LEHRSTUHL FÜR
ALLG. BWL UND WIRTSCHAFTSINFORMATIK
UNIV.-PROF. DR. HERBERT KARGL

Schwickert, Axel C.; Dandl, Jörg

HTML, Java, ActiveX – Strukturen und Zusammenhänge

ARBEITSPAPIERE WI
Nr. 6/1997

Schriftleitung:
Dr. rer. pol. Axel C. Schwickert

Information

Reihe: Arbeitspapiere WI

Herausgeber: Univ.-Prof. Dr. Axel C. Schwickert
Professur für BWL und Wirtschaftsinformatik
Justus-Liebig-Universität Gießen
Fachbereich Wirtschaftswissenschaften
Licher Straße 70
D – 35394 Gießen
Telefon (0 64 1) 99-22611
Telefax (0 64 1) 99-22619
eMail: Axel.Schwickert@wirtschaft.uni-giessen.de
http://wi.uni-giessen.de

Bis Ende des Jahres 2000 lag die Herausgeberschaft bei:

Lehrstuhl für Allg. BWL und Wirtschaftsinformatik
Johannes Gutenberg-Universität Mainz
Fachbereich Rechts- und Wirtschaftswissenschaften
Welderweg 9
D - 55099 Mainz

Ziele: Die Arbeitspapiere dieser Reihe sollen konsistente Überblicke zu den Grundlagen der Wirtschaftsinformatik geben und sich mit speziellen Themenbereichen tiefergehend befassen. Ziel ist die verständliche Vermittlung theoretischer Grundlagen und deren Transfer in praxisorientiertes Wissen.

Zielgruppen: Als Zielgruppen sehen wir Forschende, Lehrende und Lernende in der Disziplin Wirtschaftsinformatik sowie das IuK-Management und Praktiker in Unternehmen.

Quellen: Die Arbeitspapiere entstanden aus Forschungsarbeiten, Diplom-, Studien- und Projektarbeiten sowie Begleitmaterialien zu Lehr- und Vortragsveranstaltungen des Lehrstuhls für Allg. Betriebswirtschaftslehre und Wirtschaftsinformatik Univ. Prof. Dr. Herbert Kargl an der Johannes Gutenberg-Universität Mainz.

Hinweise: Wir nehmen Ihre Anregungen und Kritik zu den Arbeitspapieren aufmerksam zur Kenntnis und werden uns auf Wunsch mit Ihnen in Verbindung setzen.
Falls Sie selbst ein Arbeitspapier in der Reihe veröffentlichen möchten, nehmen Sie bitte mit dem Herausgeber (Gießen) unter obiger Adresse Kontakt auf.
Informationen über die bisher erschienenen Arbeitspapiere dieser Reihe und deren Bezug erhalten Sie auf dem Schlußblatt eines jeden Arbeitspapiers und auf der Web Site des Lehrstuhls unter der Adresse http://wi.uni-giessen.de

Arbeitspapiere WI Nr. 6/1997

Autoren: Schwickert, Axel C.; Dandl, Jörg

Titel: HTML, Java , ActiveX – Strukturen und Zusammenhänge

Zitation: Schwickert, Axel C.; Dandl, Jörg: HTML, Java, ActiveX – Strukturen und Zusammenhänge, in: Arbeitspapiere WI, Nr. 6/1997, Hrsg.: Lehrstuhl für Allg. BWL und Wirtschaftsinformatik, Johannes Gutenberg-Universität: Mainz 1997.

Kurzfassung: Das World Wide Web (WWW) ist als Hypermedia-System konzipiert; es stellt textuelle, graphische und auditive Informationen sowie Interaktionsmöglichkeiten auf WWW-Seiten zur Verfügung, auf die über graphische Benutzeroberflächen, sogenannte WWW-Browser, zugegriffen wird. Die Bereitstellung der Hypermedia- und Interaktionsfunktionalitäten im Browser erfolgt mit HTML, Java, JavaScript, AvtiveX oder Kombinationen dieser Werkzeuge. Die Auszeichungssprache HTML (Hypertext Markup Language) kennzeichnet die Struktureigenschaften eines WWW-Dokumentes sowie darin enthaltene Verweise auf andere Dokumente im WWW. Aufgrund ihrer Fähigkeiten, die erweiterten Kommunikations- und Hypermediafunktionalitäten von CGI, JavaScript, Java und ActiveX zu integrieren, werden HTML-Umgebungen zukünftig verstärkt als Organisationszentren für WWW-Projekte fungieren. Java als Alternative oder Supplement zu HTML im WWW ist der hochaktuelle Nebenschauplatz einer objektorientierten Programmiersprache, die zur Entwicklung beliebiger Software eingesetzt werden kann. ActiveX überträgt Microsofts OLE-Modell auf das Internet und ermöglicht die verteilte Datenhaltung und Interoperabilität zwischen Applikationen auf unterschiedlichen Rechnerplattformen. Das vorliegende Arbeitspapier stellt die genannten Gestaltungswerkzeuge mit ihrer Struktur, Funktionsweise und gegenseitigen Anknüpfungspunkten vor.

Schlüsselwörter: HTML, Web Publishing, CGI, Image Map, Plug in, Java, Applications, Applets, Java Virtual Machine, JavaScript, Sicherheitsmodell, AvtiveX, COM, OLE, VBX, OCX, ActiveX Controls, Code Signing, Trident

Inhaltsverzeichnis

1 HTML – Hypertext Markup Language

1.1 World Wide Web, Browser und HTML

Das World Wide Web (WWW) ist als Hypermedia-System konzipiert; es stellt textuelle, graphische und auditive Informationen sowie Interaktionsmöglichkeiten auf WWW-Seiten zur Verfügung, auf die über graphische Benutzeroberflächen, sogenannte WWW-Browser, zugegriffen wird. Die Bereitstellung der Hypermedia- und Interaktionsfunktionalitäten im Browser erfolgt mit Hilfe der Auszeichnungssprache **HTML (Hypertext Markup Language)**.

HTML wurde auf Grundlage von **SGML (Standard Generalized Markup Language)** durch eine Initiative von Tim Berners-Lee, dem Begründer des WWW, entwickelt. Aus dieser Inititative ging 1994 das **"W3-Consortium" (W3C)** hervor; alle bedeutenden Unternehmen der IT-Branche von Adobe, Apple über IBM, Intel und Microsoft bis Oracle, Xerox u. v. m. sind darin vertreten. Seit seiner Gründung zeichnet dieses Industriekonsortium für die weltweite Normung von HTML verantwortlich. Im Mai 1996 wurde die aktuelle Version HTML 3.2 vom W3C verabschiedet.

1.2 Struktureigenschaften und Tags

HTML kennzeichnet die Struktureigenschaften eines WWW-Dokumentes und darin enthaltene Verweise (**Links**) auf andere Dokumente im WWW. Die Struktureigenschaften betreffen z. B. Absätze, Titel, Listen und Überschriften, nicht jedoch exakte Schriftarten, -größen oder Einrückungen der Bildschirmdarstellung. Das letztlich für den Betrachter sichtbare Layout eines Dokumentes ist von den graphischen Darstellungsmöglichkeiten des Browsers abhängig. Für die Übertragungsgeschwindigkeit im WWW ist es entscheidend, daß der Umfang "roher" HTML-Dokumente mit eingebetteten Formatierungsanweisungen deutlich geringer ist als der Umfang komplett formatierter und im Layout fertiger Dokumente.

