Multimediale Client-Server-Systeme

Bearbeitet von
Klaus Chantelau, René Brothuhn

1. Auflage 2009. Taschenbuch. x, 357 S. Paperback
ISBN 978 3 540 79748 7
Format (B x L): 15,5 x 23,5 cm
Gewicht: 623 g

Kapitel 1
Multimediale Client-Server-Systeme

1.1 Einleitung

Die in der Einführung des Buches betrachteten Anwendungsszenarien stellen Beispiele für verteilte multimediale Systeme dar. Zum Beispiel werden bei der Darstellung eines Tourismusangebots einer Urlaubsregion neben klassischen Internetdiensten, wie z.B. Hotelsuche- und Buchungssysteme, auch umfangreiche interaktive Medien (z.B. Grafiken, Fotographien, Audio- und Videodaten) zur Darstellung der Angebote verwendet. Eine Vielzahl von Anwendern (Clients) fordern meist gleichzeitig und über mehrere Kommunikationsschritte die gewünschten Inhalte von den Serversystemen an.

Die dargestellten Medien und Anwendungen besitzen meist einen hohen Grad an Komplexität, weshalb entsprechende Serversysteme für die Bereitstellung solcher Buchungssysteme, Grafiken und Fotomaterialien erforderlich sind. Für die Bereitstellung von Audio- und Videodaten sind spezielle Medien- bzw. Streamingserver erforderlich, welche für eine hochperformante Übertragung der abgerufenen Audio- und Videostreams zuständig sind.

In allen Fällen kommt dabei das Client-Server-Modell zum Einsatz, wobei ein Server gewisse Dienste zur Verfügung stellt (z.B. Web-Server oder Medien-Server), welche von den Clients abgerufen und in Anspruch genommen werden. Aufgrund der komplexen Struktur multimedialer Anwendungen ergeben sich dabei häufig auch komplexe Strukturen zwischen Client und Server. So kann z.B. ein Server selbst wiederum Dienste eines anderen Servers in Anspruch nehmen oder auch einem Client eine Verbindung zu einem speziellen Server überreichen, welcher einen anderen Teil der multimedialen Anwendung übernimmt. Wie in Abb. 1.1 zu sehen, nutzt ein Web-Server die Dienste eines Datenbankservers und übergibt gleichzeitig dem Client eine Verbindung zu einem Medien-Server, welcher z.B. Bilder, Audio- oder Videostreams zur Verfügung stellt.

Andererseits können mehrere Server die gleichen Aufgaben übernehmen, um somit eine Lastverteilung zu erreichen. Auf diese Weise kann eine Vielzahl von Clients mit den gewünschten Daten versorgt und eine Überlastung der Server vermie-

K. Chantelau, R. Brothuhn, *Multimediale Client-Server-Systeme*, eXamen.press,
DOI 10.1007/978-3-540-79749-4_1, © Springer-Verlag Berlin Heidelberg 2010

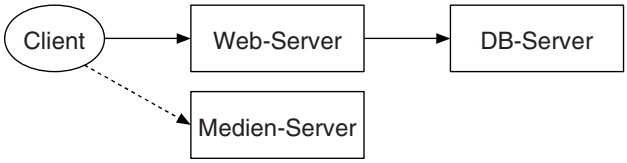


Abb. 1.1 Client-Server Struktur mit unterschiedlichen Aufgaben der Server

den werden. Dies ist z.B. insbesondere bei Videoplattformen von hoher Bedeutung, wo sehr viele Clients das gleiche Live-Video präsentiert bekommen möchten. Eine systhematische Einführung in die Architekturen und Prozesse von verteilten Systemen ist in Kapitel 2 wiedergegeben.

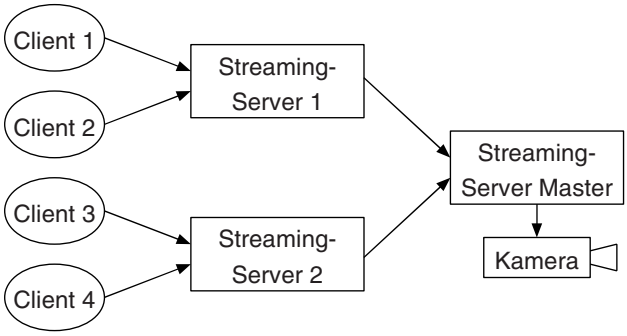


Abb. 1.2 Server-Kette mit gleichen Aufgaben der Server zur Lastverteilung

1.2 Multimediale Daten in Client-Server-Systemen

Der umfangreiche Einsatz von Medien erfordert in Client-Server-Systemen den Einsatz von besonderen Technologien, über die wir in diesem Kapitel zunächst einen Überblick geben werden. Es ist auf Grund der technischen Schwerpunktsetzung nicht erforderlich, eine umfangreiche medientheoretische Diskussion zur Klassifikation von Medien zu führen. Ein multimediales System wird schlicht als ein System aufgefasst, welches mehrere Medienformen nutzt, wobei wir folgende Medienformen unterscheiden:

- Text
- Fotografien und Rasterbilder
- Computergrafisch erzeugte Einzel- und Bewegtbilder
- Audio
- Video

Im Folgenden wollen wir (mit Ausnahme der technisch gesehen unproblematischen Medienform Text) einige grundsätzliche Aspekte dieser Formen benennen, die sich auf die in Frage kommenden Technologien auswirken.

1.2.1 Fotografien und Rasterbilder

Ein elektronisches Farbbild entsteht dadurch, dass an jeder Bildstelle drei Farbwerte mit unterschiedlicher Intensität in kleinen Bereichen angezeigt werden (ein Pixel besteht aus einem Rot-, Grün- und Blauwert). Bei einer Fotografie werden von Sensoren in einer Kamera diese drei Farbwerte an jeder Bildstelle ermittelt. Bei der Wiedergabe auf einem Bildschirm sind diese Bereiche sind so klein, dass das Auge nicht die einzelnen Farbwerte wahrnimmt, sondern eine Mischfarbe, welche sich nach den Gesetzen der additiven Farbmischung ergibt (eine genauere Darstellung erfolgt in Abschnitt 5.1.3). Bei einem 15-Zoll-Bildschirm mit einer Auflösung von 1024 x 768 Pixeln misst ein Pixel etwa 0,3 mm. Bei den meisten digitalen Rasterbildern wird jeder Farbwert durch 8 Bit = 1 Byte dargestellt. Damit können 256 (2^8) Werte unterschieden werden.

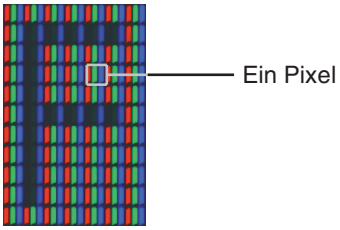


Abb. 1.3 Pixel in einem Rasterbild, welches eine Makroaufnahme des Buchstabens F zeigt, der auf einem Farbmonitor dargestellt wurde. Der Buchstabe F wird in blauer Farbe dargestellt

1.2.2 Computergrafisch erzeugte Einzel- und Bewegtbilder

Clientsysteme besitzen visuelle Elemente, welche computergrafisch generiert werden. Die meisten Anwendungen im Internet verwenden auch heute noch zweidimensionale Computergrafiken. Wenige Ausnahmen sind Anwendungen wie z.B. GoogleEarth, wo die Landschaft und Bebauung durch dreidimensionale Computergrafiken dargestellt werden. Zweidimensionale Computergrafiken sind wesentlich einfacher aufgebaut als dreidimensionale Computergrafiken. Folgende computergrafische Begriffe werden in diesem Buch benötigt:

Vektorgraphiken Bei diesen Graphiken werden auf der Basis von mathematischen Gesetzmäßigkeiten die Formen von Objekten sowie Linien- und Farbfüllungen festgelegt. Hierbei werden grafische Primitive (z.B. Kreis und Rechteck), Polygonzüge und spezielle parametrisierte Kurven (z.B. Bézierkurven) zur Festlegung von 2D-Formen verwendet. Die Grafiken sind unabhängig von der Auflösung und können ohne Qualitätsverlust bearbeitet werden. Erst zur Darstellung werden sie in eine Rastergrafik konvertiert, in den Bildspeicher auf der Graphikkarte transferiert und auf dem Bildschirm angezeigt (Abb. 1.4).

z-index In zweidimensionalen Grafiken wird keine z-Koordinate benötigt, welche die Entfernung eines Objektes angibt. Jedoch ist entscheidend, welches Objekt näher an einem Betrachter ist, um somit ein dahinterliegendes Objekt zu überdecken. Diese Anordnungsreihenfolge wird durch den z-Index angegeben.

alpha-Kanal Ein sogenannter alpha-Kanal dient zur Darstellung von Transparenzeigenschaften, ein dahinterliegendes Objekt kann dadurch teilweise sichtbar sein.

Bewegtbilder / Animationen Animationen werden bei den in diesem Buch betrachteten Technologien über Ereignisse gesteuert. Diese Ereignisse können bei jedem neu dargestellten Bild oder von speziellen Zeitereignissen ausgelöst werden und eine Neuzeichnung eines Bildes bewirken. Dabei können Werte (z.B. Position und Farbe von Vektorgraphiken) in sogenannten Schlüsselbildern vorgegeben und für alle weiteren Bilder durch Interpolation berechnet werden (sog. Keyframeanimation bzw. Tweening). Die Keyframeanimation ist nur eine Animationstechnik von mehreren Möglichkeiten, welche jedoch in vielen Fällen genutzt wird. Die Bewegtbilder müssen, bevor sie angezeigt werden können, ebenfalls in Rasterbilder gewandelt werden.

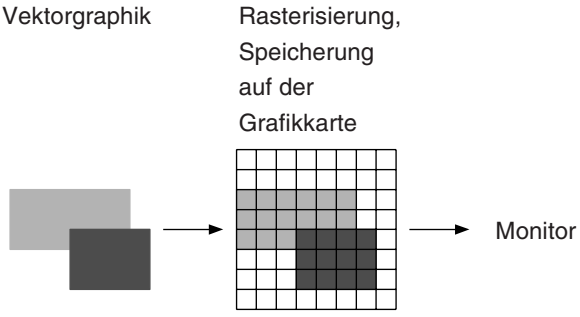


Abb. 1.4 Wandlung einer Vektorgrafik in eine Rastergrafik

1.2.3 Audio- und Videomedien

Ein Video besteht aus einer Folge von Rasterbildern, welche ein riesiges Datenvolumen generieren. Ähnliche Verhältnisse liegen bei Audiosignalen vor, welche ebenfalls aus einer Folge von Abtastwerten bestehen. Für die Übertragung von Audio- und Videomedien ist eine massive Reduzierung der Datenmenge durch geeignete Kompressionsverfahren erforderlich. Hierbei wird eine möglichst gleichbleibend hohe Medienqualität angestrebt. Ohne massive Reduzierung der Datenmenge würde eine Übertragung über das Internet eine inakzeptable Zeit in Anspruch nehmen. Streaminganwendungen wären so nicht möglich.

