, falls vorhanden, das sogenannte *Entity* der Nachricht enthält).

```
generic-message =
   start-line
   *message-header
   CRLF
   [ message-body ]

start-line =
   request-line | status-line
```

Bei der `start-line` einer Nachricht handelt es sich entweder um eine `request-line` (falls die HTTP-Nachricht einen Request darstellt), oder um eine `status-line` (falls es sich bei der HTTP-Nachricht um einen Response handelt). Nach der `start-line` enthalten HTTP-Nachrichten Header-Felder, die sich in vier unterschiedlichen Gruppen zusammenfassen lassen:

- **General Header**
  General Header finden sowohl bei Request- als auch bei Response-Nachrichten Anwendung, haben aber keine Auswirkung auf das übertragene Entity.

- **Entity Header**
  Falls das durch einen Request oder Response übertragene Entity der Beschreibung mit Hilfe einer Metainformation (z.b. Codierung, Länge) bedarf, kann dies durch die Verwendung sogenannter Entity Header in der zu versendenden Nachricht bewerkstelligt werden.

- **Request Header**
  Mit Hilfe eines Request Headers kann der Client Informationen über den Request und den Client selbst an den Server weitergeben. Request Header enthalten keine Informationen über den Nachrichtenkörper (d.h. das Entity der Nachricht).

- **Response Header**
  Response Header werden vom Server zur Übertragung von Informationen verwendet, die nicht in der status-line angegeben werden können. Sie enthalten keine Informationen über den Message Body (d.h. das Entity der Nachricht).

- **Start-line**
  In der `start-line` sind die wichtigsten Teile des Protokolls eingebettet. So zum Beispiel die Methoden der *Requests*. Sie definieren die vom Server auszuführende Aktionen.

  Einige wichtige Methoden in der `request-line`:

`GET`	Die Methode GET wird von einem Client eingesetzt, um die durch die request-URL angegebene Information (in Form eines Entity) abzurufen. Normalerweise handelt es sich dabei um ein Dokument oder um andere statische Informationen (beispielsweise um eine Grafik oder eine Audiodatei), die auf dem Server abgelegt sind.
`PUT`	Die Methode PUT kann von einem Client verwendet werden, um ein Entity unter einer bestimmten URL auf einem Server abzuspeichern. Der Request gibt die request-URL an, wobei es sich um die URL handelt, unter der das Entity gespeichert werden soll, und enthält darüber hinaus ein auf dem Server abzulegendes Entity
`OPTIONS`	Mit Hilfe dieser Methode kann ein Client Informationen über die möglichen Kommunikationsoptionen einholen. Der Client wird dadurch über die allgemeinen, nicht nur auf eine bestimmte Ressource zutreffenden Kommunikationsoptionen innerhalb der Request/Response-Kette zu einem Server informiert.
`TRACE`	Zu Diagnosezwecken kann es für einen Client interessant sein, wie eine an einen Origin Server gesendete Nachricht tatsächlich von diesem empfangen wird. Mit dieser Methode kann festgestellt werden wie eine Nachricht durch Zwischenstationen wie zum Beispiel Proxies verändert worden ist.

  Bei der *Response* wird in der `Status-Line` neben dem mitgesendeten Entity auch noch ein Code generiert, der dem Client mitteilt ob seine Methode erfolgreich war oder nicht.

  Einige wichtige Codedefinitionen:

Informational	Diese Klasse von Statuscodes zeigt an, dass der Request vom HTTP-Server erfolgreich empfangen wurde und jetzt von ihm bearbeitet wird.
Successful	Nachdem ein Server einen Request empfangen, verstanden und akzeptiert hat, sendet er einen Statuscode dieser Klasse zurück.
Redirection	Wenn ein Client einen Statuscode dieser Klasse empfängt, muss er zum Vervollständigen des Requests weitere Massnahmen ergreifen, die darin bestehen können, dass er einen Request an einen anderen in dem Response angegebenen Server schickt.
Client Error	Falls ein HTTP-Request nicht bearbeitet werden kann, weil der Client einen Fehler gemacht hat (beispielsweise einen Syntaxfehler oder das Senden einer Request Message ohne Berechtigung), antwortet der Server mit einem Statuscode dieser Klasse.

Content Negotiation

In vielen Fällen ist es möglich, dass eine angeforderte Quelle in verschiedenen Varianten auf einem Server vorliegt. Normalerweise verweist jede auf diese Quelle zeigende Referenz gleichzeitig auf alle Varianten. Mit Hilfe der Content Negotiation und dem HTTP-Protokoll kann nun in einem Request entschieden werden, welche der Varianten verwendet werden soll. Das solche Varianten durchaus Sinn machen, zeigen die folgenden Beispiele.

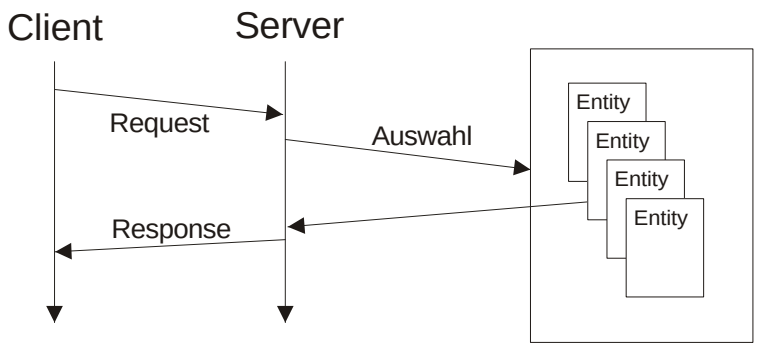
- **Sprachspezifische Varianten**
  Eine sprachspezifisches Material enthaltende Ressource (wie beispielsweise geschriebener Text in einem Textdokument bzw. einer Grafik oder Sprache in Audio- oder Videodateien) kann in verschiedenen Sprachen gespeichert werden, so kann jeder die Ressource in seiner gewünschten Sprache lesen

- **QualitätsspezifischeVarianten**
  Um auch Benutzern mit langsameren Netzwerkverbindungen einen schnellen Service zu bieten, kann mit Hilfe der qualitätsspezifischen Variante zum Beispiel die Auflösung von Graphiken und die Farbtiefe ausgewählt werden.

- **Codierungsspezifische Varianten**
  Da jeder Client nicht dieselben Möglichkeiten besitzen, ist es von Vorteil, eine passende Codierung zu erhalten. Unter Codierung versteht man zum Beispiel die Bildformate GIF oder JPEG. So kann jeder Client den passenden Code auswählen und somit eine gute Kompatibilität ermöglichen.

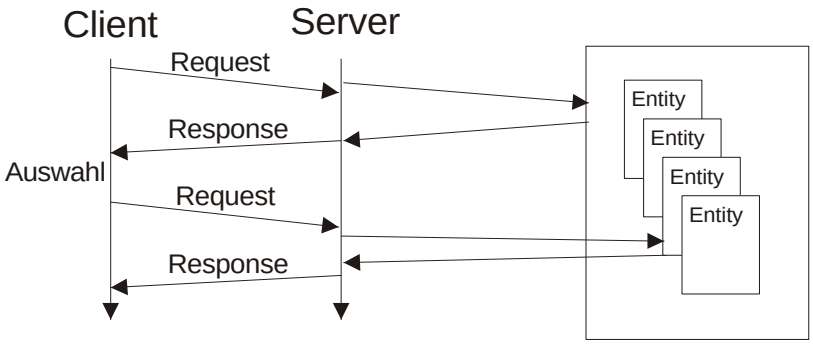
Die Arten der Content Negotiation

Server-Driven Negotiation
Bei dieser Negotiation kommuniziert der Client direkt mit dem Server. Dieser muss nun aus den aus dem HTTP-Protokoll hervorgehenden Fähigkeiten die optimale Version heraussuchen. Diese Methode ist von Vorteil, wenn sich die Auswahl nur schwer beschreiben lässt oder auf server-interne Kriterien beruht. Doch sie birgt auch Nachteile. Ein Server kann nie alle Fähigkeiten eines Clients wissen. Dies dauert sehr lange, ist ineffizient und kompliziert und kann nie zur einer ganz sicheren Auswahl führen.



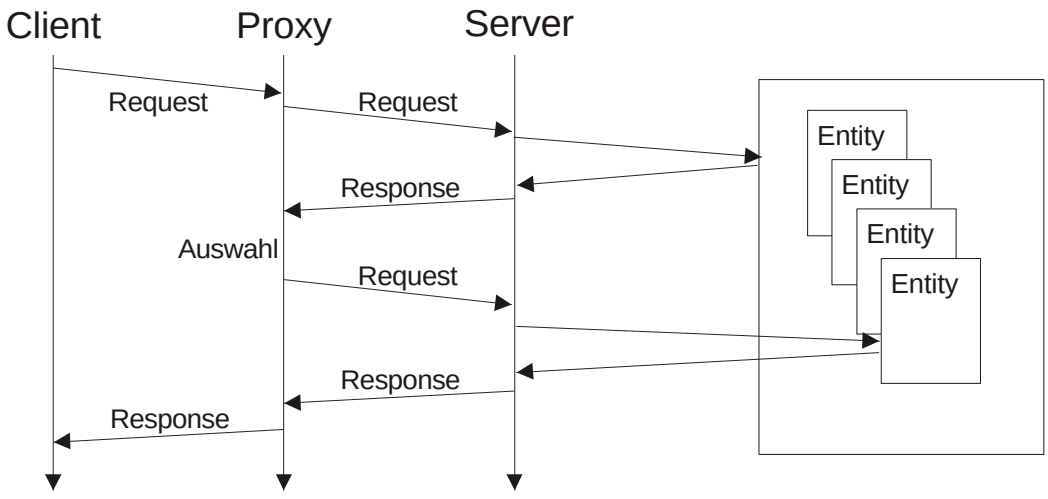
Server-Driven Negotiation

Agent-Driven Negotiation
Bei der Agent-Driven Content Negotiation wählt der Server nicht selbst die Version aus, sondern sendet eine Liste mit allen möglichen Varianten. Der Client wählt nun die gewünschte Variante aus, schickt seine Auswahl an den Server zurück, der mit der Response und damit mit der ausgewählten Variante antwortet. Da die Auswahl noch nicht automatisiert worden ist, muss der Benutzer selbst aus der Liste die richtige Variante auswählen. Dies schränkt die Möglichkeiten von Agent-Driven Negotiation stark ein.



Agent-Driven Negotiation

Transparent Negotiation
Transparent Content Negotiation stellt eine Kombination von Server-Driven und Agent-Driven Content Negotiation dar. Wie in einer Server-Driven Negotiation schickt der Client einen Request an den Proxy. Dieser übernimmt nun die ganze Aufgabe des Client, bis er die endgültigen Response erhalten hat. Somit wird die Auswahl vom Proxy getroffen und nicht mehr vom Benutzer selbst. Damit wird die Last zwischen den Agenten verteilt, was die Kommunikation erheblich beschleunigt.



Transparent Negotiation

Schlussbemerkung:

Im Internet werden täglich unzählige von Informationen abgerufen und nur selten wird eine falsche Web versendet. Dies ist hauptsächlich dem HTTP-Protokoll zu verdanken. Seine Request/Response Kommunikation liefert dem Server, Proxy oder andern Zwischenstation die benötigten Informationen über den Client, damit diese die richtige Wahl der Variante treffen können. Dies alles ist mit einer einfachen Textnachricht zu bewältigen, die bei jeder Anfrage verschickt wird. Das solche Nachrichten keine grosse Belastung für das Web darstellen, ist aufgrund seiner einfachen Struktur leicht einzusehen. Mit einigen wenigen Methoden lässt sich die Arbeit im Netz und mit HTTP schon bewerkstelligen.
Um dem Benutzer das WWW noch angenehmer zu gestalten wurde Content Negotiation eingeführt. Dies führt zu einer Verbesserung des Auswahlverfahrens und steigert somit die Chance die idealste Response für den Client zu finden. Mit Hilfe von Proxies wird sogar die Auswahl aus einer Liste von möglichen Varianten automatisiert.

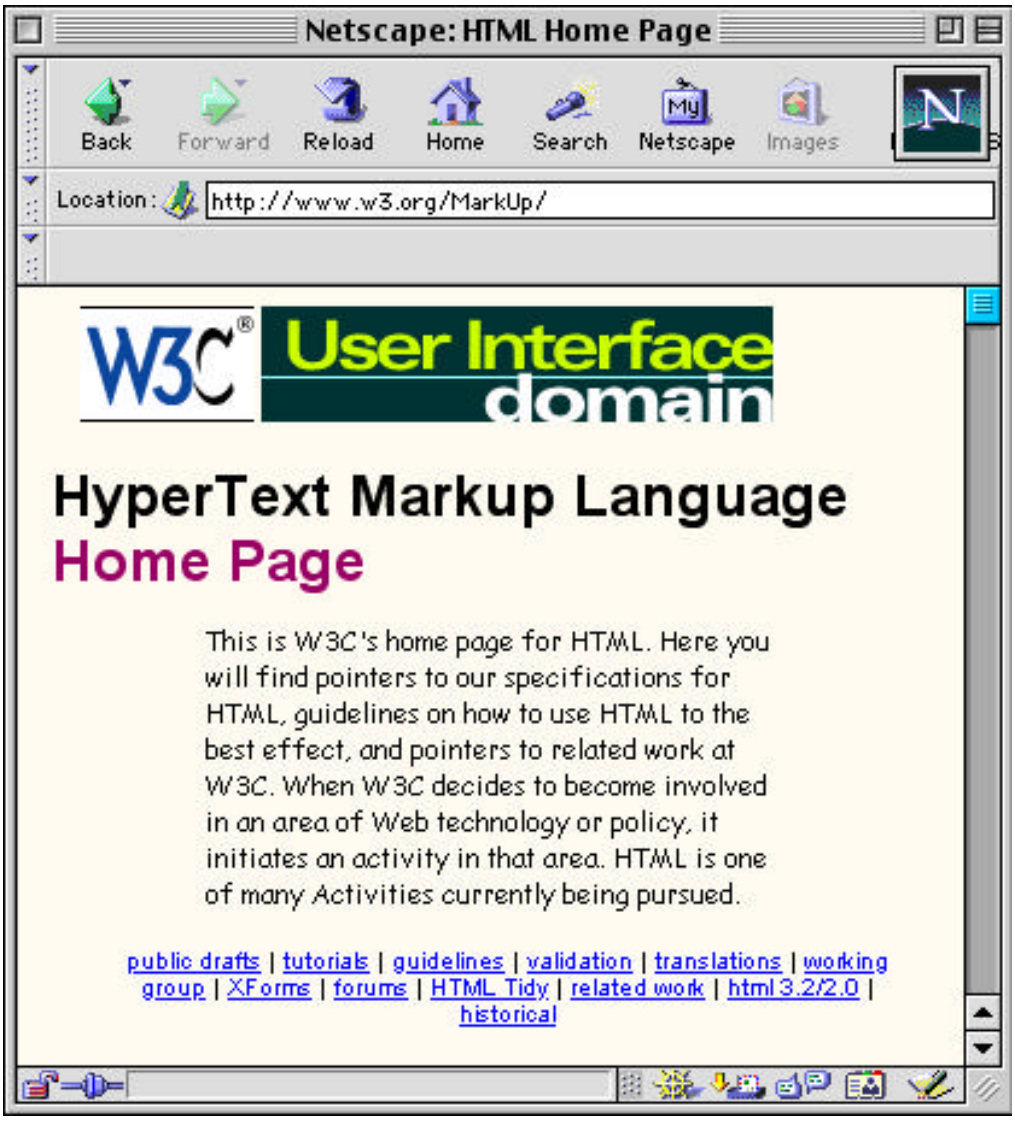
Quellenangaben:

[1] E. Wilde: World Wide Web – Technische Grundlagen; Springer Verlag, Berlin, Deutschland, 1999

[2] Diverse Webseiten zum Thema HTTP und seine Geschichte

Die Beschreibungssprache HTML

(**H**yper**T**ext **M**arkup **L**anguage)



Vortrag Nr. 7 des PPS-Seminars „Grundlagen des Internet“

30. Mai 2000

Autor: Daniel Ott

1. Einführung

In den vorangegangenen Vorträgen haben wir bereits viele Informationen über den Aufbau und die Möglichkeiten des Internet bekommen. Bei manchen Internetbenutzern wird früher oder später der Wunsch auftauchen, sich selbst im Internet präsentieren zu können. Dies kann mit der Beschreibungssprache HTML realisiert werden. HTML heisst soviel wie: **H**yper**T**ext**M**arkup**L**anguage.
HTML ist eine sogenannte Auszeichnungssprache (Markup Language). Sie hat die Aufgabe, die logischen Bestandteile eines Dokuments zu beschreiben. Als Auszeichnungssprache enthält HTML daher Befehle zum Markieren typischer Elemente eines Dokuments, wie Überschriften, Textabsätze, Listen, Tabellen oder Grafikreferenzen.
Die frühen Entwurfsziele waren die Folgenden:

- **Leistungsfähigkeit**
  HTML soll eine grosse Anzahl möglicher Anwendungen unterstützen und daher so allgemein wie möglich gehalten werden.