HTML-Dokumente enthalten den Text des Dokumentes selbst und die HTML-Auszeichnungen, welche die Struktur, die Formatierung des Dokumentes und die Verweise auf andere Dokumente oder eingefügte Medien kennzeichnen. HTML-Auszeichnungen **(Tags)** haben in der Regel eine Anfangs- und eine Ende-Kennung, die den zu beeinflussenden Darstellungsinhalt umrahmen.

Eine HTML-Datei besteht aus zwei Teilen: der Kopfteil wird vom **Head-Tag**, der Rumpfteil vom **Body-Tag** umfaßt (vgl. Abb. 1). Der Kopfteil enthält allgemeine Informationen über das Dokument, wie bspw. dessen Titel. Im Rumpfteil befindet sich der gesamte Dokumentinhalt mit seinen Auszeichnungen für Struktur und Verweise.

Tags für die Einbindung von Verweisen (**Links**) auf andere WWW-Seiten sind bezeichnend für den Hyperlink-Charakter des WWW. Ein Tag vom Typ <A HREF="URL"> kann auf eine beliebige Seite im gesamten WWW verweisen, die über einen URL (Unique Resource Locator) erreichbar ist. Wählt der Anwender auf seiner Browser-

Oberfläche einen solchen Link an, wird die adressierte WWW-Seite abgerufen und angezeigt. Auf gleiche Weise lassen sich mit dem **Img-Src-Tag** einzelne Graphiken adressieren und anzeigen (Abb. 2 zeigt das Ergebnis des HTML-Code-Beispiels aus Abb. 1)

```
<HTML>
<HEAD>
<TITLE>HTML-Beispiel</TITLE>
</HEAD>
<BODY>
<CENTER><P><B>&Uuml;berschrift: fett, zentriert
</B></P><CENTER>
<P><IMG SRC="http://wiwi.uni-mainz.de/pics/hand.gif">
<A HREF="http://wiwi.uni-mainz.de">
  Link zur Uni-Mainz</A></P>
</BODY>
</HTML>
```

Abb. 1: HTML-Code-Beispiel

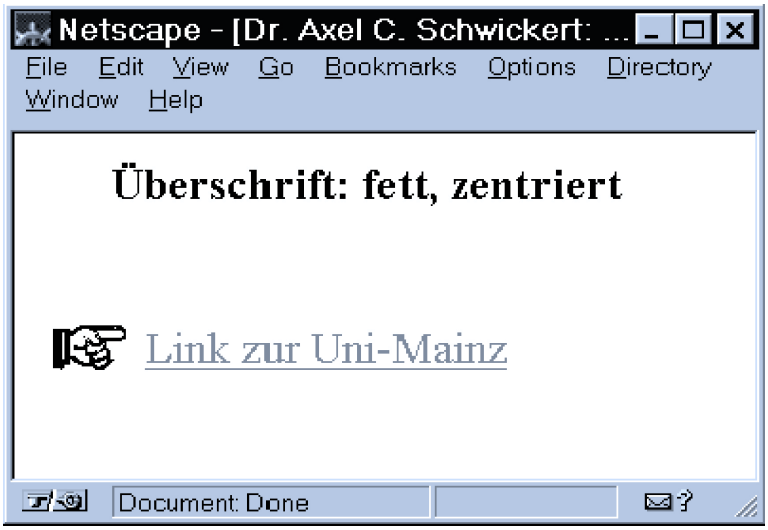

Abb. 2: Ergebnis des HTML-Code-Beispiels

Die wichtigsten Strukturierungsanweisungen für Text sind Tags für Absätze, Überschriften mehrerer Untergliederungsstufen, Listen, Textausrichtung und Zeichenformatierung. Die Kennzeichnung einer Zeichenkette als Block gewährleistet die durchgehend einheitliche Darstellung des betreffenden Abschnittes. Da HTML im ASCII-Format geschrieben wird, müssen für die Darstellung von deutschen Umlauten spezielle Codes in ein kaufmännisches Und (&) und ein Semikolon (;) gefaßt werden (z. B. "ü" für ein "ü" und "Ü" für ein "Ü").

Standardisierte Tags für Farb-, Schriftart- und Hintergrundgestaltungen, das Umfließen von Bildern mit Text, die transparente Überlappung von Anzeigeelementen sowie die Darstellung von Text und Graphiken in Tabellen geben HTML-Autoren einen breiten Gestaltungsspielraum für das Layout ihrer Dokumente. Die Browser von **Netscape und Microsoft** erkennen und handhaben jedoch auch herstellerspezifische HTML-Tags, die (noch) nicht vom W3-Consortium als Standard verabschiedet sind. Mit der **Frame-Technik** von Netscape und Microsoft lassen sich z. B. WWW-Seiten in mehrere Fenster unterteilen. Die verwendeten Frame-Tags gehören nicht zum aktuellen HTML-Standard, werden in der Praxis jedoch häufig zur Grobstrukturierung von WWW-Seiten genutzt.

Eigenentwickelte HTML-Dokumente sollten grundsätzlich offline in verschiedenen Browsern getestet werden, bevor sie online im WWW zur Verfügung gestellt werden. Für die Entwicklung von HTML-Dokumenten bieten sich **Web-Publishing-Pakete** an, die einen Browser zum Testen, einen HTML-Editor zum Handcoding und hochproduktive WYSIWYG-Werkzeuge enthalten. Die führenden Produkte sind als zeitlich begrenzte Testversionen, Shareware oder Freeware bei den Herstellern downloadbar (Microsoft Frontpage, Adobe Pagemill, Softquad Hot Metal, Corel Web Designer, Asymetrix Web Publisher, Netscape Navigator Gold, Claris Homepage, Netobjects Fusion). Mit den WYSIWYG-Werkzeugen lassen sich relativ schnell WWW-Seiten produzieren: Assistenten unterstützen ein konsistentes Seiten-Design, Formatvorlagen strukturieren Seiteninhalte, Graphiken werden per Drag und Drop plaziert. Der HTML-Code solcher Aktionen wird automatisch generiert. Über die Gegenprüfung dieses Codes im HTML-Editor kann sich auch der Einsteiger an das Handcoding von HTML für Feinarbeiten herantasten.

1.3 CGI, Image Maps und Plug ins

HTML ab der Version 2 ermöglicht die Interaktion zwischen dem Anbieter der WWW-Seite und dem Anwender über Formulare, die der lokale Browser auf Anweisung des **Form-Tags** am Bildschirm aufbaut. Für den Input des Anwenders stehen Eingabe- und Auswahlfelder sowie Schaltflächen zur Verfügung. Mit dem an den Form-Tag gebundenen Verknüpfungsmechanismus **CGI (Common Gateway Interface)** werden Formulareingaben im lokalen Browser als Argumente an ein Programm des entfernten Servers übergeben, das Programm dort ausgeführt und dessen Output an den lokalen Browser zur Darstellung zurückgegeben. Das Server-Programm kann die Formulareingaben z. B. für eine Datenbankabfrage verwenden und das Abfrageergebnis in HTML codiert an den Browser zur Anzeige senden (z. B. Search Engines, Börsenkurse). Server-Programme schreiben z. B. auch Formulareingaben von Online-Bestellungen in eine Datenbank des Produktanbieters, veranlassen eine bestätigende eMail an den Besteller sowie den Versand der bestellten Produkte. Grundsätzlich können auf dem entfernten Server beliebige Programme gestartet werden, die dort für den Zugriff per CGI vorgehalten werden. Für WWW-Interaktionen wie z. B. Gästebücher, Zugriffzähler, eMail-Versand etc. kursieren frei erhältliche CGI-Programme im WWW. CGI-Software mit speziellen Funktionen muß selbst programmiert oder gekauft werden. CGI-Pro-

gramme können mit jeder auf dem Server lauffähigen Programmiersprache erstellt werden; häufig verwendete Sprachen sind PERL, C und C++.