Ein nach DVD-PAL standardisiertes Fernsehbild hat 720 x 576 Pixel. Bei 25 Bildern pro Sekunde und je ein Byte für jeden Farbwert Rot, Grün und Blau würde folgende Datenrate entstehen: 720 x 576 x 25 x 3 Byte = 31.104.00 Bytes, also ca. 31 MB/s was einer Bitrate von ca. 250 MBit/s entspricht.

Selbst bei den heute weit verbreiteten DSL 6000-Anschlüssen, welche Daten mit einer Rate von über 5 MBit/s transportieren können, wäre es unmöglich, Videos mit einer PAL-Auflösung über das Internet in Echtzeit zu verbreiten. Die erforderliche Datenrate wäre um den Faktor 50 zu groß. Durch geeignete Transformationen der Daten (Encodierung) kann man jedoch ohne sichtbare Fehler die benötigte Datenrate auf 5 MBit/s herabsetzen, so dass ein Transport von Fernsehbildern in gewohnter Qualität über die genannten DSL 6000-Verbindungen in Echtzeit möglich wird. Diese Datenreduktion um den Faktor 50 und mehr wurde nach intensiver Forschungsarbeit in den Standards JPEG, MPEG 1, MPEG 2, MPEG 4/H264 und deren Nachfolgern erzielt. Eine ähnliche Situation liegt im Bereich der Audiostandards vor. Bei einer Übertragung von Audio- und Videomedien (AV-Medien) liegt daher folgende Verarbeitungskette vor:

- Am Server/Sender:
  1. Digitalisierung von analogen AV-Daten (Capturen von AV-Medien)
  2. Komprimierung der digitalisierten Daten (Encodierung)
  3. Verpackung der komprimierten Daten in eine Containerdatei bzw. in Datenpakete eines Netzwerkprotokolls, insbesondere im Falle einer Echtzeitübertragung (Paketierung)
- Übertragung über ein Netzwerk:

  Es erfolgt der Download der Datei oder ein (Echtzeit-) Streaming von Datenpaketen.
- Beim Client/Empfänger:
  1. Entpacken der komprimierten Daten aus der Kontainerdatei oder den Datenpaketen (Depaketierung)
  2. Dekompression der komprimierten Daten (Decodierung)
  3. Wiedergabe der dekomprimierten Daten

1.3 Interaktion mit Client-Systemen

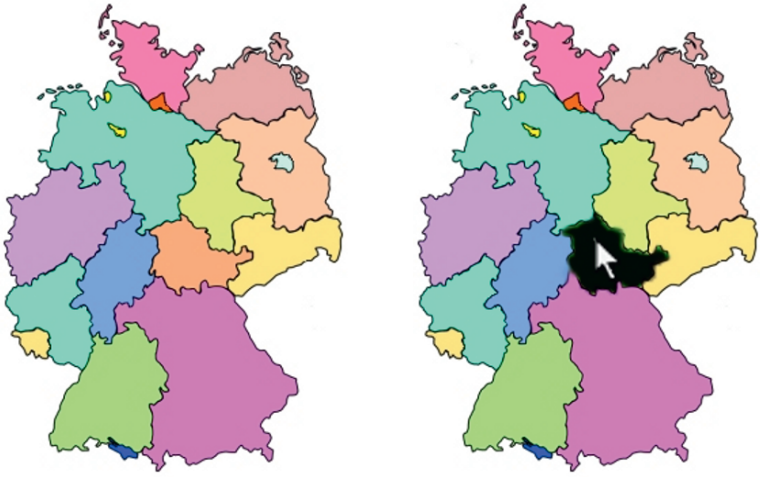
Die Mensch-Maschine-Interaktion wird auf der technischen Ebene in Eingabe- und Ausgabekomponenten unterteilt [2]. Die Ausgabekomponenten stellen Informationen auf der Basis der zuvor genannten Medien dar. Die Interaktion mit den Eingabekomponenten besteht bei heutigen grafisch orientierten Client-Systemen im überwiegenden Maße aus Maus- (oder Touchscreen-) Interaktionen. Die Tastatur ermöglicht in Ergänzung dazu eine schnellere Bedienung über sog. Short-Cuts bzw. die umfangreiche Eingabe von Text. Die Interaktion mit der Maus findet insbesondere folgendermaßen statt:

Interaktive Grafiken In Grafiken werden bestimmte Bereiche durch Rahmen oder Kurvenzüge eingegrenzt, welche Ereignisse generieren (Maus-sensitiver Bereich) und für die Steuerung einer Anwendung genutzt werden. So z.B. wird in Abb. 1.5 das Bundesland Thüringen eingeschwärzt, wenn die Maus über die Region des Bundeslandes bewegt wird. Die gebräuchlichen Eingabeelemente in Nutzeroberflächen, wie Buttons oder Menüs, bestehen ebenfalls aus Grafiken, welche Maus-sensitive Bereiche enthalten und Ereignisse erzeugen.

Interaktives Video In einem zunehmenden Maße werden auch interaktive Videopräsentationen verwendet [5], wo in einem Bild eines Video mehrere Maus-sensitive Bereiche (sog. Hot Spots) eingeblendet werden. Zum Beispiel finden sich bei der Präsentation von neuen Fahrzeugen vielfach Videos, wo das Video an einem ausgezeichneten Bild selbstständig stoppt und der Zuschauer über Hot Spots auswählen kann, was als Nächstes gezeigt werden soll, wie z.B. der Innenraum, der Kofferraum, der Motorraum oder die Federung des Fahrwerks.

Interaktiver Text/Hyperlinks Die sehr einfachen und ursprünglichen Interaktionen bei der Internetnutzung bestehen darin, über Hyperlinks (Referenzen) von einem Dokument zum einem anderen Dokument zu wechseln. Der Hyperlink stellt dabei einen Maus-sensitiven Textbereich dar. Durch einen Maus-Klick auf den Hyperlink wird eine Referenz in einem Dokument sofort (auch von unterschiedlichen Servern) abrufbar gemacht. Die Interaktion war bei den ersten Web-Anwendungen fast ausschließlich auf diese Form begrenzt. Die Hyperlinks bilden daher eines der grundlegendsten Konzepte des frühen Internets, welche in HTML durch die `<a>`-Tags abgebildet werden. Eine Gruppe von verlinkten Dokumenten bildet ein Hypertext-Medium, wo sich der Nutzer nicht mehr von einer zur nachfolgenden Seite in einer linearen Art und Weise durcharbeitet, sondern wo er sich von einem Dokument zu einer aus vielen Möglichkeiten ausgewählten Referenz (nichtlinear) weiterbewegt.[1] Da die Hyperlinks neben Texten auch andere Medienformen referenzieren können, spricht man oft auch von Hypermedia-Systemen.

[1] das sprichwörtliche Surfen im Internet



(a) Normale Ansicht der Deutschlandkarte

(b) Region wird eingeschwärzt, sobald sich die Maus darüber befindet

Abb. 1.5 Deutschlandkarte mit eingetragenen Bundesländern, jedes Bundesland bildet eine Maussensitive Region

1.3.1 Grafische Nutzeroberflächen

Eine Anwendung wird im Wesentlichen über die oben beschriebenen Interaktionen mit den Ein- und Ausgabekomponenten gesteuert. In den meisten Fällen werden für die Bedienung einer Anwendung eine Vielzahl von Interaktionen benötigt, deren Gesamtheit die Nutzerschnittstelle bildet. Diese wird meist als grafische Nutzeroberfläche bezeichnet, für deren Gestaltung die folgenden Aspekte zu berücksichtigen sind:

Layout, Nutzerführung und Design Die räumliche Anordnung dieser Komponenten (Layout), die Art und Weise wie die Komponenten miteinander verknüpft sind (Nutzerführung, Dialoggestaltung), sowie die visuelle Darstellung der Komponenten (Design) sind wichtige Gestaltungsmerkmale einer Anwendung.

Synchronisation Bei der Einbettung von zeitbasierten Medien wie Audio, Animationen und Video müssen alle diese Medien auf der Zeitachse synchron gehalten werden. Zu einem gewissen Bild in dem Video sollen z.B. ergänzende Textinformationen, Grafiken oder Hot Spots zur weiteren Ablaufsteuerung eingeblendet werden. Die zeitliche Synchronisation kann über unterschiedliche Mechanismen und Werkzeuge erfolgen. Wie bei einem Videoschnittprogramm können über einer Zeitleiste alle Medien angeordnet und durch einen entsprechenden Player gestartet und gestoppt werden. Eine andere Möglichkeit besteht in der Verwendung von sogenannten Timer-Objekten, welche zu ausgewählten Zeitpunkten Ereignisse versenden

Usability Ein wichtiges Ziel bei der Konzeption und Optimierung von Layout, Nutzerführung, Design und Synchronisation ist es, eine nutzerfreundliche Bedienung zu gewährleisten. Mit Nutzerfreundlichkeit ist gemeint, wie effizient, effektiv und mit welcher Zufriedenheit ein Nutzer Aufgaben bei der Bedienung einer Anwendung ausführen kann. Die Nutzerfreundlichkeit (Usability) wird oft über Kriterien wie Aufgabenangemessenheit, Steuerbarkeit, Erwartungskonformität, Selbstbeschreibungsfähigkeit und Fehlerrobustheit gemessen [2].

1.3.2 Rich-Clients

Die Clients in verteilten multimedialen Systemen übernehmen vielfältige und teilweise sehr rechenintensive Aufgaben (z.B. die Dekompression von Audio- und Videomedien), die durch die Integration und Kombination der oben genannten Medien entstehen. Die grafischen Oberflächen von Web-Anwendungen werden zudem immer nutzerfreudlicher. Das Design und die Nutzerführung (z.B. ansprechende Menüs sowie Drag- und Drop-Funktionalität) erreichen einen Bedienkomfort, der in vielen Fällen lokal installierten Anwendungen entspricht. Hierzu sind weitere aufwändige Verarbeitungsprozesse auf dem Client erforderlich.

Da die Anforderungen an einen solchen Client recht hoch sind, hat sich hierfür die Bezeichnung *Rich-Client* durchgesetzt (als Abgrenzung zu den Begriffen Fat- und Thin-Client siehe Kap. 2). Die entsprechenden Web-Anwendungen werden als *Rich Internet Applications* bezeichnet.

1.4 Technologien zur Entwicklung multimedialer Client-Server-Systeme

Zur Entwicklung von multimedialen Client-Server-Systemen sollen in diesem Abschnitt Technologien vorgestellt werden. Die Entwicklung von Internet- und Multimediatechnologien verläuft meist über einen Zeitraum von mehreren Jahren. Sie beginnt mit vereinzelten Forschungsprojekten, welche von Unternehmen, Institutionen und Hochschulen durchgeführt werden. Gerade in den Bereichen der Informations- und Telekommunikationsbranche ist es ab einem gewissen Reifegrad erforderlich, die Ergebnisse zusammenzufassen und zu normieren. Auf der Basis dieser Normierungen werden von den Unternehmen Produkte (z.B. Softwarebibiliotheken) entwickelt. Wir werden daher im Folgenden sowohl die Standards als auch die Softwarebibiliotheken und Programmierschnittstellen betrachten.

