

1 Web-Grundlagen

Die Anfänge des *World Wide Web* liegen etwas mehr als 10 Jahre zurück. 1990 verknüpfte Tim Berners-Lee zwei Technologien zu einer neuen Anwendung. Die dem WWW zugrunde liegenden Technologien Hypertext und Internet können auf eine lange Entwicklungszeit verweisen. Deren Ursprünge lagen in den Sechzigern und Siebzigern. Das World Wide Web entwickelte sich schnell zur Paradeanwendung des Internet. Das zu diesem Zeitpunkt nur in akademisch-technischen Kreisen bekannte Internet öffnete sich einem breiten Publikum. Heute bezeichnet der Begriff Internet eigentlich das World Wide Web. Im Grundsatz bleibt das WWW trotzdem nur einer von vielen Internet-Diensten.

Die Idee eines globalen Netzwerkes stammt aus den frühen sechziger Jahren. In der Folgezeit wurde mit öffentlichen Mitteln an verschiedenen amerikanischen Hochschulen und wissenschaftlichen Einrichtungen eine Netzwerk-Infrastruktur geschaffen. Die Grundidee bestand in einer offenen Architektur, die beliebig implementierte lokale Netze auf beliebigen Rechner-Plattformen verknüpfen sollte. Das *Internet* wurde auch nicht als Anwendung entwickelt. Es sollte vielmehr die Infrastruktur für Netzwerk-Anwendungen bereitstellen. Anfang der siebziger Jahre entstand mit *E-Mail* die erste Internet-Paradeanwendung. Etwa zur selben Zeit begann die Entwicklung der dem Internet noch heute zugrunde liegenden Protokoll-Familie *TCP/IP*. Die Betriebssystem-Integration von *TCP/IP* wurde Mitte der siebziger Jahre beispielhaft am eben entstandenen UNIX-Betriebssystem demonstriert. In der zweiten Hälfte der siebziger Jahre begann die Implementierung weiterer Internet-Dienste, zumeist in Form geschlossener Benutzergruppen. Die wichtigste Entwicklung stellt das *USENET* dar, der Vorläufer der heutigen Internet News Foren. Wesentliche Grundlage für die rasante Weiterentwicklung des Internets an unterschiedlichen Orten war der freie Zugang zur Dokumentation. Bereits 1969 wurden *Requests for Comments* (RFC) für die einheitliche Beschreibung neuer Technologien eingeführt. Mit der Erfindung von *FTP* hatte jeder Wissenschaftler Online-Zugang zu Beschreibungen aktueller Standards. Darauf aufbauend wurde ein zunehmend differenziertes Prozedere für die Verabschiedung von Internet-Standards etabliert, das noch heute Gültigkeit hat und dessen veröffentlichte Dokumente verbindlich sind.

Während das Internet für die Informationstechnologie bereits seit Mitte der achtziger Jahre große praktische Bedeutung hat, blieb es dem WWW vorbehalten, dieser Bedeutung eine neue Dimension zu geben. Durch einen im Vergleich zur klassischen Kommandozeile spielerischen Umgang mit der Maus am Arm hat das World Wide Web zu einer beispiellosen Kommerzialisierung des Internets geführt. Die Art und Weise, wie Individuen kommunizieren und die globalisierte Ökonomie ihre Aufgaben verteilt, wurde einem grundlegenden Wandel unterzogen. Das WWW hat enorme wirtschaftliche Potenziale freigesetzt.

Die 10 wichtigsten technischen Begriffe des Internet und World Wide Web werden im Folgenden beschrieben.

1.1 Akteure

Während der praktischen Nutzung des Internets kommunizieren grundsätzlich Clients mit Servern. Hinter den Begriffen *Client* und *Server* verbergen sich Rollenbeschreibungen. Der Browser spielt die Rolle eines Web-Clients, News-Clients, FTP-Clients oder eines Mail-Clients. Internet-Dienste arbeiten nach dem *Client/Server-Prinzip*. Um ein HTML-Dokument zu laden, nimmt der Browser Kontakt zu einem Web-Server auf. Er beauftragt den Web-Server mit der Lieferung dieser Ressource.

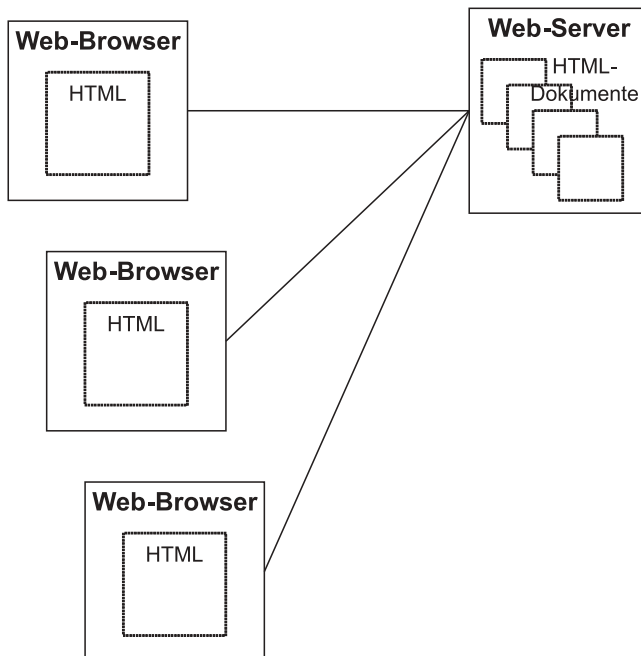


Abb. 1-1 Client/Server-Prinzip

Sowohl Client als auch Server sind ihrer Natur nach Programme oder besser Prozesse. Der Server bietet im beschriebenen Fall als Dienstleistung die Lieferung von HTML-Dokumenten an. Er wartet rund um die Uhr auf Anfragen von Clients. Kommt eine solche Anfrage, liefert der Server das gewünschte Dokument.