Ab der HTML-Version 2 lassen sich mit dem Img-Src-Tag auch **Image Maps** in HTML-Dokumente einbetten. Eine Image Map ist eine sensitive Graphik, die in verschiedenen Bereichen des Bildes mit unterschiedlichen Aktionen (meist Links) hinterlegt ist. In einem als Image Map realisierten Stadtplan können z. B. per Mauszeiger einzelne Statteile angewählt werden, zu denen jeweils eine eigene Graphikdatei mit einer Ausschnittsvergrößerung angezeigt wird. Server-seitige Image Maps funktionieren strukturell wie CGI-Anwendungen: der Server liefert die Image Map an den Browser, die Koordinaten des Mauszeigers (Benutzer-Input) werden als Argumente an die Image-Map-Anwendung des Servers zurückgegeben, der Server veranlaßt die entsprechende Aktion im Browser (Server-Output). Bei client-seitigen Image Maps liefert der Server zur sensitiven Graphik direkt auch die Aktionsanweisungen (plus evtl. zugehörige Darstellungselemente) zu den einzelnen Koordinatenbereichen der Graphik mit.

Plug-ins als client-seitige Zusatzprogramme zum Browser dienen der Präsentation von in WWW-Seiten eingebetteten Multimedia-Elementen, deren Formate nicht mit Browser-eigenen Funktionen handhabbar sind (Video, Audio, spezielle graphische Formate). Über die Anbieter von speziellen Multimedia-Elementen sind die notwendigen Plug-ins meist frei erhältlich. Bei der Installation tragen sich die Plug-ins selbst in eine Liste ein, die dem Browser signalisiert, welches Zusatzprogramm bei welchem Multimediaformat aufgerufen werden soll.

1.4 HTML, JavaScript und Java

JavaScript wurde von Netscape als unmittelbare Ergänzung zu HTML entwickelt, wird jedoch als proprietäre Script-Sprache nicht vom HTML-Standard erfaßt. JavaScript-Code wird über den **Script-Tag** in HTML eingebunden, vom Server vollständig zum Browser übertragen und dort von einem speziellen Browser-integrierten JavaScript-Interpreter ausgeführt. JavaScript ist klar von der **objektorientierten Programmiersprache Java** zu unterscheiden. **Java-Applets** sind vorkompilierte Java-Programme, die mit dem **Applet-Tag** in HTML-Dokumente eingebunden werden. Beim Zugriff wird das Applet vom Server des Anbieters in den lokalen Browser übertragen und dort völlig eigenständig von einer Browser-integrierten Java Virtual Machine (z. B. HotJava von Sun, Netscape ab Ver. 2.0, MS Internet Explorer 3.0) ausgeführt.

Die Möglichkeiten, allein mit HTML die kommunikativen und wirtschaftlichen Potentiale des WWW auszuschöpfen, sind begrenzt. CGI, JavaScript und Java knüpfen an den Grenzen von HTML an und stellen erweiterte Möglichkeiten zur Dynamisierung von WWW-Inhalten, Interaktion und Animation bereit. Aufgrund ihrer Fähigkeiten, diese Erweiterungen und Hypermediafunktionalität zu integrieren, werden HTML-Umgebungen zukünftig verstärkt als Organisationszentren für WWW-Projekte fungieren, die auch öffentlich nicht zugängliche Netzbereiche von Unternehmen (Intranet) umfassen. Führende Workgroup-Computing-Produkte wie Lotus Notes oder Novell Intranetware verwenden bereits die HTML-Technik.

2 Java - Eine objektorientierte Programmiersprache

2.1 Java Developer Kit und Java Virtual Machine

Java ist eine von Sun Microsystems entwickelte, **objektorientierte Programmiersprache**, die im Bereich des World Wide Web (WWW) an den Leistungsgrenzen von HTML anknüpft und erweiterte Möglichkeiten zur Dynamisierung von WWW-Inhalten, Interaktion und Animation bereitstellt. Die Präsenz von Java als Alternative oder Supplement zu HTML im WWW ist jedoch lediglich der hochaktuelle Nebenschauplatz einer Programmiersprache, die zur Entwicklung beliebiger Software eingesetzt werden kann. Die Anfänge von Java liegen im Jahr 1991. Das Team um James Gosling bei Sun Microsystems entwickelte ausgehend von den Unzulänglichkeiten von C++ eine Programmiersprache, welche die Steuerung elektronischer Gebrauchsartikel ermöglichte. Es entstand 1992 die Sprache Oak, die später in Java umbenannt wurde. Offiziell wurde Java am 23. Mai 1995 von Sun Microsystems vorgestellt.

Mit dem **Java Developer Kit (JDK)** bietet Sun Microsystems eine kostenlose Entwicklungsumgebung zur Erzeugung von Java-Programmen an (andere finden sich z. B. bei www.roguewave.com, www.symantec.com, www.isg.de, www.metroworks.com). Ein Java-Compiler erzeugt Programme in maschinenunabhängigem Bytecode. Dieser neutrale Bytecode wird von einem Java-Interpreter, der sog. **Java Virtual Machine**, zum Ablauf gebracht. Java Virtual Machines sind für die gängigsten Betriebssysteme und Prozessorplattformen (SunOS, SGI, DEC OSF/1, Windows NT und 95, Linux, Apple Macintosh) verfügbar. Somit ist ein Java-Programm ohne Portierungsaufwand auf jeder Hardware- und Betriebssystemplattform einsetzbar, auf der eine Java Virtual Machine läuft.

2.2 Java-Applications, Java-Applets und HTML

Mit Entwicklungsumgebungen wie dem JDK können Anwendungen entwickelt werden, die völlig eigenständig auf einem Computer auch außerhalb der HTML-Umgebung eines WWW-Browsers ablaufen. Derartige **Java-Applications** (z. B. Wordperfect in Java, Corel Office in Java) benötigen lediglich eine Java-ausführende Laufzeitumgebung wie z. B. die Java Virtual Machine in einem beliebigen Betriebssystem oder das Betriebssystem Java OS von SUN. **Applets** sind hingegen Java-Programme, die mittels einer speziellen Auszeichnung (HTML-Tag) in HTML-Dokumente eingebunden werden. Greift ein Benutzer auf eine solche HTML-Seite im WWW zu, wird das Applet neben den HTML-Anweisungen vom Web-Server des Anbieters auf den lokalen Computer heruntergeladen. Die in den Client-Browser integrierte Java Virtual Machine (z. B. bei HotJava von Sun, Netscape ab Ver. 2.0, MS Internet Explorer 3.0) führt daraufhin das Applet innerhalb der lokalen HTML-Umgebung aus. Entgegen der Einbettung von Graphiken oder anderen Multimedia-Objekten per CGI in eine HTML-Umgebung –dort findet die Verarbeitung auf dem entfernten Webserver statt und nur das Verarbeitungs-

ergebnis wird lokal angezeigt – läuft ein Java-Applet auf dem lokalen Rechner völlig unabhängig vom Web-Server ab, der das Applet geliefert hat.