- **Einfachheit**
  Es sollte so einfach anzuwenden sein, dass möglichst viele Autoren ermutigt werden, HTML einzusetzen und zu verstehen, nicht nur Informatiker.

- **Zugänglichkeit und Plattformunabhängigkeit**
  HTML sollte sich in erster Linie auf die Inhalte einer Webseite und weniger mit deren Darstellung befassen. Dadurch hat ein grosses Publikum Zugang zu Informationen über HTML-Seiten und die Plattformunabhängigkeit wird gewährleistet, da die visuelle Darstellung der HTML-Seite Sache des Browsers ist.

Zur Einfachheit ist noch hinzuzufügen, dass man für die Erstellung von HTML-Seiten auf nichts weiter als auf einen Texteditor und einen Browser angewiesen ist, obwohl natürlich die grafisch orientieren Editoren die Arbeit meist stark vereinfachen und deshalb schon aus zeitlichen Gründen der manuellen Programmierung vorgezogen werden. Leider wirkt sich dies bei einigen Editoren sehr negativ auf den HTML-Text aus, der dann möglicherweise nicht mehr der HTML-Norm entspricht und demzufolge nicht auf allen Browsern korrekt angezeigt wird.
Der HTML-Standard wird vom www-Konsortium W3C festgelegt, deren Homepage für HTML auf der Titelseite zu sehen ist.

Zitat zur universellen Einsetzbarkeit aus „Selfhtml“:
„HTML ist als Auszeichnungssprache zum Erstellen von WWW-Seiten gedacht - eigentlich. HTML-Dateien funktionieren aber nicht nur im WWW. Es ist kein Problem, eine HTML-Datei lokal auf jedem Rechner mit einem WWW-Browser zu öffnen. HTML-Dateien sind deshalb auch ideal geeignet für lokale Dokumentationen, für CD-ROM-Oberflächen, für Readme-Dateien usw. Mit HTML und seinen unmittelbaren Ergänzungssprachen CSS und JavaScript, die ebenfalls lokal funktionieren, können Sie auch anspruchsvolle Projekte realisieren, die nicht für den Einsatz im WWW gedacht sind. Egal ob Sie Ihr Tagebuch fürs nächste Jahrtausend fit machen möchten, ob Sie bei der nächsten Version Ihrer Software eine HTML-basierte Online-Hilfe beigeben wollen, oder ob Sie eine informative CD produzieren wollen - HTML ist längst das verbreitetste Dateiformat der Welt. Ihre HTML-Dateien laufen auf jedem Rechner, auf dem ein WWW-Browser installiert ist - und ein Rechner, auf dem kein WWW-Browser verfügbar ist, darf mittlerweile bei aller Rücksicht als ein "veralteter Rechner" bezeichnet werden.“

2. Historischer Rückblick

2.1 Die Anfänge von HTML

Der erste Projektvorschlag für HTML dazu entstand 1989 im Kernforschungszentrum CERN in der Schweiz, mit dem Ziel, wissenschaftliche Erkenntnisse so allzeit aktuell präsentieren zu können. Ende 1990 war dann bereits ein Prototyp mit zeilenorientiertem und grafischen Browser für verschiedene Plattformen verfügbar. Die Funktionen dieser Version waren noch recht einfach, es gab damals Elemente für Textüberschiften (<H1...6>), geordnete und ungeordnete Listen sowie als wichtigsten Bestandteil jeder HTML-Seite, das Element <A> zum Angeben von Hypertext-Links. Das sind die anklickbaren Texte oder Bilder, die den Benutzer zu einer weiteren Seite bringt. Darin besteht die Grundidee des WorldWideWeb: Das Bewegen zwischen räumlich weit entfernten Rechnern wird bei modernen grafischen WWW-Browsern auf einen Mausklick reduziert.

2.2 Version 2.0

Bereits hier tauchten die ersten Probleme wegen Nichteinhaltens der bestehenden Regeln auf: Es wurden zwei neue Browser namens Arena und Mosaic mit neuen Funktionen entwickelt, die sich natürlich voneinander unterschieden und deshalb inkompatibel zueinander waren. Man bemühte sich darauf, die Vorteile beider Produkte in der HTML Version 2.0 zusammenzufassen. Kurz nach deren Freigabe wurde die Firma Netscape gegründet, die Ihren Browser ebenfalls mit vielen neuen Elementen aufzuwerten suchte, sodass die neue HTML-Version bereits bei ihrer Freigabe veraltet war.

2.3 HTML 3.2

Mit dieser Version wurden einige Neuerungen eingeführt, wie z. B. Tabellen, Applets, Textfluss um Bilder, Sub- und Superscripte, jedoch noch keine Frames, was unter anderem einmal mehr dazu führte, dass die damals aktuelle Version nicht den Stand der Dinge darstellte.

2.4 Version 4.0

Die heute noch immer aktuelle Version trägt die Nummer 4.0. Deren wichtigste Neuerungen waren die im Internet Proposed Standard beschriebene Internationalisierung, die Unterstützung von Style-Sheets, die offizielle Anerkennung von Frames, ein verbessertes Tabellenmodell, Unterstützung der Einbindung von Multimediaobjekten und besser ausgestattete Formulare.
Einige Formatierungs-Konstrukte sollten seit dieser Version nicht mehr verwendet werden, da ihre Funktionen neu durch Style-Sheets realisiert werden können.
Um durch solche Massnahmen nicht plötzlich ältere, nicht mehr dem Standard entsprechende Seiten unbrauchbar zu machen, aber trotzdem möglichst die neuen Standards einzuführen, definiert HTML 4.0 drei Document Type Definitions (DTD):

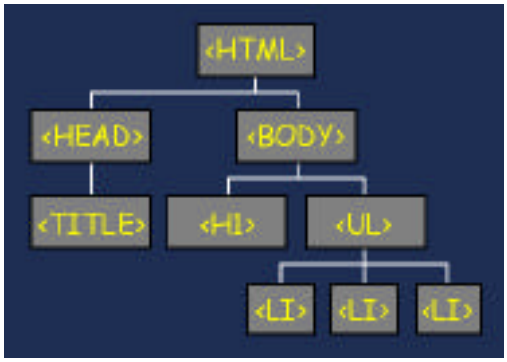
- **Traditional DTD**
  diese DTD dient ausschliesslich zum Interpretieren und nicht zur Erzeugung von HTML-Dokumenten (siehe auch nächsten Punkt). Sie enthält sowohl die neuen 4.0-Elemente, als auch viele veraltete, sodass auch ältere Seiten korrekt dargestellt werden. Einfach gesagt stellt sie die Regeln dar, an die sich ein Browser halten sollte.

- **Strict DTD**
  Hier sind nur noch die Konstrukte enthalten, die der Version 4.0 entsprechen. Zum Erstellen eigener Webpages sollte man sich also an diese DTD halten. Dies erfüllen die Editoren leider nicht immer, falls sie älter als HTML 4.0 sind, sowieso nicht.

- **Frameset DTD**
  Speziell für die nun unterstützten Frames wird eine dritte DTD eingeführt, die Framesets genauer spezifiziert, das sind die Seiten, die die Anordnungen und Grössen der einzelnen Frames festlegen. Mit Frames ist es möglich, mehrere HTML-Dokumente auf einer einzigen Seite anzuzeigen. Häufigstes Beispiel: Eine grosse Hauptseite und links davon eine schmals Auswahlmenü, mit dem man durch die Seite navigieren kann, ohne dass dieses Menü mit dem Wechsel der Hauptseite verschwindet.

3. Aufbau und Elemente einer HTML-Seite

Eine HTML-Seite lässt sich in zwei Teile gliedern: den HEAD- und den BODY-Teil Desweiteren kann man eine HTML-Seite als Baum auffassen, also ein hierarchisch gegliedertes Dokument. Dies lässt sich in der folgenden Grafik anschaulich darstellen:



3.1 HEAD

Im HEAD-Teil finden sich Informationen für den Browser wieder, die für die Darstellung der Seite wichtig sind oder einfach Informationen über die Seite und deren Inhalt bzw. Erstellung enthalten, was z. B. für Suchmaschinen von Interesse ist.

```
<HTML>
  <head>
    <meta name="generator" content="GoLive CyberStudio 3">
    <title>Home</title>
    <meta name="Content-Type" content="text/html; charset=iso-8859-1">
    <script language="JavaScript1.2" src="…"></script>
    <link rel="stylesheet" type="text/css" href="…">
  </head>

  </body>
    <--Hier steht der eigentliche Inhalte der Seite -->
  </body>
</html>
```

In diesem Beispiel wird als erstes darüber informiert, mit welcher Software diese Seite erstellt wurde. Auf der zweiten Zeile kommt das Titel-Tag zur Anwendung,

der darin enthaltenen Text erscheint beim Aufrufen der Seite als Fenstertitel.Dies ist der einzige vorgeschriebene Teil des Headers. Die zweite meta-Zeile informiert den Browser darüber, dass es sich um ein HTML-Dokument handelt und welchen Schriftsatz er dafür verwenden soll. Die Zeilen vier und fünf im Header definiert Art und Ort eines externen Java-Script-Files, sowie eines Style-Sheets. Mit Style-Sheets werden verschiedene Layoutparameter festgelegt, wie Schriftart u. ä. Java-Scripts sind Programmscrips, mit Hilfe dere sich gewise Aktionen ausführen lassen, wie z. B. die automatische Ausgabe des aktuellen Datums.

3.2 BODY

Für Interessenten möchte ich an dieser Stelle auf komplette HTML-Referenzen verwiesen, wie sie z. B. in Selfhtml oderim HTML-Buch von Günter Born zu finden sind (siehe Quellenverzeichznis). Hier kann ich, um den Rahmen der Zusammenfassung nicht zu sprengen, nur die wichtigsten Befehle, auch Tags genannt, behandeln.
Zur Illustration ist hier eine Beispielseite samt Quelltext abgedruckt, so dass man die Anwendung der Tags in der Praxis sehen kann

Darstellung der Seite in Netscape

Der dazugehörige Quelltext:

```
 1. <html>

 2. <head>
 3.   <meta name="generator" content="GoLive CyberStudio 3">
 4.   <title>Beispielseite</title>
 5. </head>

 6. <body bgcolor="white" alink="#ff00ff" link="blue" vlink="red" text="#000011">
 7.   <center>
 8.     <h1>Beispielseite</h1>
 9.   </center>
10.   div align="left">
11.     <p><font size="4">In diesem Beispiel ist</font></p>
12.     <ul>
13.       <li><font color="#ff3333">Eine Punkteliste</font>
14.       <li><font color="#0099ff">Eine Tabelle</font>
15.       <li><font color="#33cc00">ein Bild</font>
16.       <li>und Beispiele f&uuml;r Hyperlinks
17.     </ul>
18.     <p><font size="4">zu sehen.</font></p>
19.     <p><font size="4">Meine Homepage findest du <a
    href="http://www.stud.ee.ethz.ch">hier. </a></font></p>
20.     <p><font size="4">Sie ist auch <a href="Home.html">lokal</a>
    abrufbar.</font></p>
21.   </div>
22.   <center>
23.     <p><table border="4" cellpadding="0" cellspacing="2" width="216" height="61">
24.       <tr>
25.         <td>Dies</td>
26.         <td>ist</td>
27.         <td>eine</td>
28.       </tr>
29.       <tr>
30.         <td>Tabelle</td>
31.         <td>mit</td>
32.         <td>Inhalt</td>
33.       </tr>
34.       <tr>
35.         <td></td>
36.         <td></td>
37.         <td></td>
38.       </tr>
39.     </table></p>
40.   </center>
41.   <div align="left">
42.     <p><a href="mailto: dott@ee.ethz.ch"><img src="3D-E-mail.gif" width="55"
    height="60"></a></div>
43. </body>

44. </html>
```

3.2.1 Verweise

Das wichtigste Tag der HTML-Welt ist sicherlich das <a>-Tag, mit welchem Hyperlinks realisiert werden können. Ein solcher Link besteht mindestens aus

```
<a href="URL">Verweistext</a>
```

wobei „Verweistext“ der (meist unterstrichene) anklickbare Text darstellt und „URL“ als Platzhalter für die aufzurufende Seite steht (Bspe. Zeilen 19 & 20). Ein

weiterer häufig benutzter Verweis ist der Emailverweis

```
<a href="mailto:name@domain.xy">Verweistext</a>
```

der ein Fenster des Standardemailprogramms öffnet mit einer leeren Email an die angegebene Adresse (Bsp. Zeile 42).

3.2.2 Tabellen

Tabellen haben bei HTML mehrere Bedeutungen. Einerseits werden sie zum „normalen Verwendungszweck einer Tabelle verwendet, nämlich zur geordneten Darstellung von Daten. Andererseits ist es die einzige Möglichkeit in HTML (für Seitengestaltung ohne Style-Sheets), um Seitenelemente wie Textblöcke oder Bilder verschachtelt anzuordnen. Auf der obenstehenden Seite enthalten die Zeilen 23-39 den Code für eine Tabelle.

3.2.3 Bilder & Animierte GIF‘s

Da animierte GIF‘s ebenfalls eine Art Bild sind, werden sie gleich behandelt wie „normale“ Bilder. Ein Bild wird wie folgt plaziert:

```
<img src="Pfad_des_Bildes" height="…" width="…" alt="Welcome">
```

Es ist von Vorteil, die Grösse des Bildes anzugeben, was von Seiten HTML nicht zwingend vorgeschrieben ist. Wenn die Grösse explizit angegeben ist, kann die Seite fertig aufgebaut werden, selbst wenn das Bild nicht geladen werden kann. Ansonsten wartet der Browser mit der Fertigstellung der Page bis er das Bild gefunden hat, da er nicht weiss, wieviel Platz es auf der Seite einnehmen wird (Bsp. in Zeile 42).

3.2.4 Textoptionen

Für Text lässt sich die Grösse (1-6) und die Farbe bestimmen (im Bsp. an verschiedenen Stellen zu finden, in Zeilen 14-16 mit Farbzuweisung). Mit
 wird ein Zeilenumbruch erzwungen

```
<font size="4" color="#fffc11">Dieser Text erscheint gelb in Grösse 4</font>
```

Auch die Ausrichtung des Textes kann angegeben werden (Alles was zwischen Zeile 7 und Zeile 40 steht, erscheint zentriert). Dieses Tag stammt noch von HTML-Version 3.2 und sollte bei 4.0 konformen Seiten nicht mehr verwendet werden.

```
<center>Dieser Text erscheint zentriert</center>
```

Mit <hr> wird eine horizontale Linie erzeugt. Optional kann deren Dicke in Pixel angegeben werden.

3.2.5 Farb-Angaben für die ganze Seite im BODY-Tag

Im Bodytag können die Farben für Text, Link, benutzer Link und aktiver Link für die ganze Seite festgelegt werden (siehe Zeile 6). Die Farben können als Farbname (nur wenige Farben gemäss HTML-Referenz) oder in Hexadezimaldarstellung definiert werden.

4. SMIL (Synchronized Multimedia Integration Language)

Soviele Möglichkeiten die Sprache HTML auch bietet, seine Ideen im Web zu präsentieren, hat sie doch einen grossen Nachteil: HTML-Seiten sind statisch, d. h. wenn sie einmal aufgerufen werden, kann ihr Inhalt (auf HTML beschränkt, d.h. abgesehen von JAVA-Scripts, JAVA-Applets u.ä.) nicht mehr verändert werden. Es gibt keine Möglichkeit,mit einem HTML zeitgesteuerte Abläufe ins Spiel zu bringen.
Mit SMIL (ausgesprochen wie „smile“), einer erst seit 1998 von W3C verabschiedeten Sprache für Multmedia im Web, können zeitabhängig gesteuerte Seiten erstellt werden. Eine weitere Eigenschaft dieser Sprache ist es, benutzerspezifische Eigenschaften wie verwendetes System, Bildschirmauflösung und ähnliches für die Ausgabe der Seite zu berücksichtigen, wie die folgende switch-Anweisung zeigt:

```
<switch>
  <audio system-bitrate="44000" src=hi-res.aiff />
  <audio system-bitrate="16000" src=low-res.aiff />
</switch>
```

Hier wird je nach eingebauter Soundkarte zwischen der Qualität der zwei .aiff-Dateien unterschieden.
Hier ein Beispiel für die zeitabhängigen Möglichkeiten von SMIL:

```
<par>
  <text src="Title.html" region="dur="5s"/>
  <video id=Video1 src=news.mpg" region="MainVideo begin="1.4s"/>
  <audio src="news.aiff2 begin="id(Video)(5.0s)"/>
<par/>
```

Damit das Video zunächst ohne Ton startet, beginnt die Audiodatei eine halbe Sekunde später. Hier beginnen die Abläfe parallel (<par>), es ist auch möglich, Dateien automatisch nacheinander aufrufen zu lassen (<sec>).
Eine sehr nützliche Anwendung ist auch das zeitgesteuerte Einblenden von Text. So können auf einfache Weise Untertitel für Videos realisiert werden. Die Links können ebenfalls zeitliche Abhängigkeiten haben, so kann man mit einem Klick zu einer bestimmten Stelle eines Films gelangen.
Eine besonders atraktive Anwendung ist sicher eine multimediale, zeitgesteuerte Präsentation in SMIL. Der Vorteil beispielsweise gegenüber einer Makromediapräsentation wäre, dass sich dieselbe Präsentation sowohl als CD wie auch als Webseite benützen lässt. Wie bei HTML, benötigt man auch für SMIL-Seiten einfach einen entsprechenden Browser. Da SMIL jedoch nicht in HTML-Browsern läuft, sondern man auf spezielle Software wie RealPlayer G2 oder GRiNS angewiesen ist, kann man noch nicht davon ausgehen, dass die Mehrheit der Internetbenutzer SMIL-Seiten so ohne weiteres benützen kann.
In Zukunft soll dieses System auch in Fernseh- und Videogräten zur Anwendung kommen.