Die Kommunikation zwischen Client und Server wird grundsätzlich vom Client initiiert. Er generiert eine Anfrage und wartet auf die Lieferung des gewünschten Dokumentes. Nach Lieferung wird das Dokument durch den Client in geeigneter Weise präsentiert.

1.2 Infrastruktur

Die Kommunikation zwischen Client und Server erfolgt unter Nutzung der Internet-Infrastruktur. Die *TCP/IP*-Protokollfamilie stellt diese Infrastruktur bereit. Sie besteht im Wesentlichen aus den Protokollen TCP und IP.

Das Internet Protocol *IP* ist durch folgende Eigenschaften charakterisiert:

- paketorientiert
- verbindungslos
- unzuverlässig

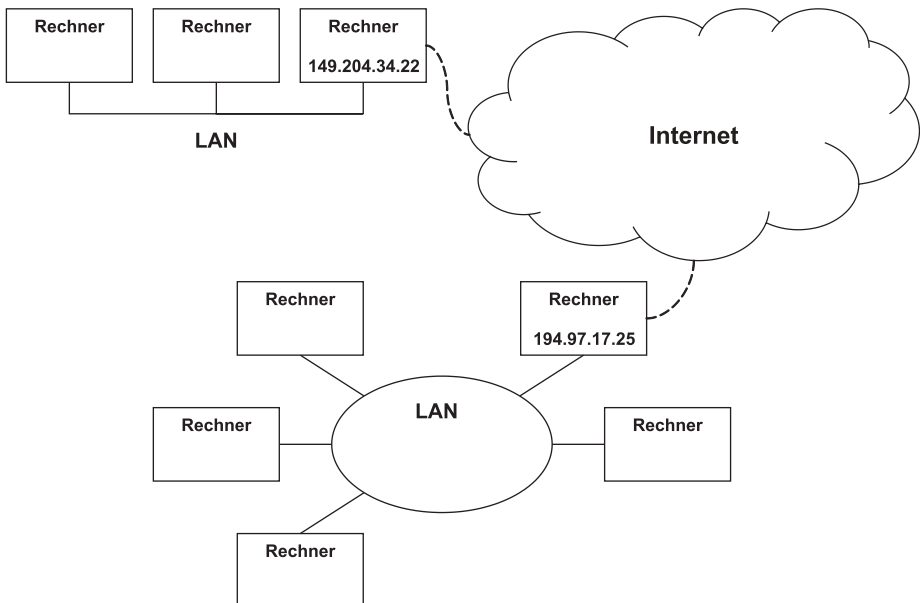


Abb. 1-2 Internet-Infrastruktur

IP stellt die Vermittlungsschicht dar. Die Hauptaufgabe besteht darin, Daten an den richtigen Rechner im richtigen physikalischen Netz weiterzuleiten. Jeder an das Internet angebundene Rechner muss dazu über eine eindeutige *IP-Adresse*

verfügen. Protokolle der höheren Schichten werden auf diese Weise davon entbunden, etwas über die physikalischen Eigenschaften des Übertragungsmediums zu wissen.

IP legt auch das Paketformat fest. Es sorgt für die Aufteilung der zu übertragenden Daten in kleine Pakete und versieht diese mit Adressen. Da keine feste Verbindung zum Adressaten besteht, wird jedes Paket für sich versendet. Aufgrund der *IP-Adresse* im Header des Paketes kann es im Netz an den richtigen Rechner weitergeleitet werden. IP verfügt über keinen Mechanismus, die Vollständigkeit und Korrektheit übertragener Daten sicherzustellen. Es gilt daher als unzuverlässig.

Das Transmission Control Protocol *TCP* stellt die Übertragungsschicht dar. Sie ist für das korrekte Zusammensetzen der Pakete beim Adressaten verantwortlich. Eine Fehlerkorrektur sorgt dafür, dass verloren gegangene Pakete erneut übertragen werden.

TCP führt darüber hinaus das Konzept der *Ports* ein. Portnummern sind kleine ganze Zahlen. Sie spezifizieren und adressieren Programme bzw. Prozesse auf dem Zielrechner, während IP den Zielrechner als Ganzes adressiert. TCP sorgt also für die Auswahl des Programms auf dem Zielrechner, das die Daten erhalten und verarbeiten soll.