Die Funktionsweise von Java-Applets soll am obligatorischen „Hello world“-Programm exemplarisch gezeigt werden. Der syntaktisch C++-ähnliche Java-Quellcode wird in einem Editor erfaßt und unter dem Namen „Helloworld.java“ abgespeichert. Aus der Quellcode-Datei erzeugt der Sun-Java-Compiler „javac“ die Datei „Helloworld.class“. Diese Datei enthält den maschinenunabhängigen Bytecode, der über eine Browser-integrierte Java Virtual Machine „Hello world!“ als lokale Bildschirmausgabe erscheinen läßt.

Die Einbindung eines nach diesem Muster erzeugten Java-Applets in ein HTML-Dokument erfolgt durch den **<applet> Tag**, der den Class-Namen, Attribute wie z. B. die Anzeigebreite und -höhe und Applet-spezifische Parameter (sofern im Java-Quellcode initialisiert) einschließt (Abb. 3).

```
<HTML>
<HEAD>
<TITLE>HTML-Beispiel</TITLE>
</HEAD>
<BODY>
<FONT SIZE=2>
<P>Java-Applet-Beispiel</P>
<APPLET code=”Helloworld.class” width=100 height=30>
<param name=text value=”Hello world!”>
</APPLET>
</BODY>
</HTML>
```

Abb. 3: HTML-Einbettung eines Java-Applets

Während der HTML-Quellcode einer jeden Page mit der „View Document Source“-Funktion der aktuellen Browser einsehbar ist, kann der Java-Quellcode wirksam geschützt werden. Von einem Applet muß lediglich der binäre Bytecode in den Class-Dateien zur Verfügung gestellt werden. Dies trägt sicherlich zum Schutz geistigen Eigentums bei, fördert jedoch zugleich die Kommerzialisierung des Internet. Mit den in naher Zukunft flächendeckend verfügbaren Electronic-Payment-Systemen läßt sich dann auch Java-Code gewinnbringend vermarkten. Dem Software-Absatz per "Click" steht dann nichts mehr im Wege. Viele Software- und Hardwareanbieter haben diesen Vertriebskanal erkannt und springen auf den Zug der "Network Computer" (NC) auf. Diese ur-

sprünglich von Sun konzipierten Arbeitsstationen sollen nicht mehr über externe Speichermedien lokal verfügen; einziger Software-Bestandteil ist ein rudimentäres Betriebssystem in Form einer hardware-codierten Java Virtual Machine. In Java codierte Anwendungssoftware, z. B. Textverarbeitungsmodule des Corel-Office-Paketes, muß zum Zeitpunkt des Bedarfs über eine Netzanbindung vom Application-Server des Software-Anbieters in den Arbeitsspeicher des NC am Arbeitsplatz des Software-Benutzers geladen werden.

Wer bereits in einer objektorientierten Programmiersprache wie C++, Smalltalk, Simula oder Objective C bereits geübt ist, wird den leichtesten Zugang zur Programmiersprache Java finden. Im Gegensatz zu C++ erzwingt jedoch die konsequente Objektorientierung von Java die vollständige Umstellung von prozeduraler zu objektorientierter Denkweise. Neben diesem neuen Programmier-Paradigma erfordert die professionelle Entwicklung von Java-Software insbesondere auch die Verinnerlichung von objektorientierten Analyse- und Designmethoden.

2.3 Java und JavaScript

Für die Gestaltung von Web Sites dagegen verspricht die proprietäre Makrosprache JavaScript von Netscape einen relativ einfachen und schnellen Einstieg ohne umfangreiche Programmierkenntnisse. JavaScript wurde von Netscape als eigenständige Scriptsprache entwickelt und ist klar von der Programmiersprache Java zu unterscheiden. Java-Applets werden als vorkompilierte Programme ausgeführt; Java-Script-Code ist hingegen vollständig und im Klartext in HTML eingebunden und wird zur Laufzeit von einem speziellen Browser-integrierten JavaScript-Interpreter ausgeführt. Zur Zeit unterstützen Netscape ab Version 2.0 und der MS Internet Explorer ab Version 3.0 JavaScript.

Im Gegensatz zu Java ist JavaScript eine unmittelbare Ergänzung zu HTML mit einem eigenen Befehlssatz. Analog zum <applet> Tag für Java-Applets erfolgt die Einbindung von JavaScript-Sequenzen über einen **<script> Tag**. Der im Vergleich zu Java begrenzte Sprachumfang und einfache Aufbau macht JavaScript relativ unkompliziert für den Programmierer; in JavaScript lassen sich jedoch keine Programme erzeugen, die außerhalb eines JavaScript-fähigen Browsers eigenständig ablaufen können.

Für den Web-Surfer als Anwender ist JavaScript in den Bereichen Performance und Stabilität kritisch zu beurteilen. Zum einen läuft das Interpretieren von JavaScript-Code sehr viel langsamer ab als das Interpretieren von vorkompiliertem Java-Bytecode. JavaScript bietet sich daher nur für begrenzte und einfache Programme an. Ein anderer gewichtiger Negativpunkt von JavaScript ist, daß ohne einen Compilierungslauf wie bei Java auch keine automatisierte Prüfung auf Programmierfehler stattfindet. Bspw. können (un)beabsichtigte Endlosschleifen oder Syntaxfehler den Client-Browser oder das gesamte Client-System zum Absturz bringen.

2.4 Zur Sicherheit von Java und JavaScript

In puncto **Sicherheit** von Java und **JavaScript** dreht sich die Grundsatzdiskussion um die Fragen, ob und wie die Ausführung von Fremdprogrammen auf einem lokalen Rechner kontrollierbar ist. Die Diskussion entspann sich Ende 1995 aufgrund einer Reihe von konzeptionellen Nachlässigkeiten und Implementierungsfehlern, die in den ersten Releases von Java und JavaScript vorgefunden, dann aber sukzessive behoben wurden. Zwischen den konkreten Sicherheitsproblemen von Java und denjenigen von JavaScript muß deutlich unterschieden werden.

Mit der Version 2.0 des Netscape Navigators ist es z. B. noch möglich, daß fremder JavaScript-Code den Client-Browser veranlaßt, eine eMail vom Account des Browser-Benutzers an eine im JavaScript-Code definierte Adresse zu verschicken, ohne daß der Benutzer darauf aufmerksam gemacht wird. Die Netscape Version 2.0 läßt ebenfalls zu, daß per JavaScript-Code unbemerkt Verzeichnis- und Datei-Listings (nicht jedoch die Dateiinhalte) aus dem lokalen Zugriffsbereich des Browser-Accounts gelesen (nicht jedoch lokal modifiziert) und an beliebige Adressen im Internet verschickt werden können. In den Release Notes zum Navigator der Version 2.02 versichert Netscape, daß ab dieser Version die verdeckte eMail-Versendung und die Listings lokaler Speicherressourcen per JavaScript nicht mehr möglich sind. Die Release Notes zur Navigator-Version 3.0 enthalten keine Statements mehr zu neuen Security Features von JavaScript. Wer seinen Browser absolut „JavaScript-sicher“ machen will, nutze die im Navigator 2.01 erstmals angebotene Möglichkeit, die Abarbeitung von JavaScript-Code zu unterbinden. Im MS Internet Explorer 3.0 ist eine solche Unterbindung nicht möglich.