5. Schlusswort

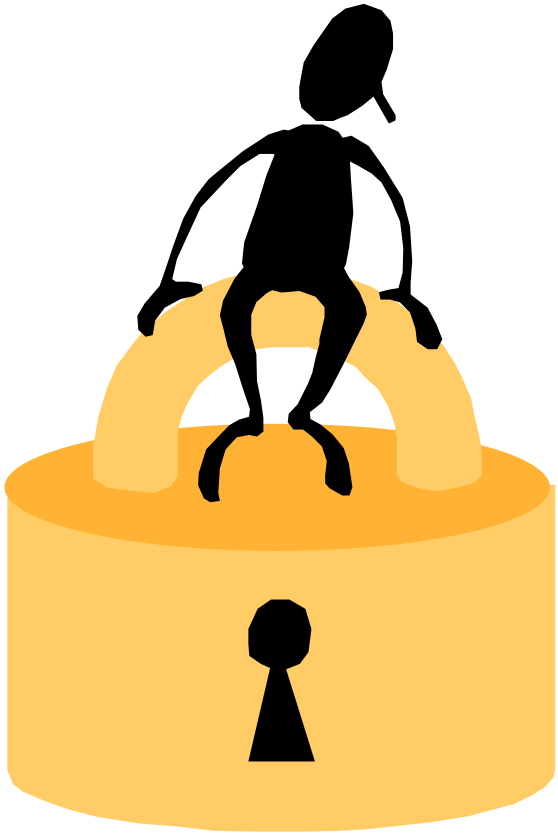
HTML wird sicher für die nächsten Jahre weiterhin als Standardbeschreibungssprache für Webseiten Verwendung finden. Es ist, wie schon gesagt, einfach anzuwenden und universell einsetzbar, sowie plattformunabhängig, was von grosser Wichtigkeit ist. Daneben werden Seiten auch in weiteren Formaten wie z. B. php3, asp, oder Flash geschrieben, jedoch sind dies bereits spezialisiertere Sprachen, die z. B. auch die Einbindung von Datenbanken oder zeitlich gesteuerte Abläufe ermöglichen.
Wie ich schon am Anfang erwähnte, ist die Einhaltung des HTML 4.0 Standards bei den Browsern ein leidiges Thema. Deren bekannteste Vertreter Netscape Communicator und Microsoft Internetexplorer unterstützen leider nicht immer dieselben Funktionen auf dieselbe Weise. So wird beispielsweise ab HTML 4.0 für viele Funktionen auf die Verwendung von Style-Sheets verweisen, was Netscape nur teilweise unterstützt. Hoffen wir auf eine diesbezüglicher Verbesserung in Netscape 6...
Die Bedeutung von SMIL ist für mich schwer abzuschätzen, da dies ein sehr neues Produkt ist, weiss natürlich noch niemand, wie sehr es Verbreitung und Anwendung finden wird. Meiner Meinung nach wird es je länge je mehr eine grosse Bedeutung haben für die bereits erwähnten Verwendungszwecke.

Quellenangaben:

- E. Wilde: World Wide Web - Technische Grundlagen; Springer Verlag, Berlin, Deutschland, 1999, Seiten 191-249
- L. Rutledge: SMIL: Synchronized Multimedia Integration Language, iX, Heft 10, 1999, Seiten 58-63
- Günter Born: HTML 4; Markt & Technik, Buch- & Softwareverlag, 1998
- Selfhtml; http://computing.ee.ethz.ch/.soft/selfhtml/selfhtml.html
- Titelbild: Homepage des W3C für HTML

PPS Grundlagen des Internet

Vortrag 8



Sichere Kommunikation – SSL, SSH

Thomas Hug

Einführung

Obwohl die Authentizität für eine Reihe von Anwendungen ausreichend sein mag, besteht in sehr vielen Fällen ein Bedarf an Geheimhaltung und somit an Mechanismen zum Sicherstellen, dass alle zwischen Client und Server ausgetauschten Informationen ausschliesslich von diesen beiden Partnern interpretiert werden können. Im allgemeinen ist es nicht möglich, das Netzwerk physisch vor Angreifern zu schützen. Die dem Internet zugrundeliegende Architektur bietet nur wenige Möglichkeiten zur Einflussnahme auf die Art und Weise der Datenübertragung zwischen zwei Kommunikationspartnern. Aus diesem Grund finden Mechanismen Anwendung, mit deren Hilfe durch Lauschen oder andere Methoden erlangte Daten für den Angreifer wertlos werden. Dies wird durch die Verwendung von Verschlüsselungsmethoden erreicht.
Es gibt jedoch verschiedene Arten von Angriffen, und in Abhängigkeit von den zum Sichern der Kommunikation verwendeten Verfahren sollte es bekannt sein, welche Arten von Angriffen technisch möglich und zu erwarten sind. Ein einfaches, Lauschangriff genanntes Verfahren (Abbildung 1), stellt eine einfache Möglichkeit zum Angreifen einer Internet-Verbindung dar und besteht aus einer dritten Stelle, die alle zwischen zwei Kommunikationspartnern übertragenen Daten abhört.

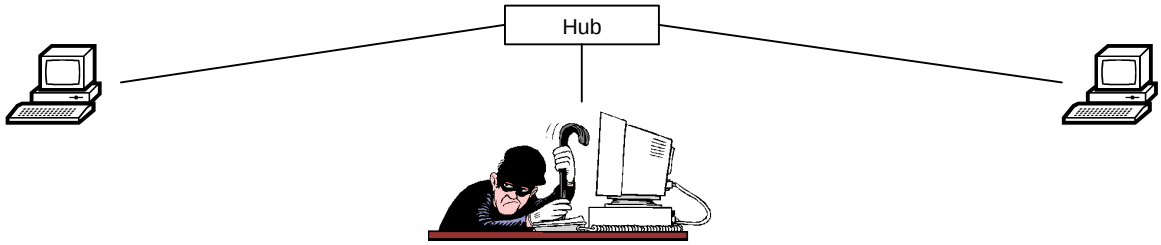


Abbildung 1 – Angriffsmöglichkeit nur durch Lauschen

Ein komplizierter und wirkungsvollerer Angriff (Abbildung 2) besteht in der Plazierung einer Mittelperson in die Mitte zwischen den beiden Kommunikationspartnern. Dieser Angreifer kann nun die Daten vor der Übertragung an die Gegenstelle verändern. Obwohl man sich mit Hilfe einiger Verfahren zur Kommunikationssicherheit vor Lausch- und Mittelpersonenangriffen schützen kann, gibt es andere Verfahren, die lediglich einen Schutz vor Lauschangriffen darstellen und bei Angriffen über eine Mittelperson unwirksam sind.

Abbildung 2 – Angriffsmöglichkeit Mittelperson

Aus Anwendungssicht gesehen existieren zwei Wege zum Sichern der Kommunikation zwischen Client und Server, je nachdem, ob die Sicherheitsmassnahmen innerhalb der Transportinfrastruktur oder innerhalb der Anwendungen implementiert werden sollten.

- *Verwenden einer sicheren Transportarchitektur (Network Level Security)*
  Bei diesem Szenario wird das Anwendungsprotokoll nicht verändert. Es wird davon ausgegangen, dass die Transportinfrastruktur selbst die Sicherheit bereitstellt, so dass als einzige zu entscheidende Frage übrigbleibt, ob die normale (unsichere) oder die sichere Transportinfrastruktur verwendet werden sollte und wie sich eine die sichere Transportinfrastruktur einsetzende Verbindung herstellen lässt. Diese Art der Sicherheit kann von allen Anwendungen genutzt werden, welche die sichere Transportinfrastruktur erkennen.

- *Verwenden eines sicheren Protokolls auf Anwendungsebene (Application Level Security)*
  Im zweiten Fall wird davon ausgegangen, dass die Transportinfrastruktur unsicher ist und aus diesem Grund das Anwendungsprotokoll dahingehend abgeändert wird, dass es selbst über Sicherheitsmerkmale verfügt. Bei diesem Ansatz sind die Anforderungen bezüglich der Transportinfrastruktur wesentlich niedriger, aber die innerhalb der Anwendung zu erledigenden Arbeiten (das Hinzufügen von Sicherheitsmerkmalen zum Protokoll der Anwendungsebene) um so schwieriger. Darüber hinaus lassen sich die Sicherheitsmerkmale lediglich für eine bestimmte Anwendung verwenden.

SSH – Secure Shell

SSH bezeichnet ein Protokoll und eine Software-Suite zur kryptographischen Absicherung unterschiedlicher Kommunikationskanäle über potentiell unsichere Netzwerke. Die Verschlüsselung basiert auf dem asymmetrischen Public-Key-Prinzip in Kombination mit symmetrischer Verschlüsselung. Dabei kann nur ein passender privater Schlüssel Daten entschlüsseln, die mit dem öffentlichen Schlüssel kodiert wurden und umgekehrt.
In der Praxis schickt ein Server „Challenges" (Herausforderungen) an einen Client. Ist dieser in der Lage, eine Challenge korrekt zu entschlüsseln oder kann der Server eine vom Client signierte Challenge verifizieren, so gilt der Test als bestanden.
Beim Aufbau der Verbindung erhält zunächst der Client den öffentlichen Schlüssel des Servers. Er generiert einen für jede Verbindung neu zu schaffenden symmetrischen Schlüssel, verschlüsselt ihn mit dem öffentlichen Key des Servers und schickt ihn diesem. Von nun an verläuft die Übertragung sämtlicher Daten zwischen beiden Rechnern verschlüsselt.
Bei der Ermittlung der Zugangserlaubnis lassen sich hauptsächlich vier Verfahren unterscheiden, wovon die ersten beiden als rechner-, die letzten beiden als anwenderbasiert gelten:

- Reine *~/.rhosts*- und *hosts.equiv*-basierte Authentifizierung. Gilt wegen der leicht vortäuschbaren falschen Rechneridentität als hochgradig unsicher und wird von SSH2-Servern nicht mehr unterstützt
- Rechnerbasierte Public-Key-Authentifizierung: Ein Benutzer erhält wie beim ersten Verfahren Zugang zum System, jedoch prüft die Software zusätzlich die Identität des Client-Rechners über das Public-Key-Prinzip.
- Benutzerbasierte Public-Key-Authentifizierung: Der Benutzer bestätigt seine Identität mit seinen eigenen Schlüsseln.
- Passwortauthentifizierung: Schlagen die ersten Verfahren fehl bzw. hat der Benutzer oder der Client-Rechner gar keine Schlüssel definiert, fällt der Server auf die herkömmliche Passwortauthentifizierung zurück. Da seit Beginn der Verbindung die Verschlüsselung aktiv ist, geht nie ein Passwort unverschlüsselt zum Server.

Mit SSH lassen sich herkömmliche r-Tools und Telnet komplett ersetzen. Einsatzmöglichkeiten sind Remote-Logins, Remote-Ausführung textbasierter und grafischer Programme sowie Dateiübertragung mit Verschlüsselung und Kompression.
Weitere Einsatzgebiete eröffnen sich durch Port-Forwarding. Dabei tunnelt SSH während einer Verbindung alle auf einem lokalen Port eintreffenden Pakete und leitet sie verschlüsselt und eventuell komprimiert an einen Port auf dem Zielrechner. Damit lassen sich zahlreiche TCP/IP-basierte Kommunikationsprotokolle wie POP und IMAP ohne eine Erweiterung der ursprünglichen Programme verschlüsseln

SSL – Secure Socket Layer

SSL verdankt seine Bezeichnung der gängigsten Programmierschnittstelle von TCP/IP, der ursprünglichen in frühen UNIX-Betriebssystemversionen implementierten Sockets-Library[1]. Obwohl TCP/IP ein wohldefiniertes Protokoll für die Kommunikation zwischen Computern darstellt, ist nicht definiert, wie auf die von ihm bereitgestellten Dienste innerhalb einer Programmierumgebung zugegriffen werden kann. UNIX-Sockets in der von Stevens beschriebenen Form haben sich zum DE-facto-Standard bei der Netzwerkprogrammierung entwickelt. SSL gibt jedoch keine einheitliche Programmierschnittstelle an. Somit weisen unterschiedliche SSL-Implementierungen verschieden Programmierschnittstellen auf und sind nicht ohne Veränderungen an der sie verwendeten Software austauschbar. Der Dienst bleibt jedoch immer der gleiche.
Mit dem Ziel vor Augen, eine sichere Kommunikation über ein unsicheres Medium zu ermöglichen, definiert SSL ein Protokoll, das eine Verbindungssicherheit bereitstellt, die drei grundlegende Eigenschaften besitzt:

- *Verbindungssicherheit*
  Nach einem anfänglichen Handshake wird mit Hilfe eines Verschlüsselungsverfahrens ein geheimer Schlüssel definiert. Zur Datenverschlüsselung wird eine symmetrische Verschlüsselungsmethode verwendet.
- *Optionale Authentifizierung*
  Die Identität des Kommunikationspartners kann mit Hilfe eines asymmetrischen Verschlüsselungsverfahrens (d.h. mit Hilfe eines öffentlichen Schlüssels) authentifiziert werden.
- *Zuverlässigkeit einer Verbindung*
  Die Verbindung ist zuverlässig. Die Nachrichtenübertragung schliesst eine mit Hilfe eines verschlüsselten *Message Authentication Code (MAC)* vorgenommene Integritätsüberprüfung der Nachrichten ein. Die MAC-Berechnung wird unter Verwendung sicherer Hash-Funktionen vorgenommen.

[1] Socket: Zugangspunkt

Im folgenden werden die Ziele des SSL-Protokolls nach ihrer Wichtigkeit geordnet aufgeführt.

- *Kryptografische Sicherheit*
  SSL sollte dazu verwendet werden, eine sichere Verbindung zwischen zwei Stelle aufzubauen.
- *Interoperabilität*
  Unabhängige Programmierer sollten SSL einsetzende Anwendungen entwickeln können, die in der Lage sind, Verschlüsselungsparameter auszutauschen, ohne jeweils den Code der anderen zu kennen.
- *Erweiterbarkeit*
  SSL versucht, einen Rahmen bereitzustellen, innerhalb dessen sich neue Verfahren sowohl zur Erstellung von öffentlichen Schlüsseln als auch zur Verschlüsselung grosser Datenmengen dem Erfordernissen entsprechend miteinander verbinden lassen. Auf diese Weise werden auch zwei Sekundärziele erreicht: es entfällt die Notwendigkeit der Erstellung eines neuen Protokolls (und damit die Möglichkeit des Auftretens neuer Schwächen), und es wird umgangen, eine vollkommen neue Sicherheitsbibliothek implementieren zu müssen.
- *Relative Wirksamkeit*
  Kryptografische Verfahren, insbesondere Operationen mit öffentlichen Schüsseln, sind häufig sehr rechenintensiv. Aus diesem Grund enthält SSL ein Schema zum Caching von Sitzungen, um so die Anzahl der von Grund auf neu aufzubauenden Verbindungen zu reduzieren. Ausserdem wurde darauf geachtet, die Netzwerkbelastung gering zu halten.

Im allgemeinen lassen sich mit Hilfe von SSL drei unterschiedliche Arten von Verbindungen zwischen Client und Server aufbauen, die sich bezüglich des jeweils eingesetzten Authentifizierungsverfahrens unterscheiden. Zum Zwecke der Authentifizierung wird ein von einer akzeptablen Authentifizierungsstelle ausgegebenes Zertifikat benötigt.