1.4.1 Internationale Standards, Protokolle, Empfehlungen und Programmierschnittstellen

Um eine korrekte Datenübertragung und Verarbeitung zwischen zwei Rechnersystemen wie Client und Server gewährleisten zu können, müssen die Daten auf der Basis normierter Formate und Prozesse verarbeitet bzw. übermittelt werden. Bei der Entwicklung normierter Formate werden auch immer Optimierungen hinsichtlich bestimmter Kriterien wie z.B. Funktionalität, Medienqualität oder Datenvolumen betrachtet.

Internationale Standards Das wichtigste zwischenstaatlich anerkannte Standardisierungsgremium ist die International Standardisation Organisation (ISO). In umfangreichen Forschungsarbeiten sind hier die wichtigsten Bild- (JPEG, PNG), Audio- (MP3) und Videoformate (MPEG) erarbeitet worden. Die Standardisierungen werden von Forschungsgruppen aus Unternehmen, Instituten und Hochschulen in umfangreichen mehrjährigen Forschungsarbeiten vorbereitet. Für die Telekommunikationsbereiche werden die Standards von der International Telekommunication Union (ITU) herausgegeben. Teilweise übernehmen die unterschiedlichen Organisationen die Standards anderer Organisationen (z.B. ISO Standard MPEG 4 Teil 10 entspricht z.B. ITU H.264)

W3C-Empfehlungen Das sogenannte W3C-Konsortium gibt wichtige Empfehlungen für Internettechnologien heraus. Die wichtigsten Empfehlungen (HTML, XML, CSS, JavaScript, SVG, SMIL, DOM, ...) besitzen jedoch die gleiche verbindliche Bedeutung wie die von der ISO herausgegebenen Standards. Sie werden jedoch nicht als Standards bezeichnet, weil es keine offizielle zwischenstaatliche Anerkennung des W3C gibt.

Internet Protokollstandards Eine ähnliche Situation liegt bei den wichtigsten normierten Protokollen (TCP, UDP, HTTP, RTP, ...) für die Übertragung von Informationen zwischen Rechnersystemen vor. Diese werden als RFCs (Request for Comments) von der ISOC (Internet Society) herausgegeben.

Proprietäre Standards In vielen Fällen werden von Unternehmensverbünden (z.B. der MIDI-Standard der Musikgerätehersteller) oder von einzelnen Unternehmen (z.B. Adobes Flash flv-Format und Microsofts wmv-Format) bedeutsame Formate herausgegeben. Der Aufbau dieser Formate wird oft nicht veröffentlicht und kann daher in vielen Fällen nur von Software- bzw. Hardwarekomponenten aus dem jeweiligen Unternehmen verwendet werden. Ein erfolgreiches Format kann ein entscheidender Schlüssel zu einem Markterfolg werden und die gleiche Bedeutung wie ein internationaler Standard erlangen. In diesem Fall spricht man von einem proprietären Standard. Proprietäre Standards sind problematisch, wenn der Hersteller vom Markt verschwindet oder die Fortführung des Produktes einstellt.

Oft arbeiten viele auch konkurrierende Unternehmen und Hochschulen zunächst unter einer Dachorganisation wie der ISO gemeinsam an einem Standard (wie z.B.

bei den MPEG Standards). Später fließen die Erkenntnisse in proprietäre Standardformate ein. So z.B. haben große Firmen wie Adobe, Apple und Microsoft an der Entwicklung des MPEG4-Standards mitgewirkt und später eigene Formate herausgebracht, die nun überall Anwendung finden. Wir werden im Folgenden diese Differenzierungen vernachlässigen und allgemein einfach von Standards sprechen.

Programmierschnittstellen Auf der Basis dieser Standards werden überwiegend von Unternehmen sogenannte Programmierschnittstellen (Application Programming Interface - API) entwickelt und für die Anwendungsentwicklung eingesetzt. Die Programmierschnittstellen stellen in den meisten Fällen umfangreiche Funktionalitäten auf der Basis von Softwarebibliotheken bereit.

1.4.2 Standards zur Entwicklung von Rich-Clients

Rich-Client-Technologien sind auf der Basis der Kerntechnologien des Internets entstanden. Diese sollten zunächst das Hypertext-Konzept technisch ausfüllen. Multimedial orientierte Standards sind dann erst Ende der 1990er Jahre hinzugekommen. Der erste Einsatz der Kerntechnologien betraf den schnellen Zugriff auf eine große Zahl von Forschungsdokumenten, welche dezentral abgelegt und miteinander über Hyperlinks verknüpft wurden. Diese Zielsetzung wurde 1989 von dem britischen Informatiker Tim J. Berners-Lee am Schweizer Institut CERN der Europäischen Organisation für Kernforschung umgesetzt. Hier sind das Adressierungsschema URL (bzw. URI), das Übertragungsprotokoll HTTP und das Dokumentenformat HTML (Hypertext Mark Up Language) entwickelt worden.

Hypertext Markup Language (HTML) Ist eine Tag-basierte Sprache zur Strukturierung und Formatierung von Dokumenten. Die Tags können neben der Darstellung von Bildern auch Bedienelemente wie Hyperlinks umfassen, mit denen man zu einem anderen Dokument wechseln kann. Durch Einfügen von Bedienelementen und Formatieranweisungen lassen sich mit HTML Nutzeroberflächen aufbauen. Eine Einführung hierzu ist in Abschn. 4.3 wiedergegeben. HTML ist ein Teil des übergeordneten Dokumentenformats SGML, der in den 1960er Jahren entwickelt wurde und heute durch XML ausgefüllt wird.

Cascading Stylesheets (CSS) Eine konsequentere Trennung des Designs und Layouts vom Inhalt von Webseiten wurde Ende der 1990er Jahre durch die Einführung von Cascading Style Sheets (CSS) ermöglicht. Eine Einführung ist ebenfalls in Abschn. 4.3 wiedergegeben.

JavaScript/Document Object Model Mitte der 1990er Jahre wurde mit Hilfe von JavaScript die Möglichkeit eröffnet, Formulare mit JavaScript auf Clientseite auszuwerten und einfache animierte Grafiken einzuführen. In Ergänzung dazu wurde das Document Object Model (DOM) eingeführt, welches eine Spezifikation zum Zugriff auf HTML- oder XML-Dokumente darstellt. JavaScript

und DOM haben sich seitdem beständig für den Einsatz in dynamischen Webseiten weiterentwickelt. Eine Einführung in diese Technologien ist ebenfalls in Abschn. 4.3 dargestellt.

Das Bemerkenswerte an allen diesen Standards ist, dass sie von allen gängigen Browsern ohne weitere Plugins unterstützt werden. Jedoch wird mit HTML, CSS und JavaScript allein noch kein Bedienkomfort erreicht, wie er mit Desktopanwendungen vergleichbar ist. Dies liegt vor allem an dem zeitlichen Reaktionsverhalten. Mit diesen Techniken ist nur eine synchrone Kommunikation zwischen Browser und Server möglich. Der Client bleibt zwischen dem Absenden einer neuen Anfrage und dem Eintreffen der Server-Antwort blockiert. Trifft die Antwort ein, so wird die alte Seite komplett durch eine neue Seite ersetzt. Auch wenn es nur Sekunden sind, die zwischen den beiden Ereignissen liegen, so wird dennoch eine flüssige Bearbeitung einer Aufgabe dadurch unmöglich. Darüber hinaus können mühsam gesuchte Informationen durch das vollständige Ersetzen der alten Seite durch eine neue Seite nicht selbstverständlich weiterverwendet werden, sondern werden vielfach einfach gelöscht. Erst durch den Einsatz von AJAX (Asynchrones JavaScript und XML) konnte dieses Defizit in den letzten Jahren weitgehend behoben werden.

Speziell für die Entwicklung von Rich-Clients sind in den vergangenen 10 Jahren ferner die folgenden multimedial orientierten Standards veröffentlicht worden:

Scalable Vector Graphics (SVG) SVG ist ein Standard zur Beschreibung zweidimensionaler Vektorgrafiken und Animationen mit Hilfe einer Tag-basierten XML-Syntax.

Synchronized Multimedia Integration Language (SMIL) SMIL ist ein XML-basierter Standard zur einheitlichen räumlichen Komposition, Synchronisation und Steuerung aller genannten Medienformen Text, Grafik, Bilder, Audio und Videos [1]. SMIL und SVG ergänzen sich gegenseitig.

Darüber hinaus gibt es weitere Standards wie VRML und X3D mit einer stärkeren Ausrichtung auf dreidimensionale Computergrafik, die in diesem Buch nicht weiter thematisiert werden. Die Formate SMIL und SVG werden nicht von allen gängigen Browsern unterstützt, hierfür wird ein weiteres Plugin benötigt.

1.4.3 Programmierschnittstellen für die Entwicklung von Rich-Clients

Für die Entwicklung von Client-Systemen werden Programmiersprachen benötigt. Allerdings brauchen immer wiederkehrende Softwarebausteine nicht von jedem Entwickler neu erfunden werden. Daher werden umfangreiche Softwarebibliotheken angeboten, in denen die wichtigsten Komponenten und Funktionalitäten schon fertig ausgearbeitet vorliegen. Eine Programmiersprache mit entsprechenden Softwarebibliotheken wird als Programmierschnittstelle (Application Programming Interface - API) bezeichnet. Gerade für die Client-Server-Kommunikation und die

Medienverarbeitung müssen diese APIs natürlich die genannten Standards unterstützen.

1.4.3.1 Die Java-Plattform

In den 1990er Jahren hat sich die Java-Technologie mit einer starken Ausrichtung auf Anwendungen im Bereich der verteilten Systeme entwickelt. Die Java-Technologie muss grundsätzlich in 2 Teile unterschieden werden: 1. die Programmiersprache Java und 2. die Java-Plattform.

Java wurde von der Firma Sun Microsystems entwickelt. Eines der Ziele war es, geschriebene Programme plattformunabhängig zu gestalten, d.h., einmal geschriebene Programme sollen ohne Änderung auf unterschiedlicher Hardware und unterschiedlichen Betriebssystemen lauffähig sein.[2] Hierzu werden die Java Programme nicht wie bei vielen anderen Sprachen üblich in Maschinencode übersetzt, sondern in einen „Zwischencode“, den sog. Java-Bytecode. Dieser Bytecode wird dann von einer Virtuellen Maschine (der Java Virtual Machine, JVM) ausgeführt. Diese JVM stellt in gewisser Weise ein Betriebssystem mit einem virtuellen Prozessor dar. Dieser virtuelle Prozessor übersetzt den Java-Bytecode auf der Zielhardware in Maschinencode und führt somit die Java Programme aus. Die JVM stellt deshalb eine Plattform dar. Ohne diese Plattform sind Java Programme nicht lauffähig.