Die bis Anfang 1997 aufgelaufenen **Sicherheitsprobleme** bezüglich unkontrollierter Zugriffe auf lokale Rechner- und Netzwerkressourcen durch **Java-Applets** beziehen sich vollständig auf Implementierungsfehler in den Java-Umgebungen, die nach ihrer Erkennung jedoch schnell behoben wurden. Der Netscape Navigator 3.0 (Endversion) und der MS Internet Explorer 3.0 (inklusive des Security Patches vom August 1996) erlauben es einem Java-Applet z. B. nur noch, auf denjenigen Host zuzugreifen, von dem es selbst stammt. Sun´s Appletviewer gibt den Zugriff nur auf diejenigen lokalen Ressourcen frei, die der Benutzer in einer „access control list“ explizit angibt. Mit den vorgenannten Browsern ist es (derzeit) Java-Applets ebenfalls nicht mehr möglich, unbemerkt eMails zu verschicken, Programme und Betriebssystembefehle lokal ausführen zu lassen oder in geschützten Netzwerkbereichen unautorisiert Berechtigungen zu erteilen. Das Secure Internet Programming Team an der University of Princeton (http:// www.cs.princeton. edu/sip/), das die Java-Sicherheitsprobleme aufmerksam beobachtet und evaluiert, vermeldet die letzten nennenswerten Java-Bugs für August 1996, die jedoch bereits in den neuesten Browser-Versionen abgefangen werden.

Konzeptionell verfügt Java mit seiner Speicherverwaltung, dem Bytecode-Verifizierer und dem Class Loader über ein ausgefeiltes **Security Model**, das die unbemerkte Informationsübermittlung sowie die Schädigung der lokalen Rechner- und Netzressourcen verhindern kann. Das Security Model umfaßt dabei die Bereiche der Programmentwicklung und des Programmablaufs.

Der Programmentwickler kann nicht wie in C++ physikalische **Speicherbereiche** direkt manipulieren. Java-Compiler referenzieren Speicher ausschließlich über symbolische Links, die erst bei Ausführung vom Java-Interpreter in echte Speicheradressen umgewandelt werden. Zusätzlich wird jeder lokale oder heruntergeladene Bytecode vor seiner Ausführung dem **Bytecode-Verifizierer** des ausführenden Java-Systems übergeben. Dieser überprüft den Bytecode auf syntaktische Korrektheit und eventuelle Schutzverletzungen, die zu einem Absturz des lokalen Systems führen könnten. Erst wenn die technischen Sicherheitsanforderungen erfüllt sind, kann ein Applet in einem Java-fähigen Browser interpretiert werden. Der Java **Class Loader** stellt dabei sicher, daß geprüfte lokale Klassen nicht von heruntergeladenen, „fremden“ Klassen überschrieben werden. Zugriffe externer Applets auf lokale Ressourcen werden in den neuesten Netscape- und Microsoft-Browsern vollständig unterbunden; Schnittstellen zu Netzwerkprotokollen (ftp, telnet, http etc.) werden über Zugriffsrechte von Domains oder Hostrechnern kontrolliert. Theoretisch geben Java-Applets damit keinerlei unkontrollierte Zugriffe auf Dateisysteme, Speicherbereiche oder Kommunikationsschnittstellen frei. Die 100%ige Sicherheit vor gewieften Crackern ist wie bei JavaScript in der Praxis nur gewährleistet, wenn der Browser-Benutzer die Ausführung von heruntergeladenem Java-Code per Disable-Checkbox abschaltet – auf die positiven Java-Features muß dann jedoch ebenfalls verzichtet werden.

2.5 Java-Potentiale

Die Potentiale von Java liegen nur vordergründig in der **Dynamisierung und Animation** von WWW-Hypermediadokumenten. Neben audiovisuellem Unterhaltungswert und bisher begrenzten Interaktionsmöglichkeiten gewinnt die **wirtschaftlich relevante Nutzung** von Java rasant an Bedeutung. Eine von Sun vorgestellte Mikroprozessorfamilie wurde für Java-basierte NCs (NetworkComputer) optimiert und wird zumindest deren Verbreitung im unternehmensinternen Intranet-Bereich vorantreiben. Seit Ende 1996 lizensiert Sun Java auch als Bestandteil von Betriebssystemen. Weitgehend alle Betriebssystemhersteller haben ihre Absicht erklärt, Java auf diese Weise der breiten Masse von Software-Entwicklern für Anwendungen mit höchster Portabilität zugänglich zu machen. In Linux z. B. ist zum aktuellen Zeitpunkt bereits ein Java-Bytcode-Interpreter enthalten. Der Kreis der Entwickler wird durch die freie Verfügbarkeit der Java-Sprache, der zugehörigen Werkzeuge und der APIs (Application Programming Interfaces), den JavaBeans, von Sun bewußt offen gehalten.

3 ActiveX – Microsofts Antwort auf Java

3.1 Framework auf Basis von OLE

Java hier und ActiveX dort – falsch verstandene Begriffe und technologische Hintergründe sind oftmals die Auslöser für Diskussionen, ob ActiveX oder Java der richtige Weg für Internet-Anwendungen ist. Im Frühjahr 1996 kündigte Microsoft die ActiveX-Technologie als neues Framework auf der Basis ihres **Objektstandards OLE** (Object Linking and Embedding) an. ActiveX überträgt das OLE-Modell auf das Internet und macht den Browser zum graphischen Frontend für beliebige Anwendungen. Wer sich mit OLE (Object Linking and Embedding) und COM (Component Object Model) beschäftigt, erkennt ziemlich schnell, daß ActiveX nicht neu ist. ActiveX ist der aktuell letzte Schritt von Microsoft in einer langen Entwicklung, die verteilte Datenhaltung und Interoperabilität zwischen Applikationen auf unterschiedlichen Rechnerplattformen ermöglicht.

Schon in den Anfängen von Microsofts Windows gab es die Möglichkeit, über Cut-Copy-Paste-Funktionen (Ausschneiden, Kopieren, Einfügen) Daten mit anderen Elementen eines Programmes auszutauschen. Später wurde diese einfache Technik um das DDE-Protokoll (Dynamic Data Exchange) erweitert. Es bot erstmals programmierbaren Datenaustausch zwischen verschiedenen Anwendungen. Der Fokus bei DDE lag auf den auszutauschenden Daten. Mit Objekten hatte dieses Protokoll nichts zu tun. OLE (Object Linking and Embedding) löste DDE ab. Objekte einer Windows-Anwendung (z. B. eine Excel-Tabelle) ließen sich nun in Objekte anderer Windows-Anwendungen (z. B. ein Word-Dokument) einbinden. Die sogenannte In-Place-Activation ermöglicht hierbei die Bearbeitung des Objektes direkt in der "fremden" Anwendung; die Excel-Tabelle läßt sich also direkt in Word bearbeiten. Die heutige OLE-Technologie in Windows basiert auf dem Microsoft Component Object Model (COM) und DCOM (Distributed COM) für verteilte Anwendungen in Netzwerken.