194.97.17.25	
20	<input type="checkbox"/> FTP-Data
21	<input type="checkbox"/> FTP
23	<input type="checkbox"/> Telnet
25	<input type="checkbox"/> SMTP
80	<input type="checkbox"/> HTTP

Abb. 1-3 IP-Adresse und Portnummern

Wie Abbildung 1-3 zeigt, wird eine Anfrage an den Web-Server auf dem Rechner mit der IP-Adresse 194.97.17.25 durch TCP an dessen Port 80 weitergereicht. Dort wartet der Web-Server auf Anfragen. Anfragen an den Port 23 werden durch den Telnet-Server entgegengenommen und verarbeitet.

1.3 Dienste

Oberhalb von IP und TCP ist im Protokoll-Stack die Applikationsschicht angesiedelt. Als Beispiel dafür dient in Abbildung 1-4 das Hypertext Transfer Protocol HTTP. Die Applikationsschicht ist für die konkrete Syntax und Semantik der zwischen Dienstprogrammen auszutauschenden anwendungsspezifischen Daten verantwortlich. Sie nutzt dazu die darunter liegenden Schichten der Internet-Infrastruktur für die Übertragung der Daten.

Beispiele für weitere *Dienstprogramme* (und deren Protokolle) sind Telnet (Telnet), FTP (FTP-Data, FTP), E-Mail (SMTP, POP3, IMAP) und Internet News (NNTP). Jedes dieser Dienstprogramme implementiert ein genau festgelegtes anwendungsspezifisches *Protokoll*. Letzteres definiert exakt, wie Client und Server kommunizieren. Die Implementierungen solcher Protokolle legen anwendungsspezifische Befehle fest. Eigene netzwerkfähige Applikationen implementieren eigene anwendungsspezifische Protokolle.

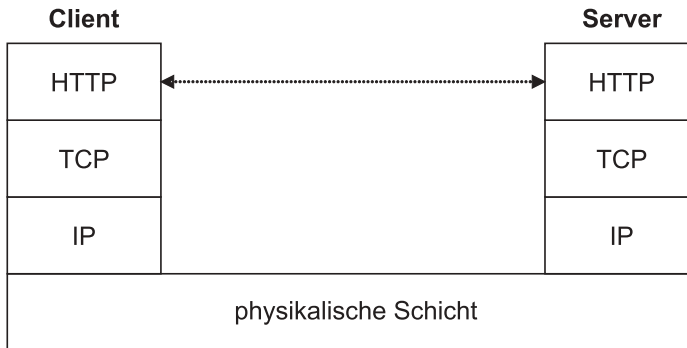


Abb. 1-4 Protokoll-Stack

Jede der beschriebenen vier Schichten stellt den von oben durchgereichten Daten einen so genannten Header mit Metadaten voran. Dieser Vorgang wird Kapselung genannt. Er ist vergleichbar mit dem Einkuvertieren eines Briefes in einen Umschlag. Die nächste Schicht packt den Umschlag in ihren Umschlag und so fort.

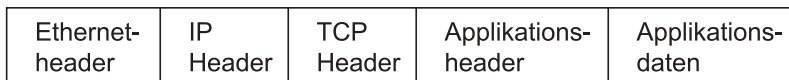


Abb. 1-5 Kapselung

Jedem traditionellen Internet-Dienst ist eine eindeutige Portnummer zugeordnet. Man spricht von *Wellknown Ports*. Vor der Kontaktaufnahme des Clients mit dem Server ist deshalb keine Vereinbarung darüber notwendig, an welchem Port der Server auf Anfragen eines Clients wartet.

Mit dem *Domain Name Service* (DNS) wurde Mitte der 80er Jahre ein für das Internet zentraler Dienst eingeführt. Rechner konnten fortan mit Namen versehen werden. DNS übersetzt diese Rechnernamen in IP-Adressen und umgekehrt. Endanwender waren bei der Adressierung eines Rechners nicht mehr gefordert, kryptische IP-Adressen anzugeben.

1.4 HTTP

HTTP steht für Hypertext Transfer Protocol und ist das Anwendungsprotokoll des *World Wide Web*. HTTP bildet die gemeinsame Sprache von Web-Client und Web-Server und eignet sich für den Austausch von Daten beliebiger Formate zwischen Client und Server. Soll ein HTML-Dokument in den Browser geladen werden, sendet dieser einen HTTP-Request an den Web-Server. Letzterer wartet laut Standard an Port 80 auf Anfragen. Für eine solche Anfrage wird zunächst eine TCP/IP-Verbindung aufgebaut. Über diese Verbindung erfolgt die Übertragung der Anfrage sowie der anschließenden Antwort des Web-Servers. Eine HTTP-Transaktion erfolgt konkret in 4 Schritten:

1. Der Client baut eine TCP-Verbindung zum Server auf.
2. Der Client sendet einen HTTP-Request zum Server.
3. Der Server schickt einen HTTP-Response zum Client.
4. Der Server baut die Verbindung wieder ab.