Darüber hinaus stellt die JVM, wie ein Betriebssystem auch, bestimmte Funktionen zur Verfügung, die von den Programmen genutzt werden können. Diese Funktionen werden dem Entwickler in Form einer riesigen Klassen-Bibliothek und den dazugehörigen APIs zur Verfügung gestellt. Im Laufe der Jahre hat sich der Funktionsumfang der Java-Plattform enorm erweitert, und es bedarf einiger Erfahrung, einen Überblick zu erhalten und die APIs effizient einzusetzen. Hinzu kommt, dass es je nach Einsatzzweck unterschiedliche Plattformen gibt:

- Java Standard Edition (Java SE) - Die Java SE repräsentiert die Standard-Version der Java-Plattform. Es gibt für die Programmierung von verteilten Systemen ein leistungsstarkes Netzwerk API (`java.net.*`-Package). Eine synchrone und asynchrone Client-Serverkommunikation lässt sich damit problemlos bewerkstelligen. Ebenso kann Java durch die umfangreichen Bibliotheken zur Entwicklung von Oberflächen (`javax.swing.*`-Package) eingesetzt werden. Mit dem `javax.swing.*`-Package werden ferner große Möglichkeiten zur Generierung von Vektorgrafiken und Animationen bereitgestellt. Mit Hilfe von Java-Applets hat man darüberhinaus die Möglichkeit, alle diese Bibliotheken für Web-Anwendungen genauso wie für Desktopanwendungen einsetzen zu können. Applets sind Java-Anwendungen, welche durch spezielle Tags in HTML-Seiten integriert und auf dem Client ausgeführt werden können. Die Java SE-Plattform ist daher insbesondere auch für die Entwicklung von Rich-Clients geeignet.

[2] Write Once, Run Anywhere

- Java Micro Edition (Java ME) - Eine speziell für kleine oder mobile Endgeräte (Handys, PDAs usw.) konfigurierte Plattform, die gegenüber der Java SE im Funktionsumfang stark eingeschränkt ist.
- Java Enterprise Edition (Java EE) - Die Java EE, „der große Bruder“ der Java SE, ist für den Einsatz im Serverbereich konzipiert und mit zusätzlichen APIs ausgestattet. Insbesondere hat sich diese Plattform für die Entwicklung zur Serverprogrammierung im Rahmen von Unternehmensanwendungen weitreichend etabliert.

Die Java ME Plattform soll in Zukunft von Sun eingestellt werden. Der Grund ist, dass mobile Endgeräte immer leistungsfähiger werden und somit eine speziell für diese Geräte angepasste Java Plattform unnötig wird. Die in diesem Buch behandelten Beispiele sind für die Java SE mit mindestens Version 5 bestimmt.

Die Sprache Java wird in dem vorliegenden Buch über weite Strecken eingesetzt. Die Sprache sowie die Architektur der Klassenbibliotheken war Vorreiter für andere Sprachen wie C# oder ActionScript 3.0 und den zugehörigen Klassenbibliotheken. Eine Einführung in die Programmiersprache Java ist in Kap. 3 wiedergegeben.

1.4.3.2 Die .Net-Plattform

In Konkurrenz zur Java-Plattform wurde von Microsoft die .Net-Plattform entwickelt. Sie unterstützt im Verhältnis zur Java-Plattform einen ähnlichen Umfang zur Entwicklung von Client-Server-Systemen. Zur Programmierung wird überwiegend die Sprache C# verwendet. Die Plattform hat eine ähnliche Ausrichtung, ähnliche Leistungsmerkmale und ähnliche Sprachkonstrukte wie die Java-Plattform.

1.4.3.3 Die FLEX/Flash -Plattform

Ähnlich wie bei der Java-Plattform werden mit dem Namen Flash unterschiedliche Werkzeuge und Komponenten zur Entwicklung von multimedialen Client-Server-Systemen bezeichnet. Die ersten Flash-Versionen wurden von dem Unternehmen Macromedia Ende der 1990er Jahre auf den Markt gebracht. Der Begriff Rich-Client wurde erstmals 2002 im Rahmen der Einführung von Flash-MX verwendet. Macromedia wurde im Jahr 2005 von Adobe Systems aufgekauft. Danach übernahm Adobe auch die Weiterentwicklung von Flash. Mit dem Namen Flash wird eine Reihe von sehr unterschiedlichen Begriffen assoziiert:

- Das Flash Autorenwerkzeug - wurde Ende der 1990er Jahre durch das Unternehmen Macromedia zur Entwicklung von Rich-Client-Systemen entwickelt. Die bereitgestellten Funktionalitäten gehen über die Standards SVG und SMIL hinaus. Der Begriff Rich-Client wurde von Macromedia seit 2002 im Rahmen der Einführung von Flash-MX maßgeblich mitgeprägt. Das Unternehmen Adobe hat 2005 Macromedia übernommen und seitdem die Entwicklung des Flash-Autorenwerkzeugs weitergeführt. Eine ähnlich leistungsstarke Entwicklungsum-

gebung existiert für die Standards SVG und SMIL nicht. Für SMIL wurde von Adobe bislang das Autorenwerkzeug GoLive herausgegeben. Die Weiterentwicklung wurde aber zugunsten von Dreamweaver (einem ähnlichen Autorenwerkzeug) eingestellt. Das Flash Autorenwerkzeug erzeugt Shockwave-Dateien (*.swf Format).

- Der Flash Player ist eine Ausführungsumgebung für Shockwave-Dateien. Dieser Player entspricht im Prinzip der Java Virtual Machine. Shockwave-Dateien können außer mit dem Autorenwerkzeug auch mit Hilfe eines Kompilers aus ActionScript- oder MXML-Quellcode Dateien erzeugt werden.
- FLEX - Die FLEX Plattform wird in Form eines Software Development Kit (FLEX SDK) als ein Open-Source Produkt zur Entwicklung von Rich-Clients zur Verfügung gestellt. Es umfasst die komplette ActionScript-Klassenbibliothek und den Kompiler zur Erzeugung von Shockwave-Dateien. FLEX-Anwendungen können ebenso mit dem FLEX-Builder, einer kostenpflichtigen auf Eclipse basierenden Entwicklungsumgebung, erstellt werden.

Bei dem Begriff ActionScript muss wie bei der Java-Plattform wieder zwischen 1. der Programmiersprache ActionScript (siehe Abschn. 4.4.3.4) und 2. einer ActionScript-Klassenbibiliothek unterschieden werden. Die ActionScript-Klassenbibliothek lässt sich grob in die folgenden zwei Packages unterteilen:

- Das ActionScript 3.0 `flash.*`- Package ist insbesondere zur Erzeugung von Vektorgrafiken und Animationen sowie für die Wiedergabe, das Streaming und die Synchronisation von Audio- und Videomedien zuständig. Der Leistungsumfang ist sehr groß und umfasst die Möglichkeiten der Standards SVG und SMIL.
- Das FLEX-`mx.*` -Package ist ebenfalls Teil der ActionScript Klassenbibliothek Die Klassen dieses Packages können in einfacher Weise durch die Tag-basierte Scriptsprache MXML zur Entwicklung von Nutzeroberflächen genutzt werden. Der Leistungsumfang kommt dem Java-`javax.swing.*`-Package nahe. Die Entwicklung von Nutzeroberflächen wird jedoch darüber hinaus stark vereinfacht. Insbesondere ist die an HTML erninnernde Syntax für viele Webentwickler besonderes einfach umzusetzen.

1.4.3.4 Microsoft Silverlight und JavaFX

Microsoft Silverlight und Sun's JavaFX sind ein neuer Zuschnitt der .Net- bzw. der Java-Plattform speziell für die Entwicklung von Rich-Client-Applikationen. Der jeweilige Aufbau ist an der FLEX/Flash Plattform orientiert. Typisch ist für alle Produkte eine an HTML orientierte Tag-basierte Möglichkeit, Oberflächen schnell entwickeln zu können. Bei Bedarf kann diese Vorgehensweise durch Programmcode ergänzt werden, welcher in den Sprachen C#, Java, JavaFX Script bzw. ActionScript 3.0 geschrieben ist. Um Silverlight oder JavaFX nutzen zu können, sind dann jedoch entsprechend aktuelle Versionen entsprechender Plugins erforderlich. Inwieweit sich Silverlight oder JavaFX in Zukunft durchsetzen werden, ist noch ungewiss.

1.4.4 Standards für die Client-Server-Interaktion

Zur Interaktion zwischen Client und Server müssen Protokolle zur Kommunikation eingehalten werden. Zwar ist für die Kommunikation zwischen Client und Server jede Netzwerktechnik denkbar, jedoch nicht immer praktisch. Zur Kommunikation zwischen Rechnersystemen haben sich mit der Entwicklung des Internets die Internet Protokollstandards durchgesetzt. Die unterschiedlichen Protokolle arbeiten dabei zusammen und sind, wie in Abb. 1.6 zu sehen, in Schichten angeordnet.

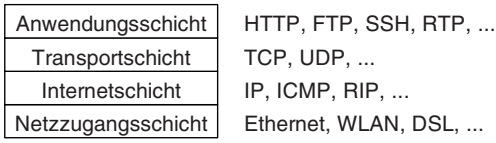


Abb. 1.6 Schichten der Internet Protokollstandards

Eine umfassendere Beschreibung der grundlegenden Protokolle IP, TCP, UDP und HTTP erfolgt zusammen mit Programmbeispielen in Kap. 3. Das für das Streaming wichtige RTP-Protokoll und entsprechende proprietäre Streamingprotokolle werden weiter unten vorgestellt.

1.4.4.1 IP-Protokoll

Mit Hilfe des Internetprotokolls (IP) ist eine Rechnerkommunikation über verschiedene Netzwerktechniken hinweg möglich. So kann z.B ein Server über eine Ethernet-Schnittstelle an das Internet angeschlossen sein und ein Client am heimischen Rechner über eine DSL-Schnittstelle. Das Internetprotokoll erlaubt nun über die Vergabe von weltweit eindeutigen IP-Adressen die Kommunikation zwischen diesen beiden Rechnern. Mit Hilfe von IP werden die zu versendenden Daten in Pakete verpackt, durch das Internet geleitet und an die Zielstation vermittelt. IP ist ein unsicheres Protokoll, d.h., die versendeten Daten können unterwegs verloren gehen oder dem Empfänger fehlerhaft zugestellt werden. Eine Fehlerkontrolle erfolgt nicht. IP ist von der Internet Engineering Taskforce (IETF) unter RFC 791 definiert worden.

1.4.4.2 TCP und UDP

Da IP nur die Adressierung von Rechnern im Internet erlaubt, ist dies für die Kommunikation zwischen entfernten Anwendungen nicht ausreichend. Es müssen ferner die kommunikationswilligen entfernten Anwendungen adressiert werden können. Dies geschieht über Portnummern. Einer im Internet kommunizierenden Anwendung wird in einem Rechnersystem eine eindeutige Portnummer zugewiesen. Ist

einer Anwendung die IP-Adresse sowie die Portnummer einer entfernten Anwendung bekannt (z.B. ein Web-Server mit Port 80), so kann sie mit dieser Anwendung Kontakt aufnehmen.