3.2 VBX – OCX – ActiveX

Der Durchbruch modularer Windows-Software ist eng mit dem Erfolg von Microsofts **Visual Basic Extensions (VBX)** verbunden. VBX sind eigenständige Softwarebausteine, die bestimmte Aufgaben auf standardisierte Art und Weise abarbeiten und die über eine Programmiersprache wie Visual Basic, Visual C++ oder Borland Delphi zu einer komplexen Anwendung zusammengefügt werden können.

Die erste Generation von VBX-Komponenten unterlag der Beschränkung, daß die Zusammenarbeit mit Visual Basic lediglich für eine 16-Bit Betriebssystemumgebung konzipiert war. Die 32-Bit-Variante, die zweite Generation, nannte sich schließlich **OCX (OLE Control Extension)** und kam 1995 mit der Einführung von Visual Basic 4.0 auf den Markt.

Mit dem Boom des Internet eröffnete sich der Komponentenidee eine neue Perspektive: Die Softwarebausteine werden über das Netz als Bestandteil von HTML-Seiten geladen – ähnlich wie Bilder oder Animationen. Auch die massive Verbreitung und der Erfolg von Java und Java Applets veranlaßte Microsoft, die vorhandene COM-Komponentenarchitektur auf das Internet zu übertragen: Aus OCX wurde Mitte 1996 ActiveX. Mit ActiveX bezeichnet Microsoft die Internet-Erweiterungen des COM.

Das **Component Object Model (COM)** hat wenig mit Objektorientierung zu tun. Es wurde mit dem Ziel entwickelt, daß eine Software-Komponente auf die Dienste einer anderen Komponente zugreifen kann. Im COM wird spezifiziert, in welcher Weise wiederverwendbare Softwarekomponenten implementiert werden sollen und wie diese Komponenten miteinander kommunizieren können. Vorgesehen sind lokale Funktionsaufrufe, Mechanismen zum Nachrichtenaustausch über Interprozeßkommunikation und Formen von Netzwerkkommunikation. COM bildet somit ein allgemeines Muster für die Interaktion zwischen den verschiedensten Bestandteilen von Software-Bibliotheken, Anwendungen und System-Software (siehe Abb. 4).[1]

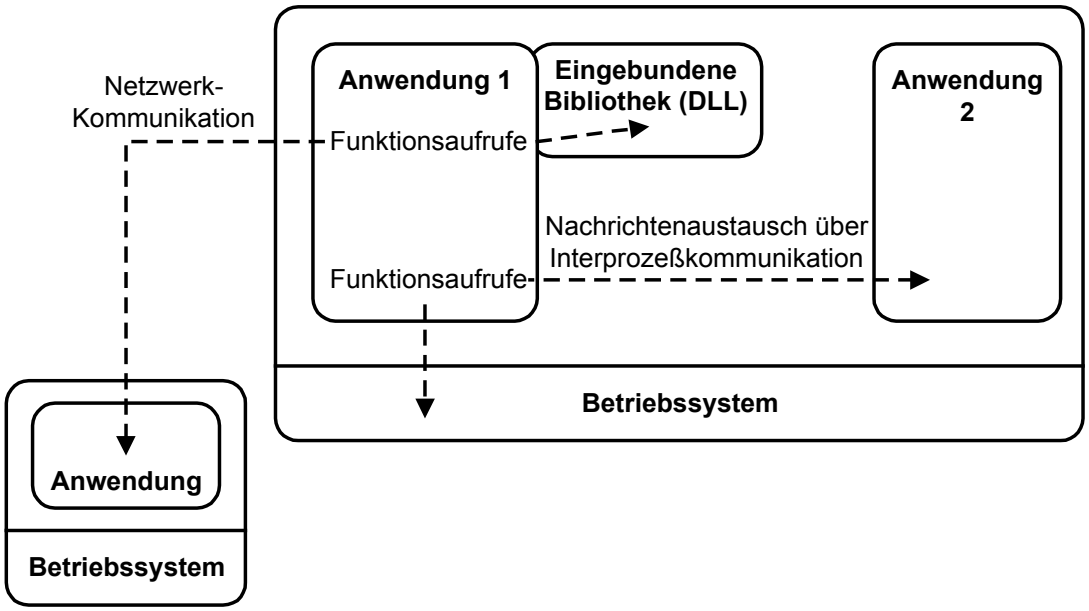

Abb. 4: COM-Prinzip für verteilte Anwendungen

Jedes COM-Objekt bietet seine Dienste über Methoden an, die in Schnittstellen zusammengefaßt sind. Beispielsweise kann eine Rechtschreibprüfung als COM-Objekt implementiert werden. Dieses Objekt könnte eine Schnittstelle unterstützen, die Methoden wie beispielsweise LookUpWord (WortNachschlagen) oder AddToDictionary (InWörterbuchAufnehmen) enthält. Über diese Schnittstelle wird beliebigen Anwendungen (Excel, Word, Powerpoint etc.) die Nutzung der Rechtschreibungsfunktionen ermöglicht. DCOM erlaubt die Verteilung solcher Dienste auf mehrere Server im Netzwerk (Remote Server).

1 Vgl. Chappel, David: ActiveX und OLE verstehen, Microsoft Press: 1996, S. 19.

3.3 ActiveX-Controls und HTML

Ein **ActiveX-Control (ActiveX-Steuerelement)** ist wie ein von objektorientierten Programmiersprachen bekanntes Objekt aufgebaut. Sein Verhalten wird in Methoden abgebildet, Eigenschaften beschreiben den Zustand des Objektes und die Kommunikation mit dem Objekt wird über Ereignisse (z. B. Benutzeringaben) gesteuert.

ActiveX-Steuerelemente enthalten den **Binärcode ausführbarer Programme** – sie sind also performanter als Java-Bytecode, der interpretiert werden muß. Ausführbarer Binärcode bedeutet jedoch, daß ActiveX-Steuerelemente nur in einer abgestimmten Systemumgebung lauffähig sind; sie benötigen einen sogenannten Steuerelementecontainer. Der Container kann über Visual-Basic, Visual C++ etc. in eine Anwendung oder einen WWW-Browser integriert sein. Microsofts Internet Explorer ab Version 3.0 und (via Plugin der Firma NCompass Labs) Netscapes Navigator ab Version 3.0 bieten z. B. Containerdienste für die Ausführung von ActiveX-Steuerelementen an.

ActiveX-Steuerelemente lassen sich im Microsoft Internet Explorer mit Hilfe des **HTML-Tags <Object...>** und beim Netscape Navigator mit dem **Embed-Tag** in Webseiten einbinden und zusätzlich über Visual Basic Script (VBScript) oder Java Script mit Logik versehen (Beispiel siehe Abb. 5). Greift der Benutzer auf eine solche HTML-Seite zu, wird das Steuerelement, sofern es auf dem lokalen Rechner noch nicht vorhanden ist, vom Web-Server auf den PC heruntergeladen. Trifft der Internet Explorer auf den Embed-Tag, wird dieses ignoriert; das ActiveX-Control wird nicht geladen und angezeigt. Dasselbe gilt für das Object-Tag beim Netscape Navigator. Im folgenden wird die Verwendung des Object-Tags unter dem Internet Explorer 3.0 beschrieben.