- *Anonymität*
  Bei diesem Szenario werden weder der Client noch der Server authentifiziert
- *Server-Authentifizierung*
  Bei der Server-Authentifizierung muss der Server ein vom Client akzeptiertes Zertifikat vorweisen. Obwohl dem Server die Identität des Clients nicht bekannt ist, kann sich der Client über die Identität des Servers sicher sein.
- *Authentifizierung beider Parteien*
  Bei diesem Szenario werden sowohl der Client als auch der Server durch Zertifikate authentifiziert, d.h. es kennt jeder die Identität des anderen.

Man sollte beachten, dass das anonyme Szenario lediglich Schutz vor Belauschung bietet, während Angriffe von zwischengeschalteten Personen immer noch möglich sind. Falls SSL in einer Umgebung eingesetzt wird, in der mit einem solchen Angriff gerechnet werden muss, sollte zumindest eine Server-Authentifizierung zum Schutz vor Angriffen von zwischengeschalteten Personen stattfinden.
SSL besteht aus zwei Phasen. Während der ersten Phase findet ein Handshake statt, bei dem die jeweiligen Fähigkeiten beider Seiten festgestellt werden und eine optionale Authentifizierung vorgenommen sowie das bei dieser Sitzung verwendete Verschlüsselungsverfahren ausgewählt wird. SSL basiert auf dem Sitzungskonzept. Unter Verwendung leistungsfähiger Verschlüsselungsverfahren wird ein Sitzungsschlüssel ausgetauscht, der zum Verschlüsseln der zwischen Client und Server ausgetauschten Daten dient. Dieser Sitzungsschlüssel verwendet ein schwächeres (aber effizienteres) Verschlüsselungsverfahren als das zum Austauschen der Schlüssel eingesetzte. Dies ist vertretbar, da der Schlüssel lediglich für die Dauer einer Sitzung eingesetzt wird. Falls eine der beiden Seiten davon ausgeht, dass der Sitzungsschlüssel nicht mehr sicher ist, wird ein neuer Handshake inklusive der Erzeugung eines neuen Sitzungsschlüssels eingeleitet.
SSL definiert eine Reihe unterschiedlicher Algorithmen sowohl für den Schlüsselaustausch als auch für den Sitzungsschlüssel unterstützten Algorithmen.

Anwendungsbeispiel – HTTPS (HTTP + SSL)
Obwohl normales HTTP das auf der Anwendungsschicht zwischen einem HTTP einsetzenden Client und einem entsprechenden Server eingesetzte Protokoll darstellt, muss der Client wissen, dass er anstelle einer normalen (unsicheren) TCP-Verbindung eine SSL-Verbindung zu einem Server aufbauen muss. Dies wird mit Hilfe eines neuen Naming Schemes für HTTPS erreicht, in dem das Präfix „https“ für URLs definiert ist. Der Browser greift dann bei einem Request, anstelle des normalen HTTP-Port 80, auf den SSL-Port 43 zu.
Obwohl SSL (oder TSL) als Transportschichtprotokoll entworfen ist, das nicht nur die Verwendung mit einem speziellen Anwendungsschichtprotokoll vorsieht, ist es zur Zeit in keiner der von den meisten Betriebssystemen bereitgestellten standardmässigen Transportprotokollschichten enthalten. Aus diesem Grund muss die Anwendung eine SSL-Implementierung enthalten.
Das Ziel wäre, dass SSL Bestandteil jedes Betriebssystems wird. Bis dies erreicht ist, wird wohl noch geraume Zeit verstreichen.

S-HTTP

Da die Verschlüsselungsfähigkeiten von HTTPS (SSL) und S-HTTP sehr ähnlich sind und S-HTTP weitaus weniger verbreitet ist, wird hier nicht mehr speziell darauf eingegangen.

PGP – Ein Beispiel für Application Level Security

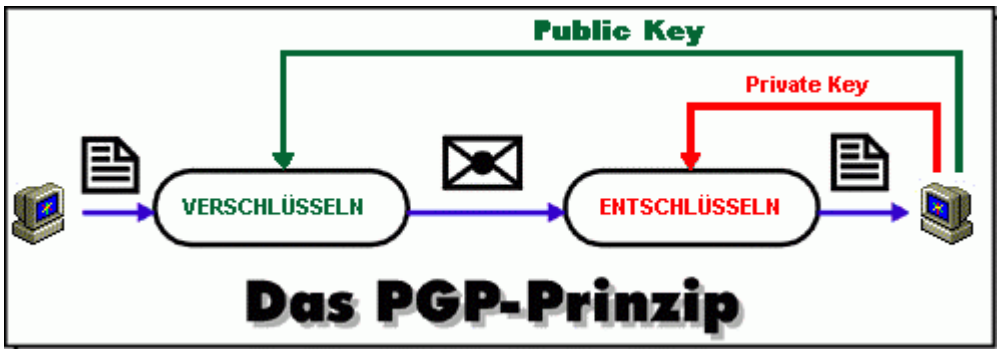


Abbildung 3

Die Verschlüsselung bzw. Entschlüsselung von PGP funktioniert mit zwei Schlüsseln (keys) (Beispiel im Anhang). PGP erzeugt ein Schlüsselpaar, das zur Verschlüsselung und Entschlüsselung von Nachrichten dient. Einen öffentlichen (public) und einen privaten (secret) Schlüssel, die das Programm jeweils getrennt in 2 Schlüsselringen (key rings) speichert. Der öffentliche Schlüssel (public key) ist im pubring, der private Schlüssel (private key) ist im secring gespeichert. Der öffentliche Schlüssel (public key) wird an ein für alle zugängliches Depot (key server) geschickt oder als Anhang an Emails verteilt. Der dazugehörige private Schlüssel (private key) wird sicher, separiert und für niemanden sonst zugänglich (!) aufbewahrt.
Wenn nun A eine private Nachricht an B schicken möchte, holt sich A den öffentlichen Schlüssel (public key) von B von einem keyserver oder direkt von B. Nun verschlüsselt A die Nachricht mit Hilfe des öffentlichen Schlüssels (public key) von B und schickt die verschlüsselte Nachricht dann an B.
Wenn B die Nachricht erhält, entschlüsselt er sie mit seinem privaten Schlüssel (private key) . Kein anderer Empfänger kann diese Nachricht entschlüsseln, weil dazu der Private Key von B und dessen persönlicher Code (passphrase) notwendig sind.

Ein Beispiel für Network Level Security ist IPv6. (siehe Vortrag 4).

Zusammenfassung

Für jede gängige Applikation gibt es eine Möglichkeit verschlüsselt zu kommunizieren. Wichtig ist immer, die verschiedenen Eigenschaften der Methoden genau zu kennen um eine geeignete Lösung (je nach Angriffsmodellen, Dauer der Verbindung etc.) für die eigene Umgebung zu finden.

Ein kurzer Überblick über die erwähnten Methoden:
- SSH: Secure Shell; dient als Ersatz für Telnet
- SSL: Secure Socket Layer; Zusatz zum HTTP-Protokoll
- PGP: Pretty good Privacy; Verschlüsselung von Emails oder normalen Texten

Bibliographische Angaben

[1] S. Leich: *Doppelt genäht*; iX, Heft 1, 2000, Seiten 146-149.
[2] E. Wilde: World Wide Web – Technische Grundlagen; Springer Verlag, Berlin, Deutschland, 1999, Seiten 127-135
[3] Internet: http://eurocontact.org/PGP/deutsch/funktion.htm

Anhang

Beispiel eines (meines) PGP public keys:

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGPfreeware 5.0i for non-commercial use

mQGiBDjx8iQRBADP5OzVw8p13UeiCvL0Xez5UwDlJ/byoguI492bSWy3gLzHHTVg
M4VA1N4GNf8HKTTED5fmbm5uMGX882ZEmCk5Z6cksGyttYEoZMf+pt7+/BIN0Fcs
5cTMJJJiCoELTqkEP7ggv6eJWsltIAfJQu/2t5ca2V1rkW7AJRR760aiZQCg/3tV
aZKtOoNaKAr3P+iboWPkvnED/iAzSSxzSY9MWcg5+IF7cHtMEyGZUvzqzbGvx5CM
6sizKVEMtmMy91UHLXeYEO/Mxa7JnVA+HkZBAri2CaXTwQRcpgjk0Ic81MRaKAT7
Eg2ZZ9fMHgGBIBP5Xs/aWZsBXu+q40wDS4g3XRVry2FASgXMlhriWT21MDj9yxr7
dGdYA/9JPJTtyRpAdIU1TSOPL/xc/Aqcm9yTBdfw4/t3baagx/Ejln+s2gOfWhOO
aCof6xjP7th0V7NIsDNyFfuZMTj9MZJ5dJnJW+Dp04xHNRy7RbCZqKDJTM7fSSeF
MCmEEfhd/+S75wXaAUU3PRNnwjam7r51D4yrSaSIdWas0ctcZLQSdGhvbWh1Z0Bl
ZS5ldGh6LmNoiQBLBBARAgALBQI48fIkBAsDAQIACgkQ1Zod7dsV2rTn8gCgm1K7
kENCDm2e+2lEiU68q1P3T2IAoMEVRlDsEufbl+wsvA1OTW5kzGh2uQINBDjx8iUQ
CAD2Qle3CH8IF3KiutapQvMF6PlTETlPtvFuuUs4INoBp1ajFOmPQFXz0AfGy0Op
lK33TGSGSfgMg71l6RfUodNQ+PVZX9x2Uk89PY3bzpnhV5JZzf24rnRPxfx2vIPF
RzBhznzJZv8V+bv9kV7HAarTW56NoKVyOtQa8L9GAFgr5fSI/VhOSdvNILSd5JEH
NmszbDgNRR0PfIizHHxbLY7288kjwEPwpVsYjY67VYy4XTjTNP18F1dDox0YbN4z
ISy1Kv884bEpQBgRjXyEpwpy1obEAxnIByl6ypUM2Zafq9AKUJsCRtMIPWakXUGf
nHy9iUsiGSa6q6Jew1XpMgs7AAICCAC7eMcP/salwSaZ7/eFaWx3cIK6gWj3yY70
GSZKRaU8mBJN753OGjZefkH8mrJr9ZGsiP/DeJ5S1fFx3eu9u+ipwxSbUrqnzzud
sRtocGbakEL89d7xULfboVb5xISxn5GMggK+0f/N3P6GH239ovYNsB0flXzDdBka
FO/Fr9oi+pRGIi6C8tpc05NBRvBNWlq3R4jMMToafTJBTTH7OVOkPcoWgTVxyGLg
9rh6jHCh3S0rHBSgHLZC1MvjWR+IAXh5Wg6mMpN4mi05ChMjJPqoW55Cfcqzhe6D
P0V/g/S/V9TLM8d3aM8H1nEwbTqeojr66lVd5SMiJtk/78LiRP7TiQA/AwUYOPHy
JdWaHe3bFdq0EQJzuACguIjAD1ElmIPQcnellIUjbaST6hcAn1aqibpZJ5VbR7u4
Wy7AF5ywsgZf
=grV5
-----END PGP PUBLIC KEY BLOCK-----

Die Datenstrukturierungssprache XML

Vortrag 9
PPS – Grundlagen der Internet-Technologie

Marc Mallepell
6. Juni 2000

Die Datenstrukturierungssprache XML

1. Einführung

HTML hat sich inzwischen als Standartsprache des WWW durchgesetzt und befindet sich nach der stürmischen Entwicklung Mitte der neunziger Jahren auf dem Weg zum Industriestandart. Die Schwierigkeit von HTML hält sich in Grenzen, mit benutzerfreundlicher Software und diversen Dokumentationen ist heute praktisch jedermann in der Lage, seine eigenen Seiten fürs Web herzustellen. Warum dann eine neue Sprache wie die eXtensible Markup Language (XML)?

1.1 Was ist der Nachteil von HTML?

Die HTML Version 3.2 besteht ungefähr aus 70 Tags und über 50 Attributen, die zum Teil fest vorgegeben sind. Mit ihnen können strukturierter Text, multimediale Objekte (Bilder, animierte Bilder, Applets) und Hyperlinks zusammengefügt werden, was man als Web-Seite bezeichnet. Bei der Entwicklung von Web-Seiten trat die Orientierung an der Struktur des Dokuments zunehmend in den Hintergrund, der Wunsch der Autoren, das Layout stärker kontrollieren zu können, vermehrt in den Vordergrund. Durch diverse Erweiterungen wie etwa Cascading Style Sheets (CSS) besteht nun auch die Möglichkeit, das Layout vom eigentlichen Dokument zu trennen. Doch trotz allem ist HTML immer noch relativ "einfach" geblieben.
In der Praxis dient das World Wide Web inzwischen für eine Vielzahl von Informationssystemen als Oberfläche. Beispiele dafür sind Datenbanken mit HTML-Frontend, Mail-Archive, Handbücher oder Warenkataloge. Diese erfordern jedoch eine reichere innere Struktur als dies HTML auszudrücken vermag.
Bei der Umsetzung solcher Projekte in Web-Dokumente findet immer wieder ein Informationsverlust statt. Bei einer über das Web abfragbaren Datenbank verschwindet die vorhandene Strukturierung der Daten in einem Meer aus Tags auf der Client-Seite. Eine Nutzung, die über das Ausschneiden von Text mittels Copy&Paste hinausgeht, ist nicht mehr möglich.

1.2 SGML

Das vorhin angesprochene Problem hat seine Berechtigung. Es wäre doch durchaus nützlich, wenn die Möglichkeit bestünde Daten, von Dokumenten mittels Drag&Drop in lokal installierte Anwendungen einfach zu übernehmen. Dies wäre vor allem im immer noch aktuellen Intranet von grossem Vorteil.
Mit der Standard Generalized Markup Language, kurz SGML, steht eine Sprache zur Verfügung, mit der Dokumente für beliebige Medien hergestellt werden können, ohne die Struktur der enthaltenen Daten zu verlieren. Doch die Mutter aller Aufzeichnungssprachen ist relativ komplex und gerade wegen ihrem Syntax eher unbeliebt.

1.3 XML als Teilmenge von SGML

An der alljährlichen Konferenz der Gemeinde der SGML-Spezialisten wurde im November 1996 die eXtensible Markup Language (XML) erstmals vorgestellt. Das Ziel von XML ist dabei, genau die oben angesprochenen Probleme der Komplexität (komplizierte Zusatzoptionen) von SGML, den engen Grenzen und für gewisse Anwendungen stark beschränkten Möglichkeit von HTML zu lösen.
Durch Weglassen diverser komplexer und selten verwendeter Eigenschaften von SGML wurde eine Sprache entwickelt, die trotz des Verlustes dieser Features die Kernidee des strukturieren Markup von SGML mit all ihrer Leistungsfähigkeit übernommen hat: Jedes Dokument wird säuberlich in die Teile Inhalt, Struktur und Layout zerlegt.
XML ist wie HTML eine Sprache. Doch sind die Attribute und Funktion von Tags in HTML bereits vorgegeben. In XML sind keine solchen vordefinierten Tags vorhanden, vielmehr ist XML eine aufwärtskompatible Teilmenge von SGML. Jedoch ist XML noch längst keine ausgereifte Technologie, auch wenn letztes Jahr die Version 1.0 der XML-Spezifikation zum offiziellen Standard

wurde. Sicher ist aber auch, dass sehr viele Einzelfragen noch offen sind und erst geklärt werden, wenn mit XML an konkreten Projekten gearbeitet wird.
Ob XML trotz ihrer enormen Leistungsfähigkeit eines Tages einen ähnlichen Grad an Popularität wie es zum Beispiel HTML erreicht, bleibt abzuwarten.
Der vorliegende Bericht soll dazu dienen einen Teil dieser Leistungsfähigkeit anhand von einigen kleinen Beispielen zu zeigen. Es soll der Idee und dem Aufbau von XML nachgegangen werden und auch die Frage in den Raum gestellt werden in welchem Anwendungsbereich XML wirklich den entscheidenden Durchbruch schaffen könnte.

2. XML-Grundlagen

Wir wissen nun, welchen Hintergrund XML besitzt und welche Aufgaben es in etwa übernehmen soll. Nun ist es an der Zeit, sich zu fragen, was XML überhaupt ist und aus welchen Bestandteilen ein XML Dokument besteht.

2.1 Dokument-Typ-Definition

XML ist eine Metasprache für das Definieren von Dokumententypen. Anders ausgedrückt liefert XML die Regeln, die beim Definieren von Dokumententypen angewendet werden. Doch was ist ein Dokumenttyp?
Wir betrachten das Element Hello World . Es besteht aus einen Start-Tag und einem End-Tag. Ein weiteres Element sieht zum Beispiel folgendermassen aus XML ist Teilmenge von SGML . Beide Elemente sind von gleichem Typ.
Bei Dokumenten verhält es sich sehr ähnlich. Wenn in Dokumenttypen, dieselben Elemententypen verwendet werden, so gehören alle diese Dokumente dem gleichen Dokumenttyp an. So gesehen sind jegliche HTML-Dokumente vom selben Dokumenttyp, nämlich dem Typ HTML.