Das IP-Protokoll muss hierzu um Portnummern erweitert werden. Dies geschieht z.B. über die Protokolle TCP (Transmission Control Protocol) und UDP (User Datagram Protocol). Eine Anwendung versieht hierzu zunächst seine Daten mit einem TCP- oder UDP-Protokoll Header und verpackt diese Daten dann in ein IP-Paket. TCP stellt dabei einen verbindungsorientierten und sicheren Dienst zur Verfügung. Vor der Kommunikation muss eine Verbindung zwischen den beiden Stationen ausgehandelt werden. Verlorengegangene oder fehlerhafte Pakete werden automatisch beim Sender erneut angefordert. Desweiteren wird die Sendereihenfolge eingehalten, und die Daten erreichen den Empfänger in der gleichen Reihenfolge wie sie abgesendet wurden.

UDP hingegen stellt einen verbindungslosen und unsicheren Dienst zur Verfügung. Die Vorteile von UDP gegenüber TCP liegen in der fehlenden Verbindungsaufnahme und der fehlenden Fehlerkontrolle. Diese fehlenden Kontrollmechanismen erlauben geringere Verarbeitungs- und Wartezeiten auf Kosten von Fehleranfälligkeit. Die geringeren Latenzen erlauben den Einsatz von UDP in Echtzeitanwendungen wie z.B. beim Videostreaming, wo eine gewisse Fehlertoleranz akzeptiert werden kann. TCP ist von der IETF unter RFC 793 und UDP unter RFC 768 definiert.

1.4.4.3 Das HTTP-Protokoll

Das HTTP-Protokoll (Hypertext Transport Protokoll) ist ein Anwendungsprotokoll der Internet Protokollstandards, benutzt TCP als Transportprotokoll und verwendet normalerweise die Portnummer 80. Es ist eines der wichtigsten Kommunikationsprotokolle bei multimedialen Client-Server-Systemen. Über HTTP werden hauptsächlich Webseiten im HTML-Format von Web-Servern übertragen. HTTP ist jedoch nicht auf das Abrufen von HTML-Seiten beschränkt, ferner kann HTTP für jeden beliebigen Datentransfer zwischen Client und Server verwendet werden.

Über HTTP kann ein Client Daten an einen Server übertragen oder spezielle Ressourcen vom Server abfragen. Solche Ressourcen können Dateien wie z.B. multimediale Daten sein (Bilder, Videos, ...) oder auf dem Server befindliche Programme, welche beim Aufruf durch den Client auf dem Server ausgeführt werden und berechnete Ergebnisse (wie z.B. Datenbankzugriffe) an den Client zurücksenden. HTTP ist ein zustandsloses Protokoll, d.h., einzelne Anfragen an den Server auch von demselben Client werden als unabhängig voneinander betrachtet. Dies erschwert den Entwurf komplexer Anwendungen, da kein Zusammenhang zwischen den einzelnen Aktionen eines Clients hergestellt werden kann. Das Herstellen eines Zusammenhangs (z.B. über Sessions oder Cookies) ist dann Aufgabe der Anwendung. Das HTTP-Protokoll ist unter RFC 2616 von der IETF definiert worden.

1.4.5 Programmierschnittstellen für die Client-Serverinteraktion

Insbesondere bei den Programmierschnittstellen für die Client-Server-Interaktion gibt es zunächst die Vertreter, welche auf der Basis sehr universell eingesetzter Programmiersprachen wie Java oder C/C++ entwickelt worden sind. Für diese Sprachen gibt es mächtige Bibiliotheken, die sehr flexibel an unterschiedlichste Kommunikationsmodelle (z.B. eine synchrone oder asynchrone Kommunikation) angepasst werden können. Der Entwicklungsaufwand ist jedoch vergleichsweise hoch, so dass sich während der fortschreitenden Entwicklung des Internets für zentrale Anwendungsfälle ganz einfach zu programmierende Client-Server-Interaktionsmöglichkeiten herausgebildet haben. Die prominentesten Vertreter hiervon sind klassische traditionelle Anfragemöglichkeiten aus HTML-Seiten heraus über das

- Aktivieren von Hyperlinks oder das
- Abschicken eines Formulars und der anschließenden Anzeige einer neuen (auf dem Server dynamisch generierten) Seite.

In Abschn. 4.2.4 wird auf der Basis der Java-Programmierung dargestellt, wie umfangreich durch das Zusammenspiel von Browsersoftware und HTML die darunterliegenden Prozesse gekapselt werden. Selbst eine mittlerweile so selbstverständliche und trivial erscheinende Funktionalität wie das Aktivieren von Hyperlinks ist in einer universellen Programmiersprache sehr aufwändig zu programmieren. Jedoch bieten auch bei den universellen Programmiersprachen entsprechende Klassen die Möglichkeit, den Aufwand für derartige Implementierungen weitreichend zu reduzieren.

1.4.5.1 Asynchrones JavaScript und XML (AJAX)

AJAX ist ein Verbund der Technologien HTML/CSS und JavaScript und kann als das asynchrone Netzwerk-API zur Client-Server-Interaktion über HTTP angesehen werden. Die Nutzerfreundlichkeit herkömmlicher Web-Anwendungen war ohne AJAX sehr eingeschränkt. Ohne AJAX musste für jede neu vom Server abgerufene Information eine neue Seite geladen werden. Durch die synchrone Berabeitung war die Anwendnung blockiert, bis alle Daten vom Server eingetroffen waren. Erst in den letzten Jahren konnte durch die Einführung der asynchronen Kommunikation mit Hilfe von AJAX die Nutzerfreundlichkeit massiv verbessert werden. Eine Webseite wird dabei nur in relevanten Teilen durch aktualisierte Daten erneuert, und es kann während der Übertragung dieser Daten weitergearbeitet werden (asynchrone Übertragung). Die Gesamterscheinung und insbesondere zuvor ermittelte und dargestellte Informationen bleiben erhalten. Neu benötigte Informationen werden unbemerkt im Hintergrund nachgeladen, während der Nutzer weiterarbeitet.

1.4.5.2 Die Netzwerk APIs in den Java-, .Net- und FLEX/Flash-Plattformen

Die Java-Plattform besitzt umfangreiche Klassenbibliotheken, mit denen synchrone und asynchrone Client-Serverinteraktionen auf der Basis unterschiedlicher Protokolle wie UDP, TCP und HTTP umgesetzt werden können. Die Möglichkeiten übersteigen das Leistungsspektrum von AJAX bei weitem. Neben Low-Level Klassen, die direkt auf den Protokollen TCP und UDP aufsetzen (eine umfangreiche Einführung wird hierzu in Kap. 3 gegeben), kapseln weitere Klassen häufig wiederkehrender Spezialfälle (siehe z.B. hierzu Abschn. 4.2.4) insbesondere für die HTTP-Kommunikation. Ähnlich umfangreiche APIs für die Client-Serverinteraktion werden in der .Net und der FLEX/Flash-Plattform bereitgestellt.

1.4.6 Standards für die Verarbeitung von Audio- und Videomedien

Die Einführung von Standards für Bild-, Audio- und Videodaten ist mit einem sehr großen Forschungsaufwand verbunden, da hier anders als bei Dokumentenstandards (HTML, XML, ...) oder computergraphischen Standards (SVG, SMIL, VRML, ...) eine umfangreiche Verrechnung der vorliegenden Daten zur Kompression erfolgen muss. Es gilt dabei herauszufinden, wie ein Kompressionsverfahren aufgebaut sein muss, um effizient zu sein. Die wichtigsten Kompressionsverfahren werden ausführlich in Abschn. 5.1.1 behandelt.

1.4.6.1 Standards zur Kompression von Audio- und Videomedien

Die Standards zerfallen meist in mehrere Unterstandards (z.B. für die Audio-Kompression, Videokompression und die Containerformate), welche auch nicht immer zeitgleich veröffentlicht werden. Bei der Entwicklung von AV-Standards zur Datenkompression und für das Streaming sind die folgenden Meilensteine zu verzeichnen, auf genaue Jahresangaben wird im Folgenden verzichtet:

Ende der 80er Jahre /Anfang der 90er Jahre:

- JPEG Standard zur Kompression von Photografien
- H.261: Standard für Bildtelefonie und Videokonferenz über ISDN
- MPEG 1: Standard für Video auf CD
- mp3: Ein Teil des MPEG 1 Standards ist der sogenannte mp3 Audio-Standard, die volle Bezeichnung von mp3 lautet: MPEG 1 Layer 3 Audiostandard

Mitte der 90er Jahre:

- MPEG 2: Standard für Video- und Audioformate zur Verwendung in den Medien DVD (Digital Versatile Disc) und DVB (Digital Videobroadcasting) sowie für High-End-Formate im Studio- und Produktionsbereich.

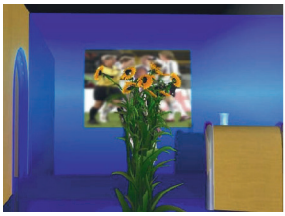
Ende der 90er Jahre:

- MPEG 4: Standard für Übertragungen über schmalbandige Kanäle (Internet, Mobilfunk) sowie für die Integration von Video in Webanwendungen. Derivate hiervon sind divX sowie die proprietären Formate der führenden Unternehmen Adobe (Flash On2 VP6), Microsoft (wmv).

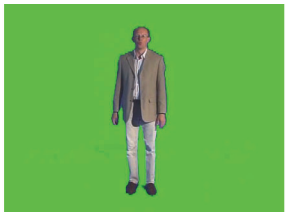
Seit 2000:

- Teil 10 des MPEG4 Standards / H.264: weitere Steigerung der Kompression.
- MPEG 4 Codierung von Video-Object-Planes (MPEG 4 VOP) und eXtensible MPEG 4 Textual Format (MPEG 4 XMT): MPEG 4 XMT ist ein XML-basierter Standard, der die Standards SMIL und SVG umfasst und die Integration von beliebig umrandeten Videoobjekten (Video-Object-Plane) in computergrafisch generierten Szenen unterstützt [4]. Beliebig umrandete Videoobjekte können durch Blue- oder Green-Screenaufnahmen und durch Verwendung eines alpha-Kanals erzeugt werden. Wenn z.B. eine Person vor einem grünen Hintergrund aufgenommen wird, so können die grünen Pixel transparent, d.h. unsichtbar gesetzt werden (siehe mittleres und rechtes Bild in Abb. 1.7). Dadurch kann die Person in computergrafisch generierte Szenen eingesetzt werden. Dies ist z.B. der Fall, wenn ein Nachrichtensprecher eine animierte Wetterkarte kommentiert und dabei mit der Hand auf spezielle Regionen zeigt. Natürlich können mit klassischen Compositing-Werkzeugen aus dem Produktionsbereich solche Effekte ermöglicht werden. Der MPEG 4 Standard hat zum Ziel, diese Technologie für beliebige multimediale Internetanwendungen in Echtzeit ohne weitere Werkzeuge (außer einem geeigneten MPEG 4-Player) zu ermöglichen. Darüber hinaus kann in dem Standard das Videoobjekt (also z.B. der Nichrichtensprecher) als Hot-Spot für Interaktionen fungieren (z.B. zur Einblendung von weiteren Zusatzinformationen). Anwendungsszenarien sind z.B. Videokonferenzen mit mehreren Teilnehmern, die sich alle in einem virtuellen Raum treffen und dort gemeinsam mit einem virtuellen Objekt oder Dokument interagieren können. Bislang existiert kein MPEG 4-Player, der dies standardkonform umsetzt. Der Flash-Player und die Verarbeitung von Videomedien auf der Basis der ActionScript-Klassen (Verwendung der `Netstream`-Klasse mit dem On2 VP6-Codec) ermöglichen als bislang einzige (proprietäre) Technologie diese Art der Präsentation (siehe Abschn. 5.3.3) und Interaktion. Die führende Position der Flash-Plattform in Be-

zug auf die Video- und Streamingtechnologie hat wesentlich zu ihrem Markterfolg beigetragen.