HTTP basiert auf dem Request/Response-Paradigma. Der Web-Server liefert nur dann einen Response, wenn der Client zuvor einen Request abgesetzt hat. Das World Wide Web ist deshalb seiner Natur nach ein Pull-Medium – der Client »zieht« aktiv die Inhalte, die er wünscht. Im Unterschied dazu existieren Protokolle, die den so genannten Server-Push unterstützen. Nach diesem Konzept verteilt der Server die Inhalte an registrierte Clients, wenn er es für notwendig erachtet. Der Server spielt die aktive Rolle. Notwendig ist das Push-Konzept beispielsweise für Online-Chats. HTTP ist für deren Implementierung ungeeignet.

Eine ausführlichere Beschreibung des HTTP-Protokolls folgt in Kapitel 10.

1.5 Inhalte

Im Rahmen der Kommunikation von Client und Server werden Daten ausgetauscht. Zur Typisierung auszutauschender Daten wurde das *MIME-System* eingeführt. Ein *MIME-Typ* stellt ein Label dar, das zusammen mit den Daten übertragen wird. So kann der Empfänger entscheiden, wie er mit den Daten abhängig von deren Typ verfahren soll. Die Notwendigkeit der Einführung von MIME-Typen hat ihre Ursache in den unterschiedlichen Dateinamenserweiterungen eines bestimmten Dateityps auf verschiedenen Betriebssystem-Plattformen.

MIME war ursprünglich eine Erweiterung des E-Mail-Protokolls und stand für Multipurpose Internet Mail Extension. Mail-Anhänge konnten so typisiert werden. Heute kommen MIME-Typen bei einer Reihe von Protokollen zum Einsatz, wenn multimediale Daten zu übertragen sind. MIME steht deshalb heute für Multimedia Internet Message Extension. Häufig wird auch einfach von Medientyp gesprochen.

Ein MIME-Typ besteht aus zwei Teilen, einem Haupttyp und einem Untertyp, die durch einen Schrägstrich getrennt sind. Beispiele dafür sind:

- text/plain, text/html
- image/gif, image/jpeg
- application/pdf, application/msword, application/octet-stream

Bei der Kommunikation mit einem Web-Server verlässt sich der Browser darauf, den richtigen MIME-Typ mitgeteilt zu bekommen. Der Web-Server verfügt über eine Tabelle für die Zuordnung von Dateinamenserweiterungen und MIME-Typen. Soll eine bestimmte Datei geliefert werden, schaut der Server in dieser Tabelle nach, um den richtigen MIME-Typ zu identifizieren und dem Browser mitzuteilen. Der Browser schaut seinerseits in einer Tabelle nach, um festzustellen, wie er mit Daten eines bestimmten Typs verfahren soll.

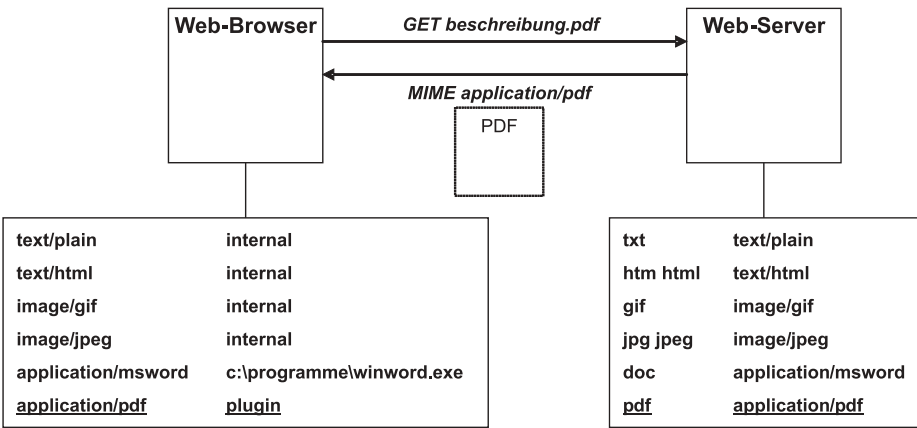


Abb. 1-6 Abbildung Dateityp auf MIME-Typ

Die standardisierten MIME-Typen sind im RFC 1521 beschrieben. Es werden sieben Hauptgruppen und eine Vielzahl von Untertypen unterschieden. Darüber hinaus ist eine Konvention für nicht standardisierte MIME-Typen festgelegt – deren Untertyp beginnt mit »x-«. Für die Benutzung solcher MIME-Typen müssen sich Browser und Server lediglich abstimmen, was insbesondere bei Intranets leicht möglich ist.

Ein Browser kann selbst nur wenige Dateitypen präsentieren. Für andere Typen ist er auf assoziierte Programme oder Programm-Komponenten angewiesen. Zu unterscheiden sind:

- externe Viewer
- Plugins

Externe Viewer, häufig auch Helper-Applikationen genannt, stellen eigenständige Programme dar. Die Konfiguration des Browsers ermöglicht deren automatisierten Start, wenn Daten eines bestimmten MIME-Typs geladen wurden. Die Daten werden dann an dieses externe Programm zum Zwecke der Präsentation überge-

ben. Jedes auf dem Client-Rechner installierte Programm kann beim Browser als externer Viewer registriert werden. Typische Beispiele für externe Viewer sind Programme der Microsoft Office Suite. In Abbildung 1–6 wird Dokumenten des MIME-Typs `application/msword` der Pfad des externen Viewers `Winword` zugeordnet.