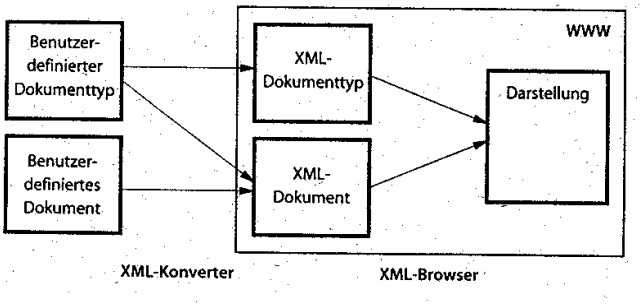


Abbildung 1

Mit XML kann man also seine eigenen Dokumenttypen definieren. Der Benutzer definiert auf der einen Seite den Dokumenttyp, auf der anderen Seite den Inhalt des Dokuments.

Die Abbildung 1 zeigt aus welchen Bestandteilen ein Dokument das mit Hilfe von XML erstellt wird, besteht. In diesem Abschnitt beschäftigen wir uns mit dem XML-Dokumenttyp, kurz mit der Dokument-Typ-Definition (DTD). Die DTD legt fest, wie in einer Datei des entsprechenden Typs die Daten organisiert werden. Bei der DTD handelt es sich um eine Grammatik, mit der Dokumenttypen hergestellt werden können.
Die Elementtyp-Deklaration ist die wichtigste Komponente der DTD. Sie besitzt die in Abbildung 2 gezeigte Gestalt:

Abb. 2

Ein kleines Beispiel soll veranschaulichen wie ein Dokumenttyp syntaktisch mit ein paar Elementen aussieht: Eine DTD für eine Adresse. Im Wesentlichen besteht eine Adresse aus einem Namen, Vornamen, einer Adresse, einer PLZ und einem Wohnort.

In der XML-Fassung sieht dies nun folgendermassen aus:

```
<!DOCTYPE adressensatz[
        <!ELEMENT name      (#PCDATA)>
        <!ELEMENT vorname (#PCDATA)>
        <!ELEMENT adresse   (#PCDATA)>
        <!ELEMENT plz       (#PCDATA)>
        <!ELEMENT ort       (#PCDATA)>
]>
```

Wie man sieht, besteht ein Dokument vom Typ Adressensatz aus den aufgelisteten Elementen und zwar genau in dieser Reihenfolge. Alle diese Elemente enhalten reinen Text, hier Parsed Character Data (PCDATA) .
Die in der DTD vereinbarten Tags werden als benannte Klammern verwendet, um den Inhalt in Elemente zu gliedern. Ähnlich wie in HTML wird <TAG> als Beginn und </TAG> als Ende des Elements gekennzeichnet. Durch die Schachtelung dieser Elemente entsteht eine Baumstruktur. In unserem Beispiel ist "adressensatz" die Wurzel. Die verschiedenen Instanzen bilden die Äste, die von der Wurzel ausgehen.

2.2 Instanz

Wie sieht nun ein Adressensatz im Detail aus? Dazu erstellen wir eine Instanz. Es ist also nichts anderes als ein Objekt vom Typ Adressensatz, das genau die in der DTD definierten Elemente "name", "vorname", "adresse", "plz" und "ort" enthält. Eine Instanz könnte folgendermassen aussehen:

```
<?xml version="1.0"?>
<!DOCTYPE adressensatz SYSTEM „adressensatz.dtd">
<adressensatz>
   <name>Muster</name>
   <vorname>Hans</vorname>
   <adresse>Gurtenstrasse 12</adresse>
   <plz>3003</plz>
   <ort>Bern</ort>
</adressensatz>
```

Man kann sich nun die Frage stellen, wozu eine DTD überhaupt nützlich ist. Die oben aufgelistete Instanz kann man auch ohne DTD lesen und verstehen. XML würde es auch erlauben, die Instanz ohne DTD zu schreiben. Dies ist jedoch nur nützlich, wenn man nur eine einzige Instanz von diesem Objekt hat. Sobald man mehrere Instanzen mit gleichem Inhalt, also zum Beispiel eine Sammlung von Adressen besitzt, so dient die DTD als Muster und Vorlage. Ein geeigneter Editor kann schon im voraus das Muster mit dem Datensatz vergleichen und auf eventuelle Fehler aufmerksam machen. Ohne DTD ist es nicht klar, aus welchen Elementen eine Adresse aufgebaut werden muss.

2.3 Attribute

Nachdem der Begriff Element geklärt ist, noch ein kurzer Hinweis auf die Deklaration der Attributlisten. Sie legen fest welche Attribute für einen bestimmten Elementtyp existieren.

```
<!ATTLIST   img
            src       CDATA     #REQUIRED
            alt       CDATA     #REQUIRED
            height    CDATA     #IMPLIED
            width     CDATA     #IMPLIED
                >
```

Mit Hilfe der Attribute kann unter anderem die Grösse, Schrift und Position innerhalb des Dokuments definiert werden. Die beiden notwendigen (REQUIRED) Attribute src und alt erwarten beide eine Zeichenkette als Wert (CDATA). Gleiche Wertetypen sind für die Höhen- und Breitenangabe erlaubt, jedoch müssen diese Werte nicht explizit angegeben werden. Das Anwendungsprogramm (bei HTML in der Regel der Web-Browser) muss gegebenenfalls die impliziten Werte (IMPLIED) verwenden. Bei Grafiken heißt das, der Browser muss die Ausmaße selbst ermitteln.

Zum Beispiel der vom HTML her bekannte Tag <h1>, besitzt ganz genau definierte Attribute. Diese sind im Dokumenttyp HTML festgelegt. Diese Attribute haben also den Zweck, dem Computer mitzuteilen, dass der Inhalt dieses Tags die Eigenschaften einer Überschrift haben soll.

2.4 Style Sheets

Da in XML nicht wie bei HTML jeder Tag eine feste Bedeutung hat, sind Style Sheets eine absolute Notwendigkeit. Noch ist nicht entschieden, welches Style-Sheet-Modell XML in Zukunft verwenden soll. Die Cascading Style Sheets werden bei XML wieder zu einem Thema. Hatte dieses Modell kaum erfolg bei HTML, es eine erneute Chance, da bei XML-Dokumenten nicht die selben Annahmen zutreffen wie bei HMTL. Zwar ist CSS in einigen Fällen nicht ausreichend genug, doch reichen die Möglichkeiten aus, ein Dokument darzustellen. Solange die Fähigkeiten von CSS ausreichen, kann sich der Benutzer entscheiden, ob er doch nicht lieber dieses Style-Sheet-Modell betrachtet, anstelle des komplexeren eXtensible Stylesheet Language (XSL).

2.5 Dokumente und Linking

Dokumente bilden das zentrale Objekt. Ein Dokument setzt sich, wie die Abbildung 1 zeigt aus der Definition des Dokumenttyps und dem Inhalt des Dokuments zusammen. Sie bilden die Grundlage von XML-Dokumenten. Neben den oben gezeigten Beispielen ist mit Hilfe von spezielleren DTDs weit aus mehr möglich, als nur gerade einfache Elemente zu deklarieren. Diese Elemente und die Kombination dieser Elemente noch weiter zu spezifizieren wäre kein Problem. Auch Daten aus anderen Dokumenten in das eigene einzublenden, wie es zukünftig mit XML möglich sein soll, ist eine von vielen weiteren Möglichkeiten, die mit XML erst möglich werden.
Es muss auch nicht nur ein Einblenden von neuem Inhalt in den laufenden Text sein, genauso gut könnte man ein Extra-Fenster mit dem gewünschten Eintrag öffnen, so bald jemand eine bestimmte Aktion mittels eines Linkfeldes oder etwas ähnlichem hervorruft.
Doch die Semantik, die es dazu benötigt, würde schnell einmal ins Detail führen und wäre hier in einer Einführung in XML weniger geeignet.

3. XML Extensible Markup Language 1.0

Im Kapitel Grundlagen haben wir ein kleines Beispiel kennengelernt und gesehen, wie ein XML-Dokument in etwa aussehen könnte. Ziel dieses Kapitels ist es nun, die Sprache an sich etwas technischer zu beschreiben, wie sie vom World Wide Web Consortium (W3C) definiert worden ist.
Wie schon erwähnt, ist XML eine Teilmenge von SGML, die dazu entworfen wurde eine einfache Implementierung und Zusammenarbeit sowohl mit SGML als auch mit dem weit verbreiteten HTML zu gewährleisten. Sie beschreibt eine Klasse von Datenobjekten (XML-Dokumente) und beschreibt teilweise auch das Verhalten von Computer-Programmen (insbesondere von Browsern), die solche Dokumente verarbeiten. Die analysierte Daten bestehen aus Zeichen, von denen einige Zeichendaten darstellen, andere aber Anweisungen über den Aufbau und Layout des Dokuments darstellen.

3.1 Herkunft und Ziele

XML wurde von einer Arbeitsgruppe entwickelt, die man auch unter dem Namen SGML Editorial Review Board kennt. Sie wurde 1996 innerhalb des World Wide Web Consortium gegründet.

Entwurfsziele für XML sind:

1. XML soll sich im Internet auf einfache Art und Weise nutzen lassen
2. XML soll ein breites Spektrum von Anwendungen unterstützen
3. XML soll kompatibel zu SGML sein
4. XML-Dokumente sollten für Menschen lesbar und angemessen verständlich sein
5. XML-Dokumente sollen leicht zu erstellen sein
6. Es sollte einfach sein, Programme zu schreiben, die XML-Dokumente verarbeiten

Dies sind nur einige der wichtigsten Punkte für die man XML entwickelte.

3.2 Dokumente

Ein Datenobjekt ist ein XML-Dokument. Jedes Dokument hat sowohl eine logische als auch eine physikalische Struktur. Physikalisch besteht das Dokument aus einer Reihe von Einheiten, genannt Entities. Diese können auf andere Entities verweisen um sie in das Dokument einzubinden. Aus logischer Sicht bestehen Dokumente aus Deklarationen, Elementen, Kommentaren, Zeichenreferenzen und Processing Instructions, die innerhalb des Dokuments durch explizites Markup ausgezeichnet sind. Logische wie physikalische Strukturen müssen korrekt verschachtelt sein.

3.3 Adressen der Definitionen

Die weiteren Definitionen und Inhalte von logischen, physikalischen Strukturen, Konformität und die Notation von XML sind in [3] und [4] zu finden.

3.5 Unterschiede zu SGML

Da XML „nur“ eine Teilmenge von SGML darstellt, gibt es doch einige Unterschiede zu SGML, die vielleicht kurz hervorgehoben werden sollten.

Processing Instruction Delimiter	Im SGML-Referenzsyntax wird der Processing Instruction Close Delimiter durch das Zeichen ‚>’ dargestellt, in XML ist das Zeichen ‚?>’ dafür zuständig.
Element-Type-Declaration	Jeder Elementtyp muss in XML über eine eigene Elementtypdeklaration verfügen. Es besteht nicht die Möglichkeit eine Elementtypdeklaration für eine Gruppe von Elementen zu verwenden.
No Exceptions	In Elemtentypdeklarationen ist es nicht möglich Exceptions zu verwenden. So ist es erforderlich den zulässigen Inhalt für jeden Elementtyp explizit zu deklarieren.
Declaration of Attribute-Definition-Lists	XML erlaubt nicht alle „Attribute Declared Values“ zu verwenden, die in SGML definiert sind. Jedes Element muss in XML seine eigene Deklaration einer Attributdefinitionliste besitzen.

Daneben bestehen noch weitere wichtige Unterschiede, die hier jedoch nicht anmerken werden, da sie bei unserem Stand der Kenntnisse über SGML nichts auszudrücken vermögen.

4. XML Linking Language (Xlink)

Wer Hypertext im allgemeinen und HTML im besonderen kennt, will natürlich die Möglichkeit nutzen, Textstellen und Dateien mit Hilfe eines Links miteinander zu verknüpfen. Es bestehen in XML zwei Möglichkeiten, dies zu realisieren: Xlink und Xpointer. Die Spezifikation des Linking ist im Vergleich zum Syntax der Sprache noch nicht abgeschlossen.
Xlink nimmt die Möglichkeit von HTML auf: Links zwischen Dateien und Dateiteilen, Adressangaben von Bildern und ImageMaps. Xpointer erlaubt es auf Dokumentteile zuzugreifen. Es wird hier aber nicht näher auf diese Möglichkeit eingegangen.

Im Vergleich zu HTML sind Links in XML jedoch genauer zu bestimmen. Man unterscheidet grundsätzlich zwei Arten von Links: Einfache und erweiterte.

4.1 Einfache Links

Der einfache Link entspricht dem aus HTML bekannten und unterscheidet sich nur geringfügig:

HTML: <A HREF="http://www.ethz.ch">Eigenössische Technische Hochschule Zürich</A>

XML: <a xml:link="simple" href="http://www.ethz.ch"> Eidgenössische Technische …

XML behandelt Elementnamen unterschiedlich, wenn sie in Gross- und Kleinbuchstaben sind. Eine Adressangabe in XML enthält normalerweise einen Universal Resource Identifier. Das heisst unter anderem, dass die aus HTML bekannten URLs auch in XML verwendet werden können. Auch können Adressangaben auf einen Teil eines Dokuments zeigen.

4.2 Erweiterter Link

Im Gegensatz zum einfachen Link kann der erweiterte Link auf mehrere Ziele verweisen.

Doch was ist besonders am erweiterten Link?

1. Es ist zum Beispiel möglich, Verweise von Read-only-Medien oder Dateien zu anderen Stellen einzurichten.
2. Man kann Links zu und von Daten aus erzeugen, die selbst kein Linking unterstützen.

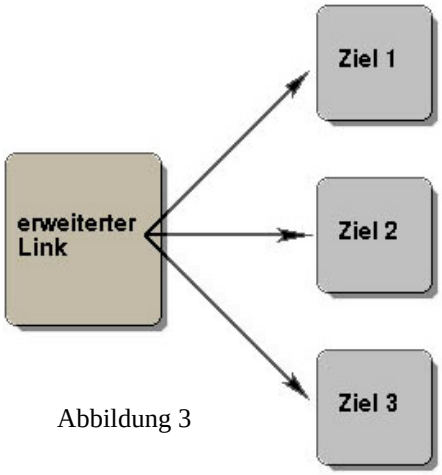


Abbildung 3

Innerhalb einer XML-Instanz könnte also ein Element mehrere Adressangaben beinhalten.

```
<erweitert xml:link="extended">
        <erwverweis href="http://www.ethz.ch "
         role="ETH" />
         <erwverweis href="http://www.ee.ethz.ch"
         role="D-Elek" />
         <erwverweis href="http://n.ehtz.ch"
         role="N-ethz" />
```

Im Beispiel wird der Linkt als erweiterter Link definiert. Innerhalb dieses Elements sind nun die eigentlichen Adressangaben aufgelistet. Zur Veranschaulichung ist die zugehörige DTD angefügt, auf die jedoch nicht näher eingegangen wird.

```
<!ELEMENT erweitert ANY>

        <ATTLIST erweitert
            xml:link     CDATA    #FIXED    "extended"
            role         CDATA    #IMPLIED
            content-role CDATA    #IMPLIED
            content-title CDATA   #IMPLIED>

        <!ELEMENT erwverweis ANY>

        <ATTLIST erwverweis
            xml:link     CDATA    #FIXED    "locator"
            href         CDATA    #REQUIRED
            role         CDATA    #IMPLIED>
```

Wiederum müssten die Anwendungen (wie zum Beispiel der Browser) aber Möglichkeiten bereitstellen, solche Links dem Anwender zu präsentieren. Beispielsweise müssten sie ein PopUp-Menü erzeugen können, dass die alternativen Adressen dem Betrachter aufzeigt und im die Wahl überlässt.

5. Zusammenfassung – XML im Web

XML Hypermedium, im Web eine neue alternative zu HTML. Mit dem Internet, wie wir es heute kennen, hat es nur die Transportschicht und den Adressmechanismus gemeinsam.
Die durch die existierenden und in ihrer Flexibilität beschränkten Werkzeuge hervorgerufene, getrennte Problematiken wie Hypertext, Datenbanken und Austauschformate können dank XML mit einem flexiblen Datentyp angegangen werden.
Für viele Anwendungen bleibt wohl auch in Zukunft HTML eine vollkommend ausreichende Markup-Language. Für die Textpräsentation mit einer eingeschränkten Interaktivität am Bildschirm, ist HTML wohl immer noch das geeigneteste Instrument. Zudem haben die vielen Erweiterungen von HTML, wie die Möglichkeit JavaScript oder Java-Applets einzubinden, schon dafür gesorgt, dass den Autoren mehr Möglichkeiten zur Verfügung stehen, seine Seite zu gestallten.
XML wird erst ab einer bestimmten Grösse der Seite zu einer interessanten Alternative. Potentielle Anwender sind, wenn überhaupt, bei grossen Verlagen, Einkaufshäusern und in der Industrie zu finden. Erst beim Auftauchen eines XML-Browsers, der im Reifegrad eines MSIE oder Netscape Navigators ist, könnte ein endgültiger Wandel vom HTML zum XML geschehen.
Eines zeichnet sich jedoch schon heute deutlich ab: XML stellt, falls es sich jemals durchsetzt, eine wichtige Innovation im World Wide Web dar.
Auch wenn man sich fragt, ob man HTML vielleicht nur bis zur Version 4.0 entwickeln soll und anschliessend XML als Nachfolger von HTML in Betrachtung zieht, so ist es doch immer eine Frage des Marktes, ob XML eine Chance bekommt sich zu etablieren.