(a) Computergrapisch generiertes Nachrichtenstudio



(b) Green-Screen Aufnahme eines Nachrichtensprechers

(c) Einblendung des Nachrichtenstudios hinter dem Nachrichtensprecher durch Transparentsetzung der grünen Pixel

Abb. 1.7 Komposition von Video-Objekten mit MPEG 4

1.4.6.2 Codec- und Containerformate

Die genannten Kompressionsstandards zerfallen oft in mehrere Unterstandards. Der zentrale Aspekt der Datenkompression wird durch die Standardisierung eines Encoder/Decoder-Paares umgesetzt, welches oft auch als Codec (z.B. ein MPEG2-Codec, H.264-Codec) abgekürzt wird. Genau genommen wird nur der Dekodierungsvorgang standardisiert und das Verfahren zur Encodierung als Empfehlung veröffentlicht. Es kann durch diese Vorgehensweise weiterentwickelt und verbessert werden, unter der Bedingung, dass ein standardisierter Dekoder zu einer korrekten Wiedergabe der encodierten Daten in der Lage ist.

Neben der Standardisierung eines Codecs erfolgt ferner immer auch eine Standardisierung, in welcher Art- und Weise (z.B. in welcher Reihenfolge) die encodierten Daten in eine Datei abgespeichert bzw. über ein Netzwerk übertragen werden. Diese weitere (weniger forschungsintensive) Normierung führt zur Spezifikation von sog. Containerformaten. Bekannte Containerformate sind z.B. das Macromedia/Adobe flv-Format, Microsofts avi-Format, Apples Quicktime mov-Format und das etwas an Bedeutung verlohrene rm-Format von Realmedia.

Ein Containerformat zerfällt grob in einen Header und den Bereich der eigentlichen komprimierten Nutzdaten. Im Header werden zunächst allgemeine Eckdaten der Mediendatei gespeichert. Diese umfassen bei Videos die Länge des Videos, die Breite und die Höhe des Bildformats sowie den verwendeten Encoder. Containerformate an sich sagen also noch nichts über den verwendeten Codec aus. In vielen Veröffentlichungen werden etwas verwirrend Codec und Containerformat gleich bezeichnet. So gibt es einen JPEG-Codec, einen MPEG4-Codec und auch ein JPEG-jpg Containerformat und ein MPEG 4 mp4 Containerformat.

1.4.6.3 Progressiver Download und Streaming von Audio- und Videomedien

Bei der Übertragung von Daten über das Internet liegt in der Regel eine große Schwankung des Datendurchsatzes vor. Eine begrenzte Menge an Daten (z.B. ein Dokument) soll in der Regel möglichst schnell vom Server auf den Client übertragen werden. Kurzzeitig wird von der Anwendung ein großes Datenaufkommen erzeugt, welches danach wieder für eine gewisse Zeit vollständig verschwindet (Burst-Anwendung). Für die Übertragung wird meistens HTTP verwendet.

Bei der Übertragung von zeitbasierten Medien (insbesondere zur Übertragung von Live-Audio- und Videomedien) wäre im Gegensatz zu Burst-Anwendungen ein hinreichend großer und nahezu konstanter Datendurchsatz für die gesamte Netzauslastung wesentlich günstiger. Ein Videostrom einer Fernsehsendung braucht mehr oder weniger kontinuierlich 5MBit/s, um mit gleichbleibender Qualität (z.B. vergleichbar der gewohnten analogen Fernsehübertragung) übertragen werden zu können.

Für eine hohe Qualität von Streaminganwendungen wären daher Netze sinnvoll, welche eine Dienstgüte, insbesondere eine minimale Bandbreite garantieren (sogenannte Quality of Service Parameter). In großen Netzen wie dem Internet ist diese Forderung jedoch unrealistisch. Das Netz hat sich aus Burstanwendungen heraus entwickelt. Um den unterschiedlichen Anforderungen bei der Übertragung von Audio-Videomedien dennoch näher zu kommen, hat man spezielle Streaming-Protokolle eingeführt. Je nachdem, welches Protokoll genutzt wird, spricht man vom progressiven Download oder vom Streaming.

Progressiver Download Die verfügbare Datenrate und das zeitliche Eintreffen von Datenpaketen am Empfänger unterliegt mehr oder minder großen zeitlichen Schwankungen. Dies ist insbesondere dann der Fall, wenn die benötigte Datenrate größer ist als die Datenrate, die vom Netz zur Verfügung gestellt wird. Um zeitbasierte Medien dennoch kontinuierlich anzeigen zu können, ist eine Pufferung erforderlich. Je länger man puffert, um so länger muss ein Nutzer warten, bis das zeitbasierte Medium abgespielt werden kann. Bei einem Download über HTTP ist die Vorgehensweise somit dieselbe wie bei einem Download von anderen Dateien. Eine Audio-Videodatei wird auf dem Client-Rechner zwischengespeichert und von dort aus abgespielt. Dabei muss nicht gewartet werden, bis die komplette Mediendatei am Empfänger angekommen ist. Hierfür steht das Wort *progressive*: Am Empfänger wird nur solange gewartet, bis alle benötigten Daten beim Empfänger angekommen sind, um die ersten Bilder einer Sequenz abzuspielen. Die Übertragung von Live-Daten wird derzeit von keiner Technologie im progressiven Download unterstützt.

Streaming Es gibt im HTTP-Protokoll keine effizienten Mechanismen, die es erlauben, die Zeit, die zum Puffern erforderlich ist, zu minimieren. Man hat daher andere Protokolle eingeführt, die es ermöglichen diese Zeit zu verkürzen. Um schnell (möglichst in Echtzeit) die Datenpakete eines Bildes identifizieren, dekomprimieren und anzeigen zu können, ist ein Zeitstempel in den Datenpaketen hilfreich. Die Ergänzung von Datenpaketen der Transportschicht um

Zeitstempel ist daher ein wesentlicher Bestandteil der Streamingprotokolle wie z.B. RTP, RTMP und MMS, die in den JMF-, FLASH- bzw. DirectShow/Media Format-Klassen eingesetzt werden. Ein weiterer Mechanismus zur Optimierung der Übertragung von AV-Medien besteht darin, die vom Netz zur Verfügung gestellte Datenrate zu messen und eine daran angepasste Qualität der Audio-Videomedien vom Server abzurufen. Die Datenpakete werden daher z.B. bei dem RTMP-Protokoll mit unterschiedlichen Prioritäten versehen. Die Videopakete bekommen die geringste Priorität. Wenn Videopakete fehlen, stockt das Bild. Dies ist weniger störend als wenn die Audio-Verbindung unterbrochen wird. Bei einer Übertragung von Audio- und Videomedien über diese Streamingprotokolle spricht man von Streaming. Insbesondere die Übertragung von Live-Medien wird derzeit nur im Streaming-Verfahren unterstützt.

Die in den letzten Jahren zur Verfügung stehenden mittleren Bandbreiten sind z.B. bei einer DSL 6000-Verbindung jedoch so hoch, dass auch bei dem Progressive Download sich die Wartezeiten nicht mehr wesentlich vom Streaming unterscheiden und insgesamt Akzeptanz finden. Progressiver Download über HTTP wird verstärkt eingesetzt. Insbesondere brauchen hierfür keine kostenpflichtigen, speziellen Streaming-Server eingesetzt werden. Streaming-Protokolle werden nach wie vor für Live-Übertragungen verwendet.

1.4.6.4 RTP

In der Transportschicht des Internets können das UDP- oder das TCP Protokoll verwendet werden. TCP arbeitet verbindungsorientiert und unterstützt einen Kontrollmechanismus, der garantiert, dass alle versendeten Pakete auch beim Empfänger ankommen. Im Gegensatz zu TCP arbeitet UDP verbindungslos und ohne einen solchen Kontrollmechanismus. Dadurch werden Verarbeitungs- und Wartezeiten vermieden, welche durch die Nachforderung von verlorengegangen Paketen entstehen. Der Verlust von Datenpaketen, welche zu räumlich eingegrenzten Störungen eines vereinzelten Bildes führen, kann bei Streaming-Anwendungen bis zu einem gewissen Grad toleriert werden. Aus diesen Gründen setzt das Realtime Transport Protokoll (RTP) in den meisten Anwendungen auf UDP auf. Da UDP nicht einmal eine Sequenznummer zur richtigen Anordnung von Paketen nach einer Übertragung vergibt, wird in den Headern der RTP-Pakete neben einem Zeitstempel ebenfalls eine Sequenznummer eingetragen. Hinzukommen einige weitere Informationen, wie z.B. Angaben zu dem Sender und zu dem Codec, mit dem die Nutzdaten komprimiert wurden. Das UDP-Protokoll wird jedoch teilweise durch Firewalls blockiert. Um diese Schwierigkeit zu umgehen, kann auch das TCP-Protokoll verwendet werden. Der Standard RTP wurde 1996 erstmalig veröffentlicht und 2003 überarbeitet.

1.4.6.5 Flash RTMP

Analog zu RTP gibt es das Real Time Messaging Protocol (RTMP), ein von Adobe Systems entwickeltes proprietäres Netzwerkprotokoll, um Audio-, Video- und sonstige Daten über das Internet zwischem einem Medien-Server und einem Flash-Player streamen zu können. Die Standard-Version benutzt eine TCP-Verbindung. Während bei RTP nur ein Medientyp in einer RTP-Verbindung übertragen werden kann, können in einer RTMP-Übertragung mehrere Datentypen (z.B. Audio, Video- und Flex-Daten) kombiniert (gemultiplext) werden. Neben dem Zeitstempel wird im Header eine ID vergeben. Das Protokoll soll von Adobe im Laufe des Jahres 2009 veröffentlicht werden.

1.4.6.6 Microsoft MMS

Das Microsoft Media Server (MMS) Protokoll ist ein von Microsoft entwickeltes und veröffentlichtes Protokoll für multimediale Streaming-Anwendungen. Die Microssoft Media Format Bibliothek nutzt dieses Protokoll für Streaming-Anwendungen. Das Protokoll wird vom Client gestartet und baut zunächst eine TCP-Verbindung zum Server auf und überträgt seine IP-Adresse und den von ihm gewählten Port. Daraufhin erzeugt der Server eine UDP-Verbindung zum Client mit dem gewählten Port. Die Übertragung der Multimediadaten erfolgt dann über diese UDP-Verbindung (MMSU), während die TCP-Verbindung für Steuerungsbefehle genutzt wird. Es kann ebenfalls auch eine TCP-Verbindung gewählt werden. MMS wird in der Microsoft Media Format-Bibliothek für hardwarenahe Anwendungsentwicklungen zur Verfügung gestellt.