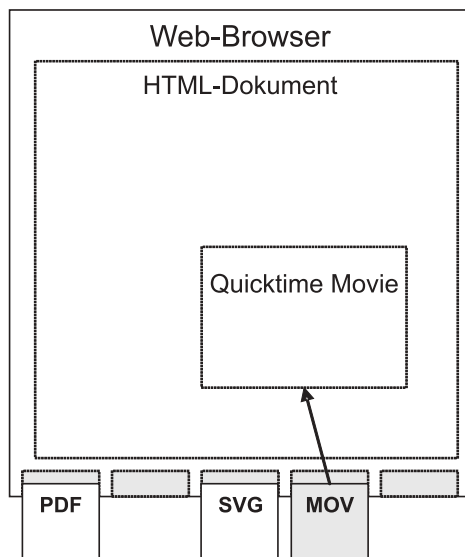


Abb. 1–7 Plugins

Plugins stellen im Gegensatz dazu Programm-Komponenten dar. Unter Windows liegen sie als DLL vor und kommunizieren mit dem Browser über eine interne Schnittstelle. Sie werden bei Bedarf geladen und präsentieren das Dokument innerhalb des Browser-Fensters. Vor allem Multimedia-Hersteller bieten Plugins für die Präsentation ihrer spezifischen Formate an. Der besondere Charme dieses Konzeptes besteht darin, dass für die Präsentation eines neuen Formates keine neue Browser-Version installiert werden muss. Der Browser kann sukzessive um die Fähigkeit erweitert werden, die gewünschten Datenformate zu präsentieren. Ein typisches Beispiel für Plugins ist der Acrobat Reader zur Präsentation von PDF-Dokumenten.

1.6 HTML-Dokumente

Das World Wide Web besteht primär aus einer Vielzahl von HTML-Dokumenten. HTML steht für Hypertext Markup Language. HTML-Seiten können mit einem gewöhnlichen Text-Editor erstellt werden. Die Darstellung erfolgt durch einen einfachen Web-Browser. Dokumente liegen häufig in Form von Dateien vor.

Hypertext bedeutet, dass ein Dokument Verweise auf andere Dokumente beinhalten kann. Solchermaßen referenzierte Dokumente können sich auf demselben

oder auf anderen Rechnern befinden. Mit Hilfe von Hyperlinks wird die logische Struktur eines Dokumentenverbundes repräsentiert. Die sich daraus ergebenden Verknüpfungen haben dem ganzen System die Bezeichnung Web gegeben.

HTML ist eine Markup-Sprache. *Markup* steht dabei für Auszeichnung und bedeutet, dass Teile eines Textes als besondere strukturelle Elemente ausgezeichnet werden können. Auszeichnungen werden dabei im Text selbst vorgenommen.

HTML beschreibt ein Dokument im Sinne seiner *Dokumentenstruktur*. Beispiele für strukturelle Elemente sind Überschriften, Absätze, Tabellen und Aufzählungen. Für jedes strukturelle Element stehen Tags zur Verfügung, die in den Text eingebunden werden. Diese Tags beschreiben also das Wesen eines bestimmten Elementes, während der Web-Client für dessen konkrete Präsentation verantwortlich ist. Web-Clients können durch den Benutzer konfiguriert werden. Der Benutzer ist also die letzte Instanz und bestimmt die Präsentation der strukturellen Elemente. Klassische Dokumentenbeschreibungssprachen basieren im Gegensatz zu HTML häufig auf der Beschreibung einer konkreten Darstellung.

Ein wesentlicher Vorteil des Strukturansatzes gegenüber dem Präsentationsansatz besteht darin, dass Dokumente unabhängig von einer konkreten Plattform sind. In einem heterogenen Netz wie dem Internet ist diese Form der Portabilität unverzichtbar.

Die enorme Entwicklung des WWW in der zweiten Hälfte der 90er Jahre und dabei insbesondere die konkurrierende Entwicklung der Browser Netscape Navigator und Internet Explorer gingen zu Lasten dieser Portabilität. Die Browser-Hersteller haben ihre Produkte um proprietäre Präsentationselemente ergänzt. So ermöglicht Netscape beispielsweise die Verwendung eines <BLINK>-Tags. Solche und ähnliche HTML-Elemente, die nicht struktureller Natur sind, sondern eine konkrete Erscheinungsform festlegen, sind nicht im Sinne von HTML.

Mit HTML 4 wurde diese Entwicklung gestoppt. Es ging zurück zu den Wurzeln. Der Standard vertritt eine *Trennung von Struktur und Präsentation*. Letztere ist durch Stylesheets auszulagern und separat zu beschreiben. Das HTML-Dokument beschreibt lediglich die Struktur und ist mit dem Stylesheet zu verknüpfen. Auf HTML wird in Kapitel 4 näher eingegangen und auf Stylesheets in Kapitel 5.