Nach der Überschrift *Click An Arrow* verwendet dieses HTML-Dokument das HTML-Tag *Object*, um ein ActiveX-Steuerelement zu laden – in diesem Fall das Steuerelement Drehfeld (Spin Button). Der Spin Button erleichtert die Einstellung von Zahlenwerten durch zwei horizontale oder vertikale Pfeil-Buttons (siehe Abb. 6). Das Tag *Object* legt die Klassenkennung, die Steuerelementeposition und den Pfad fest, von dem das Steuerelement bei Bedarf per Internet herunterzuladen ist. Bevor der Browser ein gefordertes ActiveX-Steuerelement lädt und lokal registriert, überprüft er anhand der per *CLSID* angegebenen Klassenkennung, ob dieses bereits in der lokalen Registrierdatenbank (Registry) verwaltet wird. Ist dies nicht der Fall, erfolgt das Laden per Internet mit nachfolgender Selbstregistrierung. Damit der Browser erfährt, wo sich der Code befindet, muß das Object-Tag das Attribut *Codebase* beinhalten. Liest der Browser dieses Attribut, lädt er die Datei (vorausgesetzt der Anwender bestätigt das Laden) von angegebenen Webserver und erstellt eine Instanz dieses Steuerelementes. Anders als Java-Downloads verbleiben einmal geladene ActiveX-Steuerelemente dauerhaft und wiederverwendbar auf dem lokalen System.

Neben den Angaben zum Steuerelement läßt sich in ein HTML-Dokument auch ein **Script** (in VBScript oder JavaScript) einbinden. Script-Sprachen dynamisieren WWW-Inhalte, ermöglichen Animationen und Interaktionen mit dem User. Im Beispiel aus Abbildung 5 folgt hinter dem Object-Tag ein einfaches VBScript-Programm, welches das Spin-Button-Steuerelement ansteuert. Nach dem Klicken auf einen der im Browser an-

gezeigten Pfeile soll eine Meldung eingeblendet, die besagt, welcher der beiden Pfeile angewählt wurde. Das Steuerelement reagiert auf das Auslösen dieses Klick-Ereignisses und sendet eine Nachricht an die entsprechende Ereignisprozedur *Sub SpinButton_SpinUp()* oder *SpinButton_SpinDown(),* damit diese abgearbeitet wird.

```
<HTML>
<TITLE> HTML Control Example</TITLE>
<BODY>
<H1>Click An Arrow</H1>
<P>
<OBJECT
      CLASSID="clsid:79176FB0-B7F2-11CE-97EF-00AA006D2776"
      CODEBASE="http://microsoft.com/activex/controls/spin32.ocx"
      ID="SpinButton1"
      WIDTH=100
      HEIGHT=121
      HSPACE=85>
</OBJECT>
<SCRIPT LANGUAGE=VBScript>
Sub SpinButton_SpinUp()
      MsgBox "Up arrow clicked"
End Sub
Sub SpinButton_SpinDown()
      MsgBox "Down arrow clicked"
End Sub
</SCRIPT>
</BODY>
</HTML>
```

Abb. 5: Beispiel für die HTML-Einbindung eines ActiveX-Controls

Bei genauerer Betrachtung verbirgt sich hinter ActiveX die bekannte OLE-Technologie mit optimierter Speicherverwaltung und neu konzipierten Schnittstellen. ActiveX-Controls reduzieren die Größe vormaliger OLE-Controls; sie erfordern einen geringerem Aufwand bei der Verteilung im Internet sowie kürzere Zeiten ür den Download in einen Browser. Werden Webseiten mit ActiveX-Controls angesteuert, müssen diese, sofern sie nicht auf dem lokalen Rechner vorhanden sind, auf diesen kopiert und in der Registry eingetragen werden. Nur lokal installierte ActiveX-Komponenten lassen sich in Webseiten ausführen.

Abb. 6: Ergebnis der HTML-Einbindung des ActiveX-Controls aus Abb. 5

Unter dem Sammelbegriff ActiveX sind eine ganze Reihe von Techniken sowohl für das Internet als auch für "normale" Anwendungen zusammengefaßt. So gibt es beispielsweise **ActiveX-Dokumente, ActiveX-Skripte und ActiveX-Steuerelemente.** ActiveX-Dokumente (oder auch OLE-Dokumente) sind Dateien, die Daten aus verschiedenen Anwendungen enthalten. In Word kann man sich ein ActiveX-Dokument als ein Schriftstück vorstellen, das Graphiken aus Corel Draw oder Tabellen aus Excel enthält. Mit ActiveX-Skripten lassen sich statische Web-Seiten dynamisieren. Sind in Web-Seiten programmierbare Objekte eingebunden, kann mit Skripten auf vom Benutzer ausgelöste Ereignisse wie z. B. einem Mausklick auf eine Befehlsschaltfläche reagiert werden. ActiveX-Steuerelemente übernehmen dieselbe Funktion wie VBX- bzw. OCX-Komponenten und können in individuell programmierten Anwendungen ebenso wie in Browsern eingesetzt werden. Mittlerweile gibt es eine große Anzahl von Steuerelementen für spezielle Aufgaben bei hunderten von Anbietern zu kaufen.

Mit dem neuen Visual Basic 5 CCE (Custom Control Edition) von Microsoft erhält der Entwickler erstmals eine kostenlose Entwicklungsumgebung zum Implementieren von ActiceX-Steuerelementen unter Basic. Mit der CCE als Light-Version von Visual Basic 5.0, können keine vollständigen Anwendungen, sondern nur Steuerelemente entwickelt werden. In dem von Microsoft ebenfalls kostenlos abgegeben ActiveX Software Development Kit (SDK) sind die Spezifikation für COM sowie einige ActiveX-Steuerelemente enthalten, die vor allem für Visual Basic- und Delphi-Entwickler interessant sind. Diese Steuerelemente sind nicht für die Implementierung in Webseiten vorgesehen. Sie realisieren vielmehr alle wichtigen Internet-Protokolle wie TCP, SMTP, POP, FTP, HTTP und HTML. Man kann mit wenigen Programmierzeilen einen SMTP-Mailer bauen, einen HTTP-Server realisieren oder Dateien per FTP kopieren.

3.4 ActiveX und Java

ActiveX und Java: Diese auf den ersten Blick konkurrierenden Produkte können parallel auf einer Web-Seite integriert werden und sich gegenseitig ergänzen. Bei einem Java-Applet wird der Java-Quellcode in einen rechnerunabhängigen Bytecode übersetzt. Dieser Bytecode wird dann von der Java Virtual Machine interpretiert. Da Java-Applets in Form von Bytecode und nicht als rechnerspezifische Binärdatei verteilt werden, können auf unterschiedlichen Rechnersystemen diese Applets gestartet werden.