1.4.6.7 IPTV und InternetTV

Bei der Übertragung von Fernsehsignalen über das Internet werden zwei unterschiedliche Begriffe verwendet. Das sog. Internet-TV bzw. Web-TV bezeichnet die genannte Verbreitung von Fernsehsignalen über das Internet auf der Basis von progressivem Download bzw. von Streaming wie zuvor besprochen. Ein Beispiel hierfür ist z.B. die ZDFmediathek, eine Videothek über das Internet. Darüber hinaus gibt es noch das sog. IPTV, welches eher dem Digitalfernsehen DVB entspricht. Bei IPTV werden die Daten über ein spezielles Streamingprotokoll (vergleichbar RTP, RTMP, MMS) über das Internet mit einer garantierten Bandbreite und Qualität übertragen, welches nur über spezielle, geschlossene Verteilernetze gewährleistet werden kann. Als Beispiel für IPTV ist das „T-Home Entertain“ zu nennen, bei dem digitale Fernseinhalte nur über einen T-Home Internetanschluss zu empfangen sind.

1.4.7 Programmierschnittstellen für Audio- und Videomedien

Die Entwicklung der Programmierschnittstellen für Audio- und Videomedien ist einerseits auf der Basis von universellen Programmiersprachen wie Java und C/C++ erfolgt. Andererseits gibt es einige speziell auf diesen Anwendungsbereich zugeschnittene Technologien. Alle in Frage kommenden Technologien stellen leistungsstarke Codecs in entsprechenden Softwarebibliotheken bereit und sind daher stark von der Entwicklung der MPEG-Standards beeinflusst worden. Bei der Verwendung universeller Programmiersprachen ist eine Entwicklung für die unterschiedlichsten Einsatzbereiche möglich. Jedoch ist der Entwicklungsaufwand dann vergleichsweise hoch. Aus diesem Grund haben sich ebenso wie bei der Client-Server-Interaktion spezielle Lösungen für Standardanwendungsfälle herausgebildet. So kann schon durch ein spezielles HTML-Tag in vielen Browsern ein Standard MPEG 1-Video abgespielt werden. Wenn Live-Video, interaktives Video in Kombination mit anderen Rich-Client-Funktionalitäten, Video mit weiteren synchronisierten Medien sowie aktuellste Codecs eingesetzt werden sollen, so kommen die folgenden Technologien in Betracht:

Java-Media-Framework Für die Java-Plattform existiert mit dem Java Media Framework (JMF) eine Erweiterung zur Programmierung von Streaminganwendungen. Das JMF ist eine Programmierschnittstelle, welches die Entwicklung von Streaminganwendungen auf Windows und Unix/Linux-Systemen ermöglicht. Das JMF besitzt eine offene Architektur, so dass Weiterentwicklungen auch durch Fremdanbieter sehr gut integriert werden können. Die Einzelmodule, welche zur Verarbeitung und zum Streaming von Audio- und Videomedien erforderlich sind, sind weitreichend konfigurierbar. Leider werden von Sun nicht mehr die aktuellsten und leistungsstärksten Encoder und Decoder in das JMF integriert und auch die Weiterentwicklungen des JMF selbst nicht mehr gepflegt. Viele der integrierten Encoder und Decoder sind zudem nicht für alle Betriebssysteme (Windows, Unix/Linux) verfügbar, was eine plattformunabhängige Entwicklung erschwert. Jedoch werden diese Defizite durch Erweiterungen von Fremdanbietern gemindert. Das JMF unterstützt progressive Downloads ebenso wie das (Live-) Streaming von audio-visuellen Medien. Das JMF nutzt für Streaming-Übertragungen das RTP Protokoll. Die Komposition und Synchronisation von mehreren Medien stellt in der Java-Plattform zwar kein prinzipielles Problem dar, wird aber andererseits auch nicht durch besondere Klassen oder Werkzeuge unterstützt.

Microsoft DirectShow und Media Format bilden die APIs für die Verarbeitung und zum Streaming von Audio- und Videomedien im Rahmen der .Net-Plattform. Es verwendet wie das JMF eine offene Architektur, so dass es für Weiterentwicklungen durch Fremdanbieter attraktiv ist. Obwohl das .Net-Framework veraltete Windows-spezifische Softwarearchitektur-Konzepte wie das Component Object Model (COM) überwinden soll, ist dies jedoch für DirectShow nicht umgesetzt worden. Dadurch ist die Softwareentwicklung mit DirectShow kryptisch und umständlich. DirectShow wird überwiegend für eine sehr hardwarenahe Entwick-

lung von Audio-Videoanwendungen eingesetzt. Darüber hinaus basieren in vielen Fällen die APIs zur Programmierung von speziellen Graphik- und Videohardwarekomponenten für Windows-Plattformen auf DirectShow. Leistungsstarke Encoder und Decoder werden von DirectShow unterstützt. DirectShow und das Media Format API unterstützen progressive Downloads ebenso wie das (Live-) Streaming von audio-visuellen Medien. Für Streaming-Anwendungen wird das MMS-Protokoll verwendet. Ebenso wie im JMF gibt es keine besondere Unterstützung zur Komposition und Synchronisation von mehreren Medien.

FFMPEG In der freien Bibliothek libavcodec des FFMPEG-Projekts (www.ffmpeg.org) ist eine umfangreiche Liste von Encodern und Decodern implementiert worden. Die FFMPEG-Bibliothek libavformat stellt zudem eine Reihe von Containerformaten bereit. FFMPEG ist als Open Source Bibliothek für alle wichtigen Betriebssysteme verfügbar. Das empfehlenswerte Encoder/Decoder- und Streaming-Programm Video Lan Client (VLC, www.videolan.org) ist auf der Basis von FFMPEG entwickelt worden. Weitere Bibliotheken zur Entwicklung von Rich-Clients fehlen jedoch. Eine Integration in andere Plattformen ist damit unumgänglich und nicht ganz einfach. Eine Integration dieser Bibliotheken in das JMF befindet sich in der Entwicklung (JFFMPEG). FFMPEG unterstützt progressive Downloads ebenso wie das Streaming (z.B. über RTP und MMS) von audio-visuellen Medien. Es lassen sich mit den FFMPEG-Bibiliotheken für spezielle Zielgruppen multimediale Client-Serversysteme konzipieren.

RealMedia Der RealMedia Player unterstützt im Gegensatz zum JMF, DirectShow und FFMEPG wesentliche Teile des SMIL-Standards zur Komposition und Synchronisation mehrerer Medien. Darüber hinaus wird von RealMedia ein Streaming-Server angeboten. RealMedia unterstützt progressive Downloads ebenso wie das (Live-) Streaming (über RTP) von audio-visuellen Medien. Es lassen sich mit der RealMedia-Technologie durch Einbettung in die HTML/CSS/-JavaScript/AJAX-Technologie multimediale Client-Serversysteme aufbauen.

QuickTime Eine ähnliche Situation wie bei RealMedia liegt bei dem QuickTime Format von Apple vor. Von Apple wird für das (Live-) Streaming der QuickTime Streaming-Server angeboten, welcher jedoch nur auf Mac OS X Betriebssystemen lauffähig ist. Zudem gibt es noch eine Open Source Variante, der Darwin Streaming Server, welcher auch auf Windows und Unix/Linux Systemen lauffähig ist.

Flash Die FLEX/Flash-Plattform bietet mit dem `flash.*`-Package das derzeit leistungsstärkste API für die Verarbeitung und zum Streaming von Audio- und Videomedien. Die Funktionalitäten entsprechen dem SMIL-Standard sowie die Leistungsmerkmale der MPEG 4 VOP Codierung (On2 VP6-Codec) und die Integration der leistungsstärksten Encoder (H.264-Codec). Das Flash-API ist besonders einfach gehalten und auf die Standardanwendungsfälle bei der multimedialen Anwendungsentwicklung zugeschnitten. Dabei werden durch das Flash-Autorenwerkzeug umfangreiche Werkzeuge zur Komposition und Synchronisation mehrerer Medien bereitgestellt. Jedoch ist das Flash-API kaum er-

weiterbar durch Fremdanbieter, sondern ist speziell auf die multimediale Webentwicklung innerhalb der FLEX/Flash-Plattform ausgerichtet. Flash unterstützt progressive Downloads ebenso wie das (Live-) Streaming (über RTMP) von audio-visuellen Medien. Hierfür wird ein leistungsstarker Streaming-Server (Flash Media Server) angeboten. Während mit JMF und DirectShow serverseitige Streaming-Module auf der Basis von RTP bzw. MMS auch für Live-Streaming-Anwendungen entwickelt werden können, ist man im Rahmen der FLEX/Flash-Plattform bei der Nutzung von RTMP auf den Einsatz des Flash Media Servers angewiesen.

1.4.8 Programmierschnittstellen zur Entwicklung von Serversystemen

Die Anwendungs- und Datenhaltungsschichten (siehe Kap. 2.2) bilden in dem vorliegenden Buch außer im Grundlagenbereich keinen besonderen Schwerpunkt, da hier nur selten multimediale Aspekte zu berücksichtigen sind. Im Vordergrund stehen in diesen Schichten die Unterstützung von Geschäftsprozessen in Unternehmen und Verwaltungen. Für eine Entscheidung, welche Technologie zu bevorzugen ist, werden jedoch die Unterstützung dieser Schichten auch im Rahmen multimedialer Client-Server-Systeme eine große Rolle spielen. Daher werden im Folgenden die wichtigsten Programmierschnittstellen für diese Schichten kurz beschrieben:

Java EE, Java-Servlets, JSP In der umfangreichsten Java-Plattform, der Java EE (Java Enterprise Edition), werden sogenannte Servlet-Klassen bereitgestellt. Servlet ist ein Kunstwort, das aus den Worten Server und Applet zusammen gesetzt ist und kleine Mini-Programme meint, welche auf einem Server laufen. Servlets können dynamisch HTML-Code generieren (z.B. können in Abhängigkeit der Eingabedaten des Nutzers Datenbankabfragen erzeugt werden), welcher dann per HTTP an den Client übertragen wird. Die Technologie der Java Server Pages (JSP) stellt eine Erweiterung von Servlets dar. Eine JSP-Seite ist eine normale HTML-Seite (siehe Abschn. 4.3), in die spezielle JSP-Tags eingebettet sind. Diese Tags umschließen Java-Code, der auf dem Server ausgeführt wird und durch die Servlettechnologie dann dynamisch generierten HTML-Code an den Client überträgt.

Microsoft Windows Server, ASP Microsoft stellt für die .Net-Plattform ebenfalls leistungsfähige Server-APIs zur Verfügung. Bei den Active Server Pages (ASP) handelt es sich um Skripte in der Sprache VBScript. Die Technologie ist mit Java-JSP vergleichbar. Der Windows Server fungiert sowohl als HTTP-Server als auch ebenfalls als MMS-Streamingserver.