1.6.1 Zusammengesetzte Dokumente

Der von einem HTML-Dokument darzustellende Inhalt besteht häufig nicht nur aus Text. Sind Bilder Bestandteil des Dokumentes, so wird mittels HTML lediglich festgelegt, woher sie geladen werden, wo sie dargestellt werden und welche Größe sie haben sollen. Die Bilder selbst sind nicht Bestandteil des HTML-Dokumentes. Sie werden vielmehr durch separate Anfragen des Clients an den Server geladen. So erfordert das Laden der Homepage von Amazon oder Ebay inklusive der Bilder zwischen zehn und zwanzig Anfragen an den Web-Server.

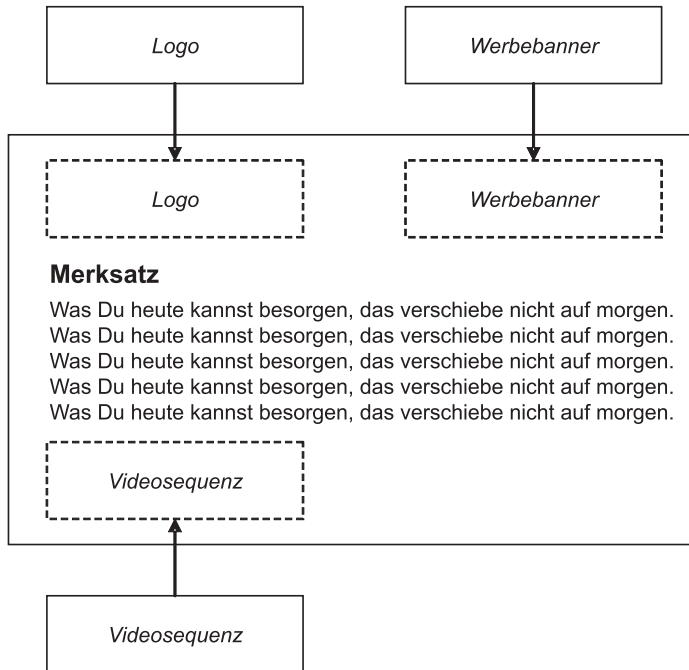


Abb. 1-8 *Zusammengesetzte Dokumente*

HTML-Dokumente sind *Container für Komponenten*. Komponenten können Daten, Programme oder eine Kombination von beidem sein. Sie können etwas präsentieren oder etwas bewirken. Komponenten können verschiedenen Typs sein, von unterschiedlichen Servern geladen werden und mit anderen Komponenten kommunizieren. Komponenten können ihrerseits wieder Container für andere Komponenten sein.

Das Konzept des zusammengesetzten Dokumentes ist offen. Im Gegensatz zu klassischen Applikationen müssen die zu verarbeitenden Datentypen nicht zur Kompilierzeit festgelegt werden. Zusammengesetzte Dokumente bieten Raum für existierende und auch zukünftige multimediale Anwendungen. Das HTML-Dokument stellt lediglich einen Layout-Rahmen dar.

1.7 Ressourcen

Die von einem Client bei der Anfrage an den Server spezifizierten Entitäten werden Ressourcen genannt. Im World Wide Web sind dies häufig HTML-Dokumente und Bilder. Das Web ist aber nicht auf solche Ressourcen beschränkt. Beispiele für weitere Ressourcen sind:

- Video-Sequenzen
- interaktive 3D-Medien
- PDF-Dokumente
- ZIP-Dateien
- serverseitige Programme

Im Gegensatz zu den ersten vier Dokumententypen werden *serverseitige Programme* nicht an den Client geliefert. Sie werden vielmehr auf der Serverseite ausgeführt. Dort können sie beispielsweise auf Datenbanken oder ERP-Systeme zugreifen. Als Resultat liefern sie typischerweise HTML-Dokumente, oftmals angereichert mit JavaScript.

Das Internet bietet eine Reihe weiterer *Ressourcen-Gattungen*. Der FTP-Server bietet vorwiegend Dateien für den Download an. Ein News-Server bietet News-Gruppen und News-Artikel als Ressourcen. Ein Telnet-Server bietet einen Benutzerzugang zur Steuerung eines entfernten Rechners.

1.8 Ressourcenadressierung

Die Adressierung jeglicher Art von Internet-Ressource erfolgt durch einen so genannten *URL*. Dies steht für Uniform Resource Locator. Damit können Dateien, aber auch E-Mail-Adressen, News-Artikel oder serverseitige Programme adressiert werden. Ein URL zeigt auf eine bestimmte Ressource an einem bestimmten Ort.

Ein URL spezifiziert in der Regel:

- ein Protokoll für den Zugriff auf die Ressource
- den Namen des Servers
- die Position der Ressource auf dem Server

Die allgemeine Syntax für einen URL ist:

```
protokoll://host[:port]/pfad/zu/ressource[#sektion]?parameter]
```

Zwei Beispiele dafür sind:

```
http://www.dozent.de/jur/impressum.html  
http://www.dozent.de:8080/jur/impressum.html
```

Das *Protokoll* wird gefolgt von einem Doppelpunkt und zwei Schrägstrichen. Im Beispiel war dies »http://«. Mögliche weitere Protokolle sind u.a.:

- file: Zugriff auf das lokale Dateisystem
- ftp: Kommunikation mit FTP-Server
- news: Kommunikation mit News-Server
- telnet: Kommunikation mit Telnet-Server

Der *Host* ist der Name des Servers, der die Ressource zur Verfügung stellt. In beiden Beispielen ist dies `www.dozent.de`. Alternativ dazu kann auch die IP-Adresse des Hosts angegeben werden, beispielsweise `194.97.203.126`.