6. Bibliographische Angaben

[1] E. Wilde: World Wide Web – Technische Grundlagen; Springer Verlag, Berlin, Deutschland, 1999, Seiten 359-393

[2] Behme, Henning; Mintert, Stefan: XML in der Praxis, m. CD-ROOM, ADDISON-WESLEY, MÜNCHEN, Neuaufl. 2000

[3] XML: Professionelle Alternative zu HTML
http://www.heise.de/ix/artikel/1997/06/106/artikel.html

[4] Das Web automatisieren mit XML
http://members.aol.com/xmldoku/

Server
für HTTP

Thorsten Eichholz
Grundlagen der Internet-Technologie
Vortrag 10 / 6.6.2000

Server für HTTP

Einleitung

Das HTTP-Protokoll [1] regelt den Zugriff auf den Server ; aber wie genau funktioniert eigentlich der Server an sich? Wie kann er den Inhalt eines Requests interpretieren, verstehen und daraufhin den passenden Response senden? Wie schafft er es, den Datenverkehr in die richtigen Bahnen zu lenken und dabei noch schnell und effizient zu sein? (Moderne Server bearbeiten hunderte von Requests pro Sekunde!) Der HTTP-Server ist also das wichtigste Bindeglied in der Web-Infrastruktur, obwohl der Web-Benutzer fast nie direkt damit in Berührung kommt.
Der erste HTTP-Server war ein sehr kleines Programm, das den Namen der angeforderten Ressource auf einen Dateinamen abbildete und den Inhalt dieser Datei als Antwort sendete. Heute sind über 50% aller in Betrieb befindlichen Server Apache-Web-Server. Dieser unterliegt der General Public License und ist kostenlos, aber dennoch sehr leistungsfähig und zuverlässig.

Server-Konfiguration

Die zwei grundsätzlichen Möglichkeiten, wie ein Web-Server bei einem bestimmten Host-Namen konfiguriert werden kann, sind entweder als Proxy-Server oder als Origin-Server (siehe Vortrag „HTTP“ von Marco Somaini [1]). Vom Server wird erwartet, dass er auf einen Request „wartet“, diesen bearbeitet und dann das Ergebnis als Response sendet. Da aber der Server auch nur ein Programm ist, das gestartet werden muss, gibt es zwei Möglichkeiten: entweder der Server läuft ständig, oder er startet nur auf Requests:
Bei Letzterer gibt es die Variante des Internet-Superservers, ein Programm, welches an einer Anzahl von Ports, die Diensten zugeordnet sind, auf Requests wartet. D.h er kann vieles bereitstellen, ohne dass diese Prozesse ständig auf dem Server laufen müssen. Bei einer grossen Anzahl von Requests verursacht der Internet-Superserver allerdings viele Prozessstarts und Beendigungen und ist somit ungeeignet. Da heutzutage fast ausschliesslich auf den (Internet-)Port 80 zugegriffen wird, ist das Modell des Internet-Superserver mittlerweile verdrängt worden.
Im Gegensatz zu diesem requestabhängigen Verhalten des Servers können beim permanenten Betrieb die Anfragen unmittelbar beantwortet werden.

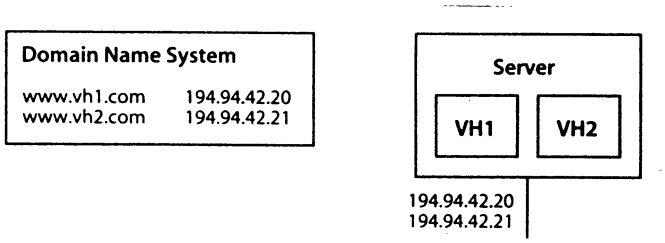
Ports:
(Ports ermöglichen das Ansprechen unterschiedlicher Dienste. Zusammen mit den IP-Nummern bilden die Portnummern Kommunikationsendpunkte. Verglichen mit beispielsweise einer Telefonanlage, stehen Portnummern auf der gleichen Stufe wie Nebenstellenanlagen. Die Netzadresse entspricht dabei der Ortsvorwahl, die Hostadresse der Rufnummer, und der Port entspricht der Durchwahl. Mit diesen Angaben kann eine Verbindung zu einem bestimmten Dienst aufgebaut werden).

Häufig werden auch sogenannte Virtuelle Hosts gebraucht: D.h. ein Server (einzelnes Programm, nicht die Hardware) kümmert sich um mehrere untergeordnete Hosts (siehe Abb.1!), die durch DNS-Namen identifiziert werden. Diese Methode vereinfacht die Verwaltung des Server erheblich, da man viele Hosts neu konfigurieren oder aktualisieren

kann, indem man nur einen Server aktualisiert. Oder man verwendet sie dort, wo sich die Konfiguration von anderen Hosts unterscheidet. Auch sinkt die CPU-Belastung, wenn nicht mehrere Server für einen Host laufen müssen und die einzelnen Prozesse können dynamischer zugeordnet werden.
Nun kann man Virtuelle Hosts IP-basiert oder Nicht-IP-basiert konfigurieren:

- IP-basiert: Der Server überwacht alle IP-Adressen, was bedeutet, dass die Konfiguration eine Liste enthalten muss, die allen virtuellen Hosts ihre IP-Adressen zuordnet. Wenn ein

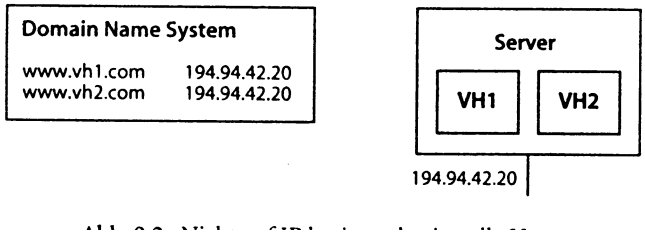
Abb.1:



IP-basierte Virtuelle Hosts

- Verbindungs-Request empfangen wird, weiss der Server, an welchen virtuellen Host dieser gerichtet ist. Diese Art der Konfiguration ist allerdings bei der heutigen Knappheit an IP-Adressen eine grosse Verschwendung, ausserdem kann es Probleme mit den Routern geben, die für einen solchen Betrieb nicht immer ausgelegt sind.

- Nicht-IP-basiert: In der neuen Version **HTTP 1.1** führte man das Header-Feld „HOST“ ein, welches den (virtuellen) Host-Namen enthält, für den der Request gesendet wurde. Der Server überwacht jetzt nur eine IP-Adresse, und da jeder Request nun eine Identifikation enthält, kann er entsprechend der Konfiguration für den virtuellen Host antworten. Der Vorteil dabei ist, dass es einfacher ist, wenn nötig, einen Virtuellen Host neu zu erzeugen, als einen IP-basierten (der ja schon festgelegt ist). Dies geschieht einfach mit einem neuen DNS-Eintrag mit der Adresse des Servers.



Nicht IP-basierte Hosts

<u>Behandlung von Requests</u>

Wenn sich eine Anfrage weder auf eine Ressource im Dateisystem noch auf ein auszuführendes Skript gibt es hierfür verschiedene Fehlermeldungen: eine für Tippfehler (Statuscode 404 (not found)), eine für eine veraltete URL, eine für eine nicht mehr verfügbare Ressource (Statuscode 401 (gone))....
Der Administrator kann nun in der Konfiguration festlegen, welche Verzeichnisinformationen über das Web zugänglich gemacht werden sollen:

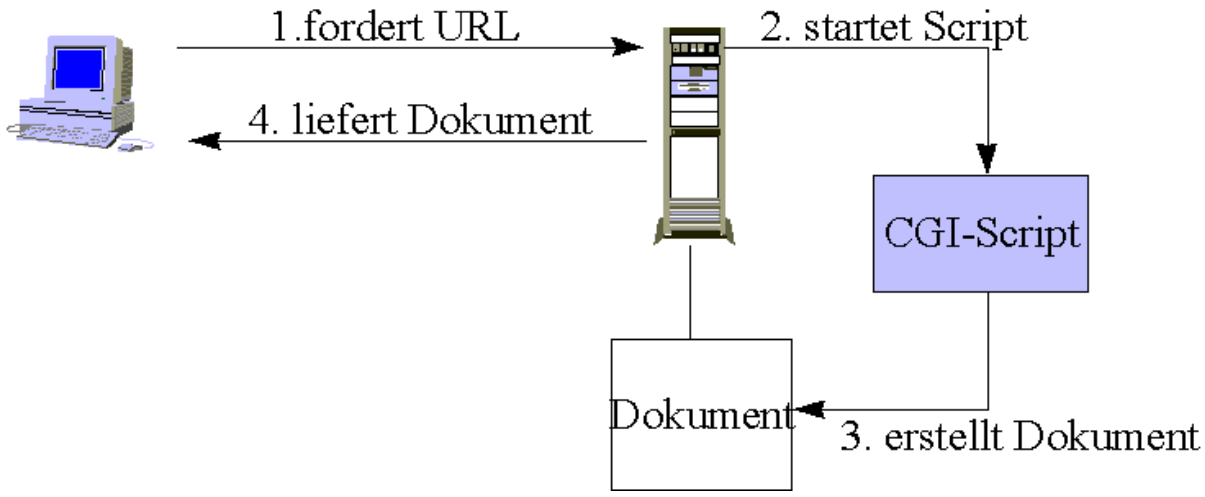
- Akzeptieren von Verzeichnisnamen, d.h., Requests die in der URL einen Verzeichnisnamen aufweisen werden zugelassen

- Verwenden von standardmässigen Dateinamen, d.h., es wird im akzeptierten Verzeichnis nach Standaddateinamen wie z.B. index.html oder welcome.html gesucht und der Inhalt als Antwort geschickt
- Senden von Verzeichnislisten als Response (wenn die URL zwar stimmt, aber keine Standardnamen zu finden sind – sehr unsicher!)

Solange die Antwort eine gültige HTTP-Nachricht ist, welche die vom Client erwarteten Informationen (wie etwa Medientyp und Datumsangaben) enthält, ist es dem Server freigestellt, wie er Requests interpretiert.
Wenn allerdings die Anfrage einen Request nach einem CGI-Skript (**C**ommon **G**ateway **I**nterface) enthält, muss ein **externes Programm** gestartet werden. (siehe weiter unten)
Der URL-Pfad eines solchen Requests kann z.B. im Web-Verzeichnis unter /cgi-bin/scriptname liegen. Dies hat den Sinn und Zweck darin, da externe Programme das grösste Sicherheitsproblem für den Web-Server darstellen und desshalb diese in einem dafür vorgesehenem Verzeichniss stehen sollten, zu welchem nur der Administrator Schreibzugriff hat:

Common Gateway Interface (CGI)

Abb.2:

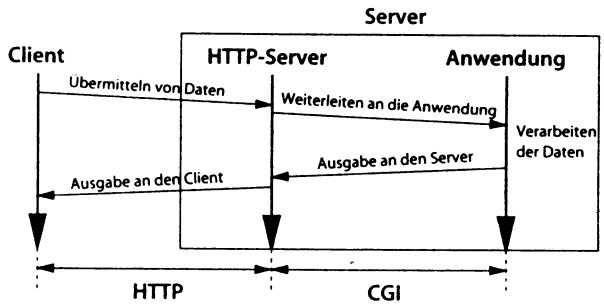


Requestbearbeitung mittels CGI-Script

Um Aufgaben besser verteilen zu können, ist es sinnvoll, einen externen Prozess zu erzeugen und einen Mechanismus einzusetzen, mit dem der Server den Prozess (und die Ein- und Ausgabe der Daten) initiieren kann. Dazu wird das CGI verwendet. Es ist als rein lokale und sprachneutrale Schnittstelle definiert, die Standardmethoden zur Kommunikation zwischen Prozessen verwendet. Somit können nun Informations-Server und externes Programm miteinander in Verbindung treten, und zwar plattformübergreifend.

Dies geschieht folgendermassen:

Abb.3:



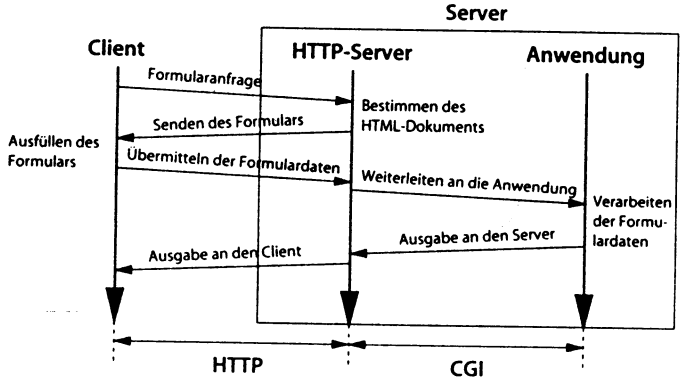
Das CGI-Script zwischen Server und Client

ein Request für eine Ressource, welche durch ein CGI-Script dargestellt wird, wird empfangen, und der Server setzt eine Reihe von Umgebungsvariablen wie z.B. SERVER_PORT, SERVER_SOFTWARE und SERVER_NAME ...u.s.w.

- Der Server startet das CGI-Skript, möglicherweise mit Parametern wie z.B. <ISINDEX> wenn es Beschränkungen gegenüber dem Betriebssystem gibt. Dies wird dann in die QUERY_STRING Umgebungsvariable im Header eingesetzt. Wenn im Header noch andere Informationen enthalten sind, wie z.B. HTTP-POST oder -PUT-Requests, werden diese noch an das Script übergeben.
- Danach übergibt der Server die Daten über die Standardeingabe an das Script, und der Server sendet so viele Bytes,, wie in der Umgebungsvariable CONTENT_LENGHT festgelegt ist. Wenn im Request der Inhaltstyp der Daten angegeben war, setzt der Server die Umgebungsvariable CONTENT_TYPE auf den MIME-Typ der Daten, die über die Sandardeingabe an das Skript gesendet werden sollen.
- Nach dem Empfang der Eingabedaten aus den Umgebungsvariablen, der Standardeingabe und evtl. der Befehlszeilenparameter verarbeitet das CGI-Skriptdie Eingaben und erzeugt eine an den Client zu sendende Ausgabe mit passensem Header:
- Hier kann man nun zwischen Parsed Header und No-Parsed Header unterscheiden:
  Beim Parsed Header übergibt das Skript einige Informationen an den Server und dieser erzeugt einen gültigen Header den er dann in die Response einbaut; danach folgen dann die Daten.
  Beim No-Parsed Header soll der Server die Ausgabe nicht zerlegen. Hier ist das CGI-Skript alleine für die Erzeugung eines gültigen Headers verantwortlich.

Die häufigste Anwendung von CGI ist die Verarbeitung von Formularen.

Abb.4:



Formularbearbeitung mittels CGI-Script

Dies geschieht mit Hilfe der Methoden in der Request-Line. Die wichtigsten sind PUT und GET (wie schon im Vortrag „HTTP“ beschrieben)
GET ruft das zugehörige Skript auf, welches es aus dem Abfrage-String bestimmt (in der Umgebungsvariable QUERY_STRING)
POST erzeugt dann logischerweise den POST-HTTP-Request, die Formulardaten werden in den Request-Body eingebunden. Der Server kann dann das passende Skript bestimmen und aufrufen und der Request-Body wird auf die Standardausgabe des Skript geschrieben.
Als konkretes Beispiel kann hier der Begriff der „**Suchmaschine**“ genannt werden: (siehe Abb.4: Der Benutzer ruft eine entsprechende Seite auf („Formularabfrage“), der Server schickt sie („Senden des Formulars“), der Benutzer füllt das Formular aus und schickt es wieder zurück („Übermitteln der Formulardaten“). Daraufhin bearbeitet der Server den Request, indem er die Formulardaten z.B. an eine Datenbank weiterleitet („Weiterleiten an die Anwendung“ und „Verarbeiten der Formulardaten“). Zum Schluss gibt die Anwendung eine Ausgabe an den Server zurück - in diesem Fall das Ergebnis der Suche – und dieser übermittelt die Daten an den Client („Ausgabe an den Server“ und „Ausgabe an den Client“)

Schlussbemerkung

Man sieht: Ein gutes CGI-Script bedeutet – neben den „Links“ – die einzige Interaktiviät im „Netz“! Und: Interaktivität macht ja das Internet zu so einem interessanten Medium! Die richtige Konfiguration eines HTTP-Servers ist schwierig, jedoch lässt sie, wenn sie gut auf die Bedürfnisse abgestimmt ist, eine hohe Performance zu. Je besser die Struktur und Automatisierung, desto schneller und direkter findet die Response ihren Weg zurück zum Client. Wie oft beschwert man sich über die Langsamkeit des Internets? – Oft liegt dies nur schon an der „letzten Meile“ bzw. der Verbindung zwischen User und Server! Wichtig ist vor allem, dass der Server nicht mit unnötigen Request/Response Aufrufen das ohnehin schon strapazierte Web zu stark belastet, und alle Prozesse und Operationen möglichst intern bearbeitet, denn : Ein System ist immer nur so schnell, wie seine langsamste Komponente!