PHP Ebenfalls mit der JSP-Technologie vergleichbar ist PHP (PHP Hypertext Preprocessor). Auch hier werden Kommandos in HTML-Seiten eingebettet, die dann auf dem Server ausgeführt werden. PHP ist eine Open Source Entwicklung.

Die herausragenden Eigenschaften von PHP sind die zahlreichen Datenbank-Schnittstellen (nahezu jede Datenbank kann angesprochen werden) sowie die leichte Erlernbarkeit. Zusätzlich gibt es umfangreiche Funktionsbibliotheken, für nahezu jeden Anwendungsfall existieren vorgefertigte Bibliotheken.

1.5 Vergleich der Technologien zur Entwicklung multimedialer Client-Server Systeme

In diesem Abschnitt soll ein kurzes Resümee zum Einsatz der genannten Technologien gezogen werden. Zunächst werden einige Kriterien aufgezeigt, aufgrund derer eine Abgrenzung der Technologien untereinander möglich ist. In den letzten Abschnitten wurden Technologien vorgestellt, welche eine Unterstützung

- der Entwicklung von Rich-Clients
- der Client-Server-Interaktion
- der Verarbeitung und dem Streaming von Audio- und Videomedien
- der Entwicklung von Serversystemen

ermöglichen. Wenn die gewünschte Zielgruppe für eine geplante Anwendung nicht stark eingegrenzt werden kann (was bei den meisten Web-Anwendungen der Fall ist), dann kommt dem

- Verbreitungsgrad auf Clientrechnern

ebenfalls eine wesentliche Bedeutung zu. Mit Verbreitungsgrad ist gemeint, wie weit eine jeweilige Technologie ohne weitere Installationen von Plattformen oder Plugins auf potenziellen Clientrechnern einsatzbereit ist. Zusätzliche Installationen führen bei vielen Nutzern dazu, dass diese sich nicht weiter mit einer Webseite beschäftigen wollen. Der Verbreitungsgrad von Silverlight, JavaFX, FFMPEG, RealMedia und QuickTime ist derzeit vergleichsweise gering, sodass diese Technologien im Speziellen in diesem Buch nicht weiter betrachten werden. Doch insbesondere für Silverlight und JavaFX kann sich dies innerhalb weniger Jahre ändern.

Folgende Technologiebündel kommen aufgrund der o.g. Kriterien in die engere Wahl:

AJAX (HTML, CSS und JavaScript) mit PHP, JSP oder ASP Dieses Technologiebündel bildet den einzigen Vertreter, der bei den Client-Technologien ausschließlich auf Standards setzt, welche auf über 99 % aller Client-Systeme verfügbar sind. Eine mit diesem Technologiebündel entwickelte Web-Anwendung kann daher auf fast allen Clientsystemen ohne weiteres in vollem Umfang ausgeführt werden.

Die Java- oder die .Net-Plattform Diese Plattformen bieten eine sehr gute Unterstützung bei der Entwicklung von Serversystemen sowie bei der Client-Server-Interaktion und haben aus diesem Grund in Unternehmen eine hohe Akzeptanz gefunden. Darüber hinaus existieren äußerst umfangreiche Produktpaletten mit

leistungsstarken Entwicklungswerkzeugen (z.B. NetBeans und Eclipse) der unterschiedlichsten Hersteller. Als Vertreter dieser Technologieausrichtung werden wir in dem vorliegenden Buch die Java-Plattform anwenden.

Die FLEX/Flash-Plattform in Kombination PHP, JSP oder ASP Dieses Technologiebündel bietet eine leistungsstarke Unterstützung für die Verarbeitung von Audio- und Videomedien. Neben der sehr gut zur Medienverarbeitung geeigneten ActionScript-Klassen-Bibliothek der Flash-Plattform wird von Adobe eine große Produktpalette mit Werkzeugen zur Entwicklung von Medien und Rich-Clients angeboten. Angefangen mit der Medienproduktion z.B. mit Photoshop, Premiere und AfterEffects, über Werkzeuge zur internetgerechten Kompression der Medien, wie dem Flash Media Live Encoder, über die Webentwicklung mit Flash/Flex bis hin zur Veröffentlichung aller Komponenten mit dem Flash-Media-Server werden alle Entwicklungsbereiche abgedeckt.

Neben diesem Überblick, der sich an den herausragenden Stärken der jeweiligen Technologien orientiert, muss jedoch hinsichtlich einiger Kriterien etwas weiter differenziert werden.

1.5.1 AJAX in Kombination mit PHP, JSP oder ASP

Unterstützung der Entwicklung von Rich-Clients Der Aufwand attraktive Nutzeroberflächen zu entwickeln ist im Vergleich zu anderen Technologien recht groß, da HTML nur einfache Komponenten zur Generierung von Nutzeroberflächen zur Verfügung stellt. Zudem gibt es hinsichtlich CSS und JavaScript einige Brwoser-Inkompatibilitäten, sodass je nach Browser unterschiedliche Implementierungen benötigt werden. Um ansprechende, interaktive Bedienelemente zu erzeugen (z.B. Drag- und Drop-Funktionalitäten) muss mit Hilfe von JavaScript und CSS ein beträchtlicher Aufwand betrieben werden. Der Entwicklungsaufwand lässt sich mit Hilfe von JavaScript- oder AJAX-Frameworks erheblich verringern, weshalb der Einsatz eines solchen Frameworks zu empfehlen ist. Wenn eine Web-Anwendung mit AJAX (HTML, CSS und JavaScript) auskommen kann, so fällt derzeit und wohl auch mittelfristig die Entscheidung zu Gunsten dieser Technologie, da hierdurch die Installation weiterer Player oder Plugins beim Anwender vermieden werden kann. Jedoch müssen mit steigender Komplexität der Bedienelemente der Web-Anwendungen und der damit verbundenen aufwändigeren AJAX-Frameworks auch größere Ladezeiten berücksichtigt werden. Das AJAX-Framework muss zunächst beim Betreten der Web-Seite auf den Client übertragen werden. Dies geschieht auch, wenn sich der Anwender zu einer anderen Web-Seite bewegt, welche das zuvor geladene AJAX-Framework ebenfalls benutzt. Aufgrund dieser zusätzlichen Wartezeiten beim Betreten einer Seite können die Grenzen der Akzeptanz beim Nutzer überschritten werden

Unterstützung für Audio- und Videomedien Leistungsstark komprimierte zeitbasierte Medien wie Audio und Video und anspruchsvollere Animationen kön-

nen mit dieser Technologie nicht wiedergegeben werden. Die Installation eines weiteren Players oder Plugins ist dann unverzichtbar. Zur Zeit ist die Verwendung des Flash-Plugins naheliegend, da dieses für Streaming-Anwendungen den mit Abstand größten Verbreitungsgrad und das größte Leistungsspektrum aufweist. Wenn jedoch der Flash-Player ohnehin installiert werden muss, stellt sich die Frage, ob nicht weitere Teile des Rich-Clients mit der FLEX/Flash-Plattform entwickelt werden sollten.

1.5.2 Die Java-Plattform

Verbreitungsgrad auf Clientrechnern Bislang sind mit der Java-Plattform vorwiegend Thin-Client Systeme auf der Basis von JSP und Java Server Faces (JSF) entwickelt worden. JSF bilden ein serverseitiges Komponentenmodell, welches Klassen für komplexe Nutzeroberflächenelemente wie in dem Paket `javax.swing.*` bereitstellt. Eine clientseitige Programmierung mit JavaScript kann dann durch eine leistungsstärkere Java-Programmierung ersetzt werden. Für die Entwicklung von Rich-Clients können innerhalb der Java-Plattform nun zwei Wege beschritten werden:

1. Man verwendet weiterhin die JSP- und JSF-Technologie und zusätzlich ein an diese Technologien angepasstes AJAX-Framework. Wie auch bei dem HTML-, CSS- und JavaScript-Ansatz wird durch AJAX die Nutzerfreundlichkeit wesentlich verbessert. Der Verbreitungsgrad auf Clientrechnern liegt dann bei über 99%. Jedoch ist damit das Problem der Integration von Audio- und Videomedien nicht gelöst, und ebenso existiert der Nachteil von hohen Ladezeiten durch das AJAX-Framework.
2. Man verwendet einen Java-Applet-basierten Fat- bzw. Rich-Client-Ansatz. Dieses setzt eine installierte Java Laufzeitumgebung (Java Runtime Environment, JRE) auf dem Client voraus. Die Entwicklung von Fat/Rich-Client-Systemen mit Applets eröffnet dann die gesamte Fülle der Möglichkeiten der J2SE-Plattform. Ebenso kann die neuere JavaFX-Technologie als ein Applet auf dem Client ausgeführt werden. Die Verbreitung von Applets ist anfänglich nicht in dem Maße fortgeschritten, wie es erwartet wurde. Ein maßgeblicher Grund hierfür ist, dass das JRE in der benötigten Fassung nicht von allen Browsern unterstützt wird bzw. nicht aktiviert ist. Über den Verbreitungsgrad liegen nur sehr unterschiedliche und unverlässliche Zahlen vor (50% bis 90%), die auch hinsichtlich der Aktualität der entsprechenden Version unterschieden werden müssen. Der Verbeitungsgrad scheint jedoch zuzunehmen und so werden auch in neueren Projekten (z.B. für graphische Konfiguratoren auf Webseiten der Automobilindustrie) wieder stärker Applets eingesetzt.

Unterstützung der Entwicklung von Rich-Clients Die Funktionalitäten sind bei der Verwendung auch von älteren JRE sehr umfangreich und mit dem Leis-

tungsspektrum von FLEX/Flash vergleichbar, solange keine Audio- und Videomedien integriert werden müssen. Für die Integration von Audio- und Videomedien ist entweder die zusätzliche Installation des JMF, eines JavaFX-Plugins oder einer sehr aktuellen JRE, welche auch JavaFX in Applets unterstützt, erforderlich. Der Verbreitungsgrad solch aktueller Plugins bzw. JREs ist jedoch gering. Die Flash-Plattform bietet dann ein größeres Leistungsspektrum und auch einen höheren Verbreitungsgrad.

1.5.3 Die FLEX / Flash Plattform

Verbreitungsgrad auf Clientrechnern Der Nachteil gegenüber AJAX (HTML, CSS und JavaScript) ist natürlich in der zusätzlichen Installation des Flash-Players zu sehen. Über den Verbreitungsgrad liegen wie für das Java Runtime Environment sehr unterschiedliche Zahlen vor, die je nach Quelle und Aktualität der Player-Version zwischen 50% bis über 90% variieren.

Literaturverzeichnis

1. Bultermann D C A, Rutledge L (2004) SMIL 2.0 Interactive Multimedia for Web and Mobile Devices. Springer
2. Eberleh E, Oberquelle H, Oppermann R (1994) Einführung in die Software-Ergonomie. de-Gruyter
3. Ohm R (1995) Digitale Bildcodierung. Springer
4. Pereira F, Ebrahimi T (2002) The MPEG 4 Book. Prentice-Hall
5. Plag F, Riempp R, (2007) Interaktives Video im Internet mit Flash. Springer