Die Angabe der Portnummer ist optional. Sie ist nur dann notwendig, wenn der Dienst nicht auf dem Standard-Port 80 angeboten wird. Bietet der `dozent.de`-Server seinen WWW-Dienst auf Port 8080 an, ist die Portnummer explizit anzugeben, wie das zweite Beispiel zeigt.

Der dritte obligatorische Teil des URLs ist der *Pfad* zu der gewünschten Ressource. Typischerweise ist dies ein Dateisystempfad mit einem Dateinamen als abschließender Komponente. Die Syntax orientiert sich an UNIX, d.h. die einzelnen Bestandteile der Pfadangaben werden durch Schrägstriche getrennt.

Gelegentlich wird auf die Angabe des Dateinamens verzichtet. In diesem Fall hängt es von der Konfiguration des Servers ab, was er liefert. Häufig ist dies die Datei `index.html`. Es ist auch möglich, dass eine Liste der Dateien im spezifizierten Verzeichnis angezeigt wird oder dass eine Fehlermeldung geliefert wird.

Optional kann bei einem HTML-Dokument noch eine Sektion angegeben werden. Dabei handelt es sich um den Bezeichner einer Textmarke. Diese muss im HTML-Dokument festgelegt worden sein. Das Dokument wird dann beginnend mit dieser Positionszeile angezeigt. Die Sektion wird formal durch ein `#` vom Dateinamen getrennt.

Handelt es sich bei der referenzierten Ressource um ein Programm, können an den URL noch Parameter angehängt werden. Das Programm wird auf diese Weise zur Ausführung gebracht und bekommt die Parameter übergeben. Diese Parameter werden vom Dateinamen des Programmes durch ein `?` getrennt.

Die bisherige Beschreibung bezog sich auf *voll qualifizierte URLs*. Sämtliche Komponenten sind dabei anzugeben. Bei Spezifikation eines URLs in einem HTML-Dokument müssen nicht sämtliche Angaben gemacht werden. Von vorne beginnend können einzelne Komponenten weggelassen werden. In solch einem Fall erbt der URL die Werte nicht angegebener Komponenten von dem Dokument, in dem er spezifiziert ist.

Abbildung 1–9 zeigt ein Fragment des HTML-Dokumentes »`impresum.html`«. Der erste Hyperlink zeigt einen voll qualifizierten URL. Die restlichen vier URLs sind nicht voll qualifiziert. Der zweite und dritte URL beginnt jeweils mit einem Slash. Solche URLs werden *absolute URLs* genannt. Sie beschreiben lediglich den Pfad der Ressource. Sowohl das Protokoll als auch der Domainname werden von dem Dokument `http://www.dozent.de/jur/impresum.html` geerbt.

Die letzten beiden URLs sind Beispiele für *relative URLs*. Sie erben neben Protokoll und Domainname auch das aktuelle Verzeichnis. Der Pfad zur Ressource wird relativ zum Verzeichnis des aktuellen Dokumentes beschrieben. Das führende »..`<` im Beispiel steht für das direkt übergeordnete Verzeichnis.

`http://www.dozent.de/jur/impressum.html`

```
<a href=„http://www.dozent.de/index.html“>Home</a>
```

```
<a href=„./index.html“>Home</a>  
<img src=„./img/logo.gif“>
```

```
<a href=„../index.html“>Home</a>  
<img src=„../img/logo.gif“>
```

Abb. 1–9 Absolute und relative URLs

1.9 Web-Server

Der mit großem Abstand am weitesten verbreitete Web-Server ist *Apache*. Die folgenden Angaben beziehen sich auf dieses frei verfügbare Programm. Im Installationsverzeichnis befinden sich die Konfigurationsdateien des Apache-Servers im Unterverzeichnis »conf«.

Die wichtigsten Einstellungen sind:

- **ServerRoot:** absoluter Pfadname des Installationsverzeichnisses
- **Port:** Port, an dem der Web-Server auf Anfragen wartet
- **DocumentRoot:** absoluter Pfadname des Einstiegsverzeichnisses für HTML-Dokumente und andere statische Ressourcen
- **ScriptAlias:** absoluter Pfadname des Skript-Verzeichnisses `/cgi-bin/`
- **Alias:** absoluter Pfadname eines virtuellen Verzeichnisses
- **User:** Benutzername, unter dem der Serverprozess läuft

Wird im Rahmen eines URLs ein Pfad für ein HTML-Dokument spezifiziert, gilt dieser relativ zu *DocumentRoot*. Beginnt die Pfadangabe mit `/cgi-bin/`, wird sie relativ zu *ScriptAlias* interpretiert. Darüber hinaus können weitere Aliase definiert werden. Der Beginn des Pfadteiles eines URLs wird auf das jeweils angegebene Verzeichnis abgebildet.