Referenzen und Literaturverzeichnis

Ref.[1]: PPS Grundlagen des Internets, Vortrag „HTTP“ von Marco Somaini

E. Wilde: World WIDE Web – Technische Grundlagen; Springer Verlag, Berlin,Deutschland 1999

Diverse Webseiten zum Thema „ Konfiguration von Web-Servern

IP-Telefonie

Vortrag 11, 20. Juni 2000

Telefonieren über TCP/IP: Spielerei oder Zukunftstechnik?

Christoph Hunziker

Einführung

1996 schien die IP-Telefonie der nächste grosse Schub für das Internet zu sein. Der damalige Netscape-Chef Jim Clark sprach sogar von der Abschaffung des klassischen Telefons. Nur gerade vor zwei Jahren hiess es: Die Experten diskutieren heute nicht mehr darüber, ob es sinnvoll ist Sprache durch das Internet zu leiten, sondern welcher Prozentsatz des Telefonverkehrs demnächst auf diesem Weg laufen wird. Andere prophezeiten die völlige Verschmelzung von Telefonnetz und Internet.

Der klare Vorteil der Internet-Telefonie liegt natürlich bei den Gesprächskosten. Jedes Gespräch, sei es in der gleichen Stadt oder zum nächsten Kontinenten wäre zum Ortstarif zu haben. Doch es gibt auch schwerwiegende Nachteile gegenüber dem normalen Telefonieren, die in der ersten Euphorie unterschätzt wurden:

- Der Gesprächspartner muss ebenfalls online sein
- Man muss die IP-Adresse des jeweiligen Gesprächspartners wissen
- Für jedes Telefon müsste zuerst der eigene PC-hochgefahren werden
- Beim Gesprächspartner fallen ebenfalls kosten (zum Ortstarif) an
- Die Sprachqualität war zu Beginn miserabel
- Die ersten Softwareangebote arbeiteten nur im Halbduplex-Modus

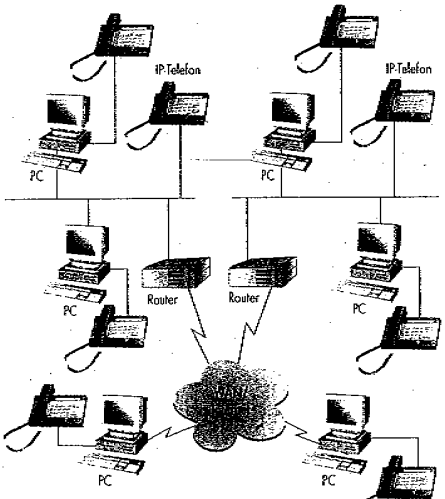
Diese Nachteile genügten, um die Internet-Telefonie vorerst auf eine Nische von Online-Spielern und Technikfreaks zu beschränken.

Inzwischen hat man sich der Technik jedoch wieder angenommen. Unter dem Oberbegriff Voice over IP (VoIP) versuchen sich Telecom-Carrier und Internet-Provider neue Geschäftsfelder zu erschliessen.

Heutige Anwendungen von VoIP

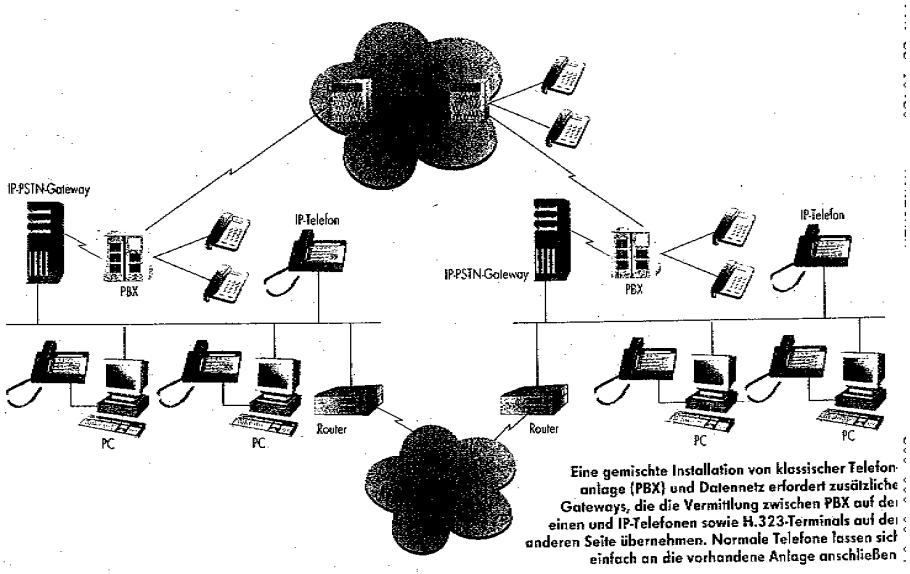
Für den Normalanwender sind Voice-over-IP-Techniken momentan nur dann interessant, wenn er eine bestehende Internet-Verbindung gleichzeitig für einen Voice-Chat benutzen will. Diese Gruppe ist damit also nach wie vor ziemlich eingeschränkt. Es gibt aber zwei andere Kundengruppen, die ein massives wirtschaftliches Interesse an VoIP haben.

Zum einen sind das grössere Unternehmen für die diese Technik in vielen Fällen Kostenvorteile bietet. Die Vereinheitlichung der internen Netze verbilligt die Administration derselben erheblich und meistens bringt die Anschaffung eines Netzes weniger Kosten mit sich als die Installation von zwei parallelen Netzen. Ausserdem ist eine Erweiterung des Netzes so viel einfacher. In diesem Bereich sehen die Hersteller von integrierten Routern (wie z.B:. Siemens, 3Com, Cisco oder Lucent), die sowohl Sprache als auch Datenverkehr weiterleiten ihren grössten Markt. (Fig.1) zeigt eine durch VoIP erweiterte Telefonanlage. Auf (Fig.2) ist die parallele Installation von Telefonanlage und Internet gezeigt.



Ein existierendes LAN läßt sich durch Ausbau der vorhandenen Router um eine Telefonanlage erweitern – die Sprachübertragung erfolgt dann komplett mit Voice over IP, normale Telefone sind in diesem Fall nicht einsetzbar. Auch kann man nicht mit Teilnehmern im normalen Telefonnetz sprechen, solange der Internet-Provider kein entsprechendes Gateway anbietet.

Fig.1: IP-Telefonanlage



Eine gemischte Installation von klassischer Telefonanlage (PBX) und Datennetz erfordert zusätzliche Gateways, die die Vermittlung zwischen PBX auf der einen und IP-Telefonen sowie H.323-Terminals auf der anderen Seite übernehmen. Normale Telefone lassen sich einfach an die vorhandene Anlage anschließen

Fig.2: parallele, umständliche Installation von Internet und analogem Telefon

Zum anderen haben auch sogenannte Call Center ein grosses Interesse an VoIP. Dies sind Dienstleistungsbetriebe, die entweder eigenständig oder innerhalb einer Firma als Zentrum für Marketing und Support agieren. Sie profitieren natürlich auch von einer einfacheren Verwaltung und Ausbaumöglichkeit, andererseits ergeben sich durch VoIP für sie ganz neue Serviceangebote. Diese Online-Angebote könnten mittels VoIP aber auch durch Dienste wie Click to Dial oder Click to Fax ausgebaut werden. Ein Anwender der Support benötigt könnte durch einen Link im Web gleich ein Telefongespräch aufbauen lassen.

Funktionsweise

Die Technik stellt der breiten Einführung von VoIP eigentlich nicht im Wege, auch wenn bestimmte Bereiche momentan noch nicht optimal gelöst sind. Verbesserungen sind zum Beispiel in den Codecs (Codierer und Decodierer) noch möglich. Diese haben die Aufgabe eine bessere Komprimierung der Sprachpakete bei einer steigenden Qualität der Sprachübertragung zu ermöglichen.
Das grundlegende Problem beim Telefonieren über TCP/IP sind aber die verschiedenen Ansätze die das Telefonnetz und die VoIP-Technik verwendet. Datennetze arbeiten im Gegensatz zum analogen Telefonnetz paketorientiert. Somit ist die logische, und zeitliche Abfolge der Pakete, wie sie beim Empfänger ankommen grundsätzlich beliebig. Daher gestaltet sich die zeitliche Synchronisation eher schwierig.
Daneben unterscheidet sich der Adressierungsmechanismus in Daten- und Sprachnetzen ebenfalls. Um eine Verbindung zu ermöglichen braucht es eine Umsetzung von der Adressierung des herkömmlichen Telefonsystems und der in Datennetzen verwendeten IP-Adressierung. Dafür gibt es seit 1996 einen standardisierten Rahmen, der die Übertragung von Sprache in Datennetzen beschreibt, die H.323-Norm.

Die H.323-Norm

Die H.323-Norm umfasst eine Vielzahl an verschiedenen Protokollen, die unterschiedliche Schnittstellen zur Sprach-, Video-, und Datenübertragung bieten. Nur so ist es möglich, dass PC's mit verschiedenen Ausrüstungen miteinander Kommunizieren können. Der Protokollumfang ist in (Fig.3) gezeigt.

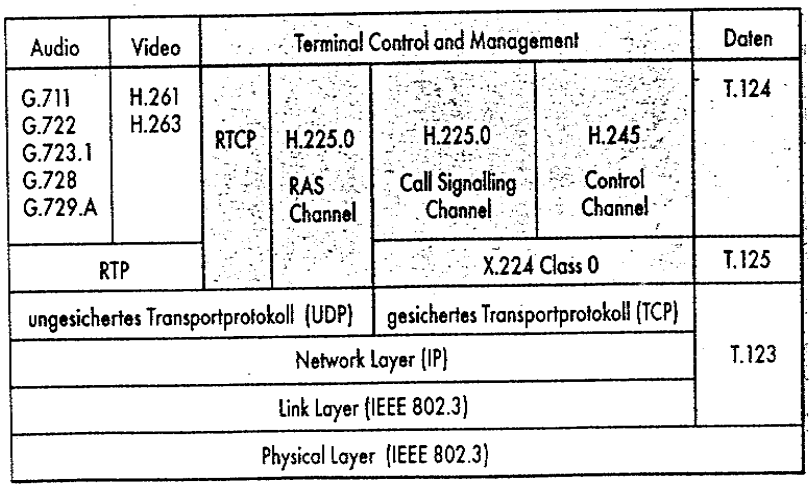


Fig.3: Die H.323-Norm

Um die Funktionsweise des VoIP näher zu verstehen, beschreibe ich hier den ungefähren Ablauf eines Anrufs:

Der **Anrufaufbau** kann in drei Phasen aufgeteilt werden:

- **RAS/ H.225:** Das H.323 Terminal sendet eine Message zum Gatekeeper, mit Name und Telefonnummer der anzurufenden Person. In dieser Phase erledigt der Gatekeeper drei Funktionen. Er übersetzt die Telefonnummer in die IP-Adresse und überprüft und steuert die Verbindung.

- **Q.931:** Über das TCP wird nun die direkte Verbindung hergestellt.

- **H.245:** In dieser Phase machen die Endgeräte aus welche Dienste sie unterstützen (Audio, Video oder Daten), wobei dann die Verschiedenen Codecs (G.711 bis G.729) zum Einsatz kommen.

Bei der Nachfolgenden Kommunikation wird das Real-Time-Protocol (**RTP** und **RTCP**) verwendet. RTP sorgt für eine Synchronisation zwischen Sender und Empfänger indem es Zeit- und Synchronisationsinformationen in die IP-Pakete einfügt. Für den Transfer setzt H.323 die ungesicherte Datenübertragung UDP ein. Eine Fehlerkorrektur hätte nur eine Verzögerung und eine Verschlechterung der Sprachqualität zur Folge.

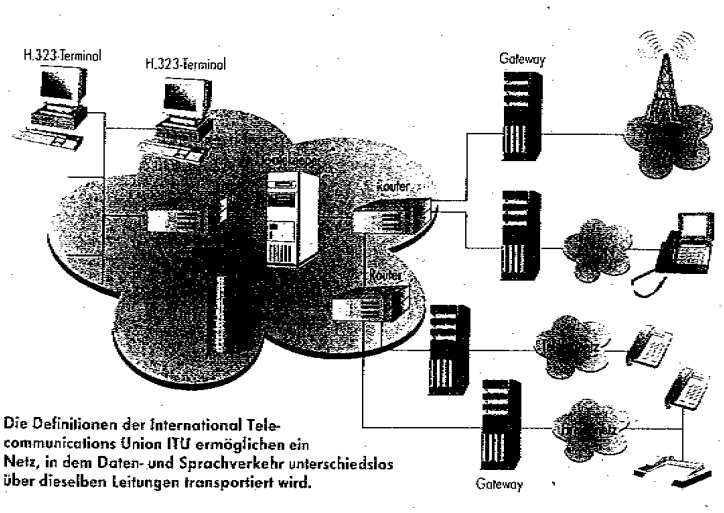


Fig.4: Daten- und Sprachnetz

Unter dem H.323 Standard ist es grundsätzlich möglich drei verschiedene Objekte im Netzwerk anzurufen. Wie in (Fig.4) gezeigt sind das entweder normale H.323-Terminals (PC's oder IP-Telefone), sogenannte Multipoint Control Units (MCU's), die Telefonkonferenzen ermöglichen, oder dann Gateways, die als Schnittstelle zu verschiedenen anderen Netzen dienen. Der Gatekeeper hat dabei die eigentliche Aufgabe einer Telefonanlage im IP-Netz. Soll nur innerhalb eines Netzes telefoniert werden reicht ein Gatekeeper schon aus. Eine Verbindung zum normalen Telefonnetz ist damit aber noch nicht möglich, dazu sind zusätzliche Gateways notwendig. Endgeräte in solchen VoIP-Netzen bezeichnet man als H.323 Terminals. Dies ist aber nur ein Oberbegriff für alle Geräte, die Sprachübertragung über ein Datennetz ermöglichen. Es kann sich also etwa um Telefone mit Ethernet-Anschluss oder PC's mit entsprechender Softwareausstattung handeln. Die Umsetzung von Sprache in Datenpakete wird ebenfalls vom Gatekeeper übernommen. Ein separater

Rechner ist dafür aber nicht unbedingt notwendig. Der GK kann ebensogut im Terminal selber implementiert sein oder eine Software übernimmt seine Aufgaben (Microsoft NetMeeting ist dafür ein gutes Beispiel). Während der GK immer im IP-Netz der H.323 Terminals steht ist dies für das Gateway zum Telefonnetz anders. Im Prinzip kann es überall da eingerichtet werden, wo ein Übergang zwischen Leitungen des IP- und des Telefonnetzes möglich ist. Für beste Sprachqualität ist natürlich eine Position möglichst nah an den H.323-Terminals sinnvoll, da dadurch die Auswirkungen des mässigen Echtzeitverhaltens von IP-Netzen klein gehalten werden. Allerdings ist dann die Strecke, die über normale Telefonleitungen zurückgelegt wird entsprechend hoch. Um Kosten zu sparen sollte aber gerade im Gegenteil eine möglichst grosse Strecke über das Internet zurückgelegt werden.

Ausblick in die Zukunft

Angesichts der diversen Gruppen, die ein Interesse an der Durchsetzung von Voice over IP haben, und angesichts der weitgehend vorhandenen Standardisierung und zum Einsatz bereitstehenden Techniken ist es nur noch eine Frage der Zeit, bis diese Technik allgemein verfügbar ist und zu einem einheitlichen Netz führt. Momentan allerdings hat Voice over IP für den Endanwender noch kaum eine Bedeutung - ausser in sehr eingeschränkten Bereichen und bei Technikfreaks.

Die VoIP-Techniken dienen aber als Werkzeug für Firmen und Dienstanbieter und helfen mit neue Geschäftsfelder zu erschliessen und die internen Strukturen zu vereinfachen. Wenn sich VoIP dann einmal auf breiter Basis etabliert hat, dürfte es auch nicht unbedingt mehr billigere Verbindungen ermöglichen, denn momentan sind Bestrebungen im Gange zusätzliche Abrechnungsmechanismen einzuführen, wenn das herkömmliche Telefonnetz und IP-Netze zusammengeschlossen werden.

Quellenangaben

• Axel Kossel	Netzgespräche	Reportage c’t 1999 Heft 10
• Jürgen Kuri	Sprache in Päckchen	Reportage c’t 1999 Heft 10
• Rizzetto/ C. Catania	A voice over IP service architecture	Internet Computing June 1999

Drahtlose Kommunikation

—

WAP

Philippe Hefti
Vortrag 12 / 20.6.2000
PPS Seminar: Grundlage der Internet-Technologie

1. Einführung

Sogenannte Digital Wireless User Agents (z.B. Handy) gewinnen in den letzten Jahren immer mehr an Bedeutung und stellen einen der aussichtsreichsten und expansionsfreudigsten Bereiche in der Telekommunikations- und Informationstechnologie-Branche dar. Bereits jetzt gibt es mehr Mobilsysteme als PCs (nach [2]). Der Trend geht dabei weg vom reinen Telefon, in Richtung mobiler und drahtloser Geräte für Applikationen jeglicher Art: Spiele, Kalender, Adressbücher und weitere Anwendungen gehören schon jetzt zur Standardausstattung vieler Handys. Die Handys werden auch immer mehr als Kommunikationshilfe zwischen Laptops oder PDAs (Personal Digital Assistent) und dem World Wide Web verwendet.

Was liegt also näher, als eine direkte Verbindung zwischen den Digital Wireless User Agents und dem Internet zu ermöglichen und die abgerufenen Informationen ohne Verwendung weiterer Geräte auf deren Display darzustellen. Die Übertragung von herkömmlichen HTML-Seiten auf ein Handy ist allerdings aufgrund folgender Punkte nur begrenzt sinnvoll:

1. stark begrenzter Speicher mobiler Geräte
2. zu kleines Display (erlaubt nur Darstellung von vier bis fünf Zeilen Text), meist schwarz-weiss
3. zu niedrige Übertragungsrate (9,6 kbit/s) für die Darstellung umfangreicher Seiten
4. die meisten Seiten basieren auf Grafiken, die auf einem kleinen Display nicht dargestellt werden können.

Es musste also ein Standard geschaffen werden, der Internetseiten Handytauglich macht und mit dessen Hilfe der Aufbau HTML-ähnlicher Seiten implementiert werden kann.

Das Ergebnis ist das **WAP** *(Wireless Application Protocol)* mit einer neuen, auf XML basierenden Sprache names **WML** *(Wireless Markup Language)* sowie der zugehörigen Scriptsprache **WMLScript** (beide Bestandteil des Protokolls WAP)

Die Idee hinter WAP ist es demnach, Internetangebote für Handys zugänglich zu machen, d.h. eine direkte Verbindung zw. den Handys und dem Internet, um die abgerufenen Informationen auf deren Display darzustellen.

WAP wurde von den grossen Handyherstellern Nokia, Ericsson, Motorola sowie dem Software Unternehmen Unwired Planet (mittlerweile Phone.com) 1997 initiert. Diese Firmen haben das WAP-Forum gegründet, welches bis heute über 90 Firmen umfasst (u.a auch Microsoft).

WAP ist ein offener Standard und sowohl zum US-Standard CDMA (Code Division Multiple Access), als auch zum europäischen GSM (Global System for Mobile Communications) kompatibel.

Das erste WAP-Handys, welches Ende 1999 auf den Markt kam, ist das Nokia 7150 (Abb. 1).

Abb. 1: Nokia 7150

2. WAP-Request

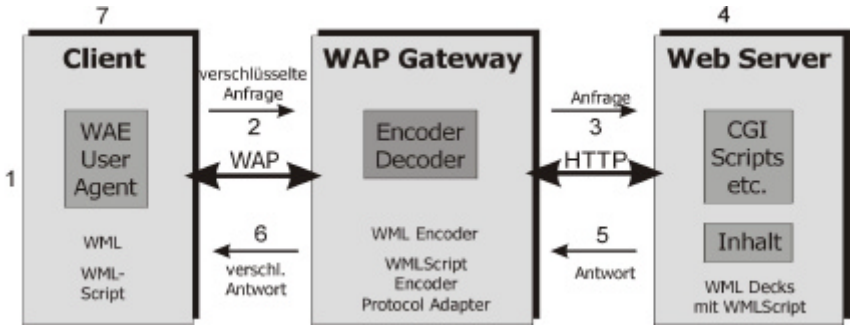


Abb. 2: WAP-Request

Wie in der Abbildung skizziert, verläuft ein typischer WAP-Request nach folgendem Schema:

1. Die Eingabe einer URL, die zum Beispiel als Link im Browser hinterlegt sein kann, erzeugt innerhalb des Client einen Request.
2. Der Request wird mit Hilfe des WAP-Protokolls an ein WAP-Gateway übertragen.
3. Das WAP-Gateway wiederum transformiert den Request in einen herkömmlichen HTTP-Request und leitet diesen an den entsprechenden Web-Server weiter.
4. Der Web-Server bearbeitet den HTTP-Request wie gewohnt und gibt ein sogenanntes WML-Deck mit zusätzlichem HTTP-Header an das WAP-Gateway zurück.
5. Innerhalb des WAP-Gateway erfolgt die Verifizierung des WML-Decks sowie des HTTP-Header und anschliessend die Kodierung in das binäre WML-Format.
6. Abschliessend erzeugt das WAP-Gateway eine WAP-Response und sendet diese an den Client.
7. Der Client empfängt die WAP-Response, arbeitet das binäre WML ab und stellt die erste Card des WML-Decks dar.

Liefert der Web-Server anstelle von WAP-Inhalten – in Form von WML oder WMLScript – 'normale' WWW-Inhalte, zum Beispiel HTML-Seiten, so kann ein HTML-Filter innerhalb des WAP-Gateway für die mehr oder weniger automatische Transformation der Inhalte genutzt werden. In der Praxis handelt es sich allerdings bei diesem technisch durchaus akzeptablen Ansatz nicht wirklich um eine sinnvolle Alternative zu den, speziell auf die Besonderheiten der Mobile Devices abgestimmten, WML-Dokumenten.

3. WAP-Architektur

Die WAP-Architektur basiert auf einem schichtenförmigen Modell, wie man es auch von anderen Netzwerkprotokollfamilien wie beispielsweise TCP/IP her kennt. WAP wird durch fünf Schichten beschrieben: Anwendungs-, Session-, Transaktions-, Sicherungs- und Transportschicht. In jeder dieser Schichten (Abbildung 3) kommen Anwendungen und Protokolle gleichermaßen zum Einsatz.

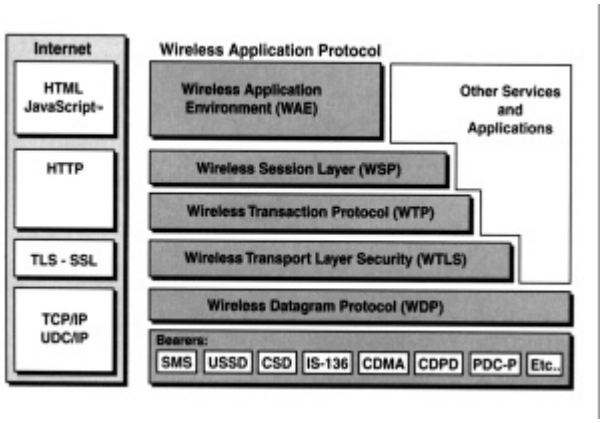


Abb. 3: WAP-Modell

Komponenten der WAP-Architektur

1. **Anwendungsschicht:** Hier findet man das Wireless Application Environment (WAE), das als Anwendungsumgebung auf WWW- und Telefonietechnologien basiert und in erster Linie als Ausführungsumgebung von WAP-Anwendungen dient. WAE unterstützt insbesondere Wireless Markup Language (WML), WML-Script und Wireless Telephony Applications (WTA).
    a. **Wireless Markup Language (WML):** WML basiert auf XML und bietet Möglichkeiten zur Darstellung von Text und Grafiken. In einigen Belangen wurde WML aber an die unterschiedliche Umgebung angepasst. Die Identifikation von Webseiten erfolgt auch hier über eine eindeutige URL, danach folgt aber aufgrund der kleineren Displays noch eine zusätzliche Unterteilung der HTML- Seiten in mehrere Cards. Diese enthalten zwischen vier und zehn Zeilen mit je acht bis zwölf Zeichen. Die Navigation durch diese Seiten ähnelt also dem Lesen einer langen SMS.

    b. **Scriptsprache WMLScript:** Diese Programmiersprache basiert auf ECMAScript (European Computer Manufactures Association) und ist damit ähnlich aufgebaut wie JavaScript. WMLScript interpretiert Bytecode und ist direkt ausführbar auf den Endgeräten. Dies spart Zeit und Speicherressourcen.

    c. **Wireless Telephony Application (WTA):** WTA dient dem Zugriff auf erweiterte Telefoniedienste. Hierüber wird ein gesicherter Zugriff auf die Telephony API des Endgeräts bereitgestellt, der beispielsweise eine differenzierte Kostenkontrolle ermöglicht.

2. **Session-Schicht:** In dieser Schicht sorgt das Wireless Session Protocol (WSP) für die Bereitstellung von zwei Diensten. Es handelt sich zum einen um einen verbindungsorientierten Service, der oberhalb von Wireless Transaction Protocol (WTP) operiert, zum anderen um einen verbindungslosen Service, der als Datagramm-Service agiert.

3. **Transaktionsschicht:** Hier sorgt das neue Wireless Transaction Protocol (WTP) für die Ausführung von als »zuverlässig« und als »unzuverlässig« deklarierten Transaktionen.

4. **Sicherungsschicht:** Sie dient der Sicherung der Datenintegrität, Privatsphäre und Authentifizierung. Außerdem bietet diese Schicht Schutz vor Denial-of-Service-Attacken. Kernstück ist die Funktion Wireless Transport Layer Security (WTLS), die technisch auf dem SSL-Nachfolger TLS basiert.

5. **Transportschicht:** Als allgemeiner Transportmechanismus ist das Wireless Datagram Protocol (WDP) für die Kommunikation zwischen dem Bearer und den darüber liegenden Schichten zuständig. Der Ausdruck "Bearer" bezeichnet somit Schnittstellen zwischen WAP und physikalischen Netzen wie GSM- oder TCP/IP-Netzen).

Wie die WAP-Seiten auf den Endgeräten dargestellt werden, definiert die Spezifikation von WAP nicht. Dies ist den Herstellern der Endgeräte überlassen. Durch verschiedene Anpassungen, wie z.B. die Übersetzung vom HTTP-Header von normalem Text in Binärcode, wird der Umfang einer WAP-Verbindung im Vergleich zu TCP auf weniger als die Hälfte der Kommunikationsschritte reduziert.

4. WML-Programmierung

Die Programmierung von WML Dokumenten gestaltet sich nach einiger Einarbeitungszeit genauso einfach wie die von HTML-Seiten. Grundvoraussetzung für gelungene WML-Anwendungen ist allerdings neben der reinen Beherrschung der WML-Syntax ein gutes softwareergonomisches Verständnis für die Besonderheiten der mobilen Geräte und ihrer Displays.
In der Regel gelten für diese Art von Geräten - im Vergleich zu PCs – folgende Restriktionen:

1. weniger leistungsfähige CPUs
2. deutlich weniger RAM und ROM
3. kleinere Displays mit geringerer Farbtiefe (in der Regel 1Bit)
4. weniger komfortable Eingabemöglichkeiten

bei der Erstellung von WML-Dokumenten muss ausserdem beachtet werden:

5. geringere Stabilität der Verbindung
6. erhöhte Warte- und Verzögerungszeiten
7. geringere Bandbreite

WML-Struktur:

Basiselement innerhalb des WML-Syntax ist die sogenannte Card. Mehrere dieser Cards lassen sich wiederum in einem Deck zusammenfassen, wobei ein Deck als eine logische Einheit angesehen werden kann und das oberste Element eines WML-Dokumentes darstellt (siehe Abbildung). Empfängt das WAP-Gerät ein solches Deck, zeigt er immer die erste Card des Decks an.

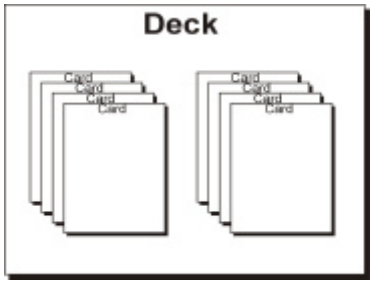


Abb. 4: Cards in Decks

Ein Beispiel zu WML:
Das folgende Beispiel zeigt ein einfaches WML-Deck, das eine einzelne Card enthält:

```
1  <?xml version='1.0'?>
2  <'DOCTYPE wml PUBLIC '-//WAPFORUM//DTD WML 1.1//EN'
          'http://www.wapforum.org/DTD/wml_1.1.xml'>
3  <wml>
4    <card id='First_Card' title='First Card'>
5         <p>
6         The first WML example
7         </p>
8    </card>
9  </wml>
```

1. Die erste Linie ist eine Standard XML-Anweisung, welche von allen XML Dokumenten verlangt wird.
2. Die zweite Linie bezeichnet die XML Dokument Type Definition, welche auch von allen XML-Dokumenten benötigt wird, die wie WML externe Dokument-Typen benützen.
3. Ein WML-Deck ist durch das wml-Tag definiert. Sämtliche Informationen und Cards innerhalb eines Decks werden somit durch <wml> und </wml> begrenzt.
4. Die folgenden Linien definieren eine Card, welche einen Start-Tag, einen End-Tag und den Text, der angezeigt werden soll, beinhaltet.

Nach dem Laden des Decks, sollte es wie in der Abbildung angezeigt werden:

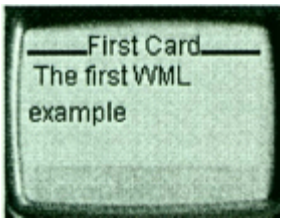


Abb. 5: WML-Beispiel

Die WML-Spezifikation definiert lediglich eine Syntax zur Implementierung von WML-Dokumenten. Die Art der Darstellung dagegen bleibt zum grössten Teil den Browsern überlassen. Dies führt, besonders am Anfang einer neuen Spezifikation zu inkonsistenten Darstellungen. Diese Problematik gilt für die Darstellung innerhalb eines Digital Wireless User Agents um so mehr, da die Unterschiede in Ausmass und Fabrtiefe der Displays der verschiedenen heute am Markt existierenden und für die Zukunft geplanten WAP-Devices deutlich extremer sind, als es bei WWW-Browsern je der Fall war.

Die Hersteller von WAP-Devices haben deswegen für ihre Geräte sogenannte Design Guidelines für die Erstellung von WAP-Seiten veröffentlicht, um den Entwicklern eine Möglichkeit zu geben, die Besonderheiten der Geräte kennen zu lernen und bei der Umsetzung von WML-Dokumenten zu berücksichtigen.

5. Zusammenfassung und Schlussfolgerung

WAP steht erst am Anfang der Entwicklung. In der Praxis gibt es noch viele (Anfangs)-Probleme, z.B. mit der Programmierung von WML-Seiten, die gelöst werden müssen.

Man ist zum Teil auch der Ansicht, dass den echten Durchbruch von WAP erst der schnelle zukünftige Standard UMTS bringen wird. UMTS steht für das Universal Mobile Telefone System, welches zwischen 2002 und 2003 eingeführt wird. Für diesen Umstieg zu UMTS wird es aber auch neue Frequenzen, neue Anbieter, neue Infrastrukturen und auch neue Telefone brauchen. Bei den zukünftigen Handys der 3.Generation werden aber viel grössere Übertragungsraten zur Verfügung stehen, so dass auch Bilder und Videodaten empfangen werden können. Die WAP-Telefone und die WAP-Angebote, die heute auf den Markt kommen, sind demnach typische Übergangslösungen, für die passend nach und nach höhere Geschwindigkeiten und Dienste angeboten werden. Sie sind daher für die Koexistenz mit UMTS eingerichtet, doch keine Geräte der dritten Mobilfunkgeneration.

Abb. 6: UMTS Mobiltelefone (Quelle: Nokia [4])

Quellenangaben:

[1] Andreas Hitzig: *Drahtlos surfen mit Volldampf*; iX 10/99, S. 128 - 132

[2] Lars Röwekamp: *Handy HTML*; iX 2/2000, S. 52 - 57

[3] Wireless Application Forum: *WAP: Wireless Application Protocol*; White Paper, October 1999

[4] Nokia Wap Developer Pages: www.forum.nokia.com/developers/wap/wap.html

[5] diverse Internet-Seiten