Abbildung 1–10 zeigt die Verzeichnishierarchie einer Beispielinstallation unter UNIX:

Die konkreten Werte einiger Einstellungen gemäß Schaubild sind wie folgt:

- **ServerRoot:** `/usr/local/apache`
- **DocumentRoot:** `/usr/local/apache/htdocs`
- **ScriptAlias:** `/cgi-bin/ /usr/local/apache/cgi-bin/`

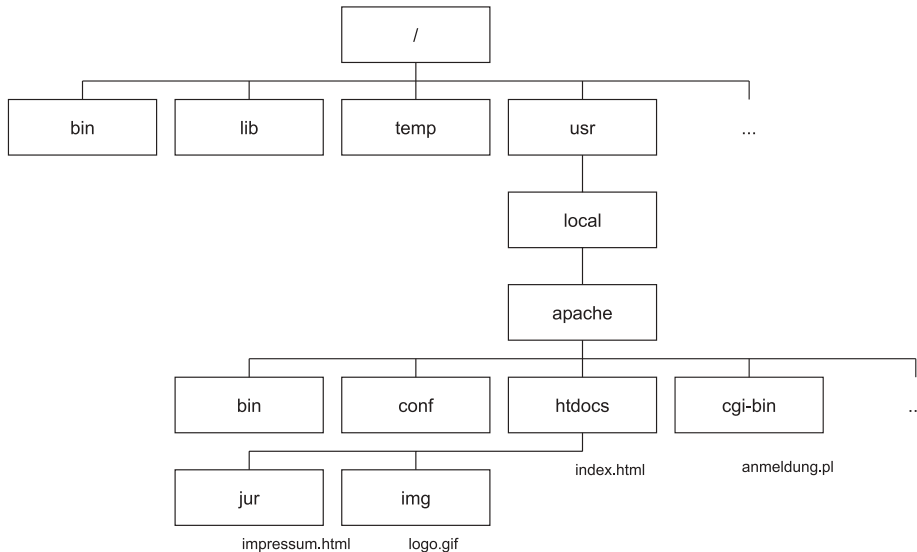


Abb. 1-10 Verzeichnishierarchie Apache Web-Server

1.10 Proxy-Server

Zwischen Web-Client und -Server kann sich ein Proxy-Server befinden. Insbesondere erfolgt der Zugang zum World Wide Web für die meisten Anwender in lokalen Netzen über einen Proxy-Server. Der Proxy-Server ist ein intelligentes Zwischenglied zwischen Client und Server. Dem Web-Client gegenüber tritt er als Web-Server auf und dem Web-Server gegenüber als Web-Client.

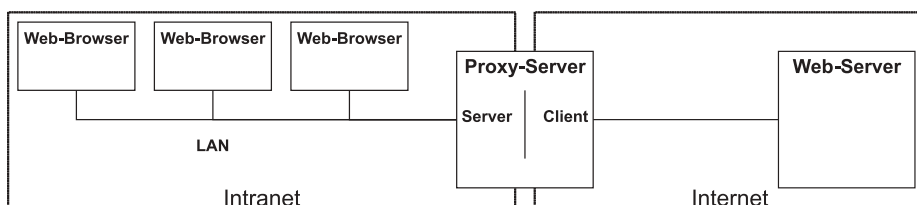


Abb. 1-11 Proxy-Server

Der Proxy-Server leitet die Anfragen eines Web-Browsers an den adressierten Web-Server weiter und nimmt die vom Server gelieferten Ressourcen entgegen, um sie an den Client zurückzuliefern.

Ein Proxy-Server hat folgende Aufgaben:

- Schutz des LANs gegen Angriffe von außen
- Beschränkung des Zugriffs von LAN-Rechnern auf das Internet
- Caching von Internet-Ressourcen

Proxy-Server sind häufig Bestandteil einer Firewall. Eingehende Pakete lassen sich nach bestimmten Kriterien, wie beispielsweise Ziel-Rechner oder -Port, herausfiltern. Bei Internet-Zugriffen aus dem Intranet heraus tritt der Anfragende mit der Absender-IP-Adresse des Proxy-Servers auf. Seine konkrete IP-Adresse ist im Internet damit nicht bekannt.

Proxy-Server eignen sich auch für die Beschränkung des Internet-Zugriffes. Bei erstmaligem Internet-Zugriff verlangt der Proxy-Server eine Authentifizierung. Abhängig vom Profil des Benutzers kann der Zugriff auf bestimmte Bereiche des Internets beschränkt werden. Außerdem können auf dem Proxy-Server schwarze Listen mit nicht zulässigen Web-Adressen hinterlegt werden.

Proxy-Server können so konfiguriert werden, dass sie die von einem Web-Server gelieferten Ressourcen zwischenspeichern. Wird dieselbe Ressource innerhalb eines bestimmten Zeitraumes nochmals aufgerufen, ist der Aufbau einer externen Internet-Verbindung nicht mehr notwendig. Der Client bekommt die zwischengespeicherte Ressource unmittelbar vom Proxy-Server geliefert. Dies erspart dem Unternehmen Leitungskosten und dem Anwender lange Wartezeiten. Der Vorteil des Caching kommt insbesondere dann zum Tragen, wenn die Anwender regelmäßig dieselben Ressourcen abrufen.