Microsoft hat eine Java Virtual Machine in den Internet Explorer implementiert und integriert auf diese Weise Java- und COM-Objekte (siehe Abb. 7). Ein Teil dieser Integration sieht so aus, daß durch die Java Virtual Machine für den Programmierer nach außen kein Schnittstellen-Unterschied zwischen einem Java-Objekt (z. B. Java Applet) und einem COM-Objekt (z. B. ActiveX-Control) erkennbar ist. Um die Java-Objekte wie COM-Objekte aussehen zu lassen, werden alle Dienste der Objekte transparent von der Java Virtual Machine zur Verfügung gestellt. Da Java-Applets für den Programmierer wie ActiveX-Steuerelemente aussehen, können auch Applets über Skripte angesteuert werden. Mit einer Skript-Sprache kann auf die von einem Applet bereitgestellten Methoden zugegriffen werden. Somit können auf einer Web-Seite Java-Applets parallel neben ActiveX-Steuerelementen verwendet werden.

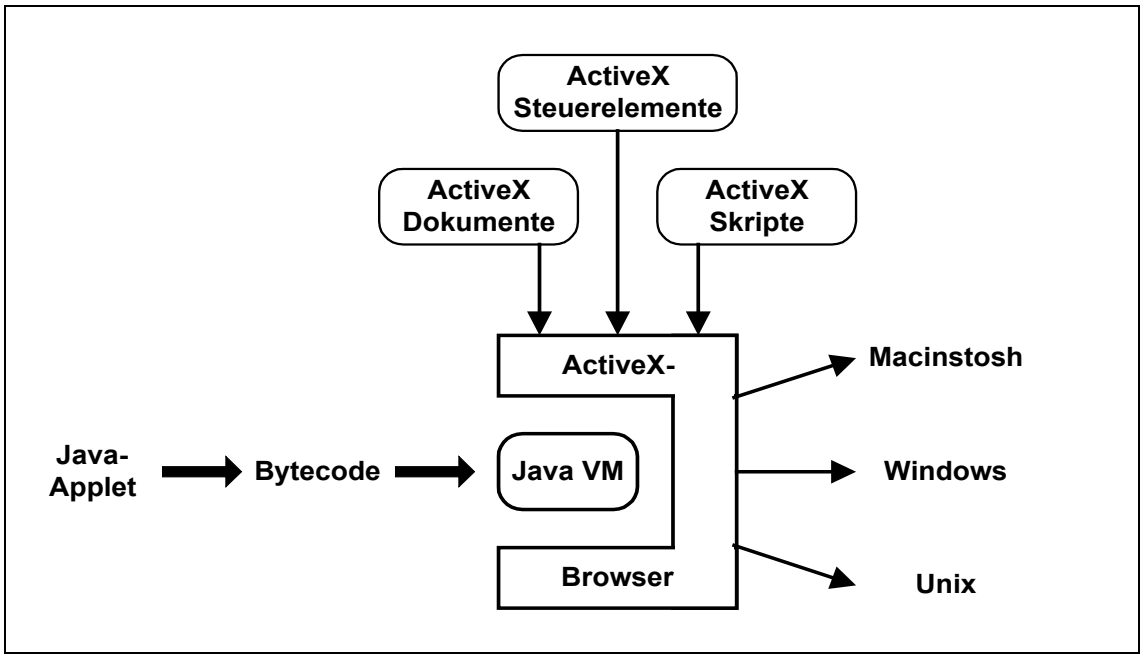

Abb. 7: Schema der COM- und Java-Integration von Microsoft

Die hohe Akzeptanz von Java ist eng mit der Betriebssystemunabhängigkeit von Java-Applets verbunden. Die ActiveX-Technologie steht bisher nur unter Windows 95 bzw. Windows NT zur Verfügung. Microsoft arbeitet an einer Integration der ActiveX-Tech-

nologie auf Macintosh- und Unix-Systemen, um die Funktionalität von ActiveX auch auf anderen Plattformen verfügbar zu machen.

Mitte 1996 reagierte Sun mit dem API-Rahmenwerk **"Javabeans"** auf die massive Marketing-Promotion der ActiveX-Technologie. API steht für "Application Programming Interface" und bezeichnet ein Konzept für Anwendungsschnittstellen. Java-Beans sind feiner strukturiert als Java-Applets und können aus einer einzigen Objektklasse bestehen. Die Beans sollen in erster Linie die Kommunikation und Kooperation zwischen Anwendungsmodulen verschiedenster Rahmen-Technologien (Java-Application, Java-Applets, ActiveX-Controls, OpenDoc, LiveConnect) ermöglichen. Bezüglich der Realisierung modularer Anwendungen stehen Java-Beans und ActiveX somit in direkter Konkurrenz zueinander.

3.5 Zur Sicherheit von ActiveX

Da es sich bei ActiveX-Steuerelementen um binäre Module handelt, die vollen Zugriff auf das darunterliegende Betriebssystem haben, spielt der Sicherheitsaspekt eine gewichtige Rolle. Es sind eine Reihe von Fällen bekannt geworden, bei denen z. B. ActiveX-Steuerelemente aus dem Internet auf den lokalen Rechner geladen wurden, die anschließend die Festplatte formatiert oder Windows 95 heruntergefahren haben (http://www.halcyon.com/mclain/activex).

Berichten des Chaos Computer Clubs zufolge ist mit dem Microsoft Internet Explorer 3.0 und "mit ActiveX eine Fernsteuerbarkeit des eigenen Computers, ohne daß dies vom Benutzer nachvollzogen werden kann", möglich.[2] Hacker demonstrierten einen "Kontoklau". Möglich wird dies, wenn sowohl Microsofts Internet Explorer und eine Homebanking-Software gleichzeitig auf dem lokalen Computer läuft. Sieht sich der Benutzer vermeintlich harmlose Webseiten an, wird durch ein entsprechend manipuliertes ActiveX-Steuerelement in einer Homebanking-Software-Datei ein Geldtransaktionssatz angefügt. Beim nächsten Abgleich mit der Bank wird der Benutzer nach einer Anzahl von TAN's (Transaktionsnummern) gefragt, ihm wird aber nicht mitgeteilt, wofür diese Verwendung finden.

"Microsoft antwortet auf das in Java vorhandene Sicherheitskonzept nur optisch eindrucksvoll. Ein Authentifizierungsalgorithmus stellt sicher, daß eine Komponente, die sich auf dem eigenen Rechner einnisten will, auch tatsächlich aus der urspünglichen Quelle stammt."[3] Bei diesem sogenannten **Code-Signing** wird die binäre ActiveX-Komponente in eine CAB-Datei (Cabinet) eingebettet. Diese Datei kann dann mit einer digitalen Signatur versehen werden und soll dem Anwender garantieren, daß das Steuerelement auf dem Weg vom Server zum Client nicht verändert worden ist. Bei einer digitalen Signatur handelt es sich um eine Bytefolge, mit deren Hilfe überprüft werden

2 Vgl.: Chaos Computer Club: Glasfaser bis in die Brieftasche. URL: http://www.ccc.de/CRD/DRC280197.html [Stand: 08.04.1997]

3 Vgl. Schmitt, H.-J.: Bewegliche Ziele – ActiveX: Microsofts Antwort auf Java, in: c't – Magazin für Computertechnik, 12/1996 S. 258.

kann, ob die mit ihr verbundenen Informationen (der heruntergeladene Code) von einer bestimmten Entität zur Verfügung gestellt werden. Ist das ActiveX-Steuerelement auf dem Client angekommen, wird vor seiner Ausführung zunächst abgeprüft, ob es sich um eine mit einem Zertifikat ausgestattete Komponente handelt. Dieses Zertifikat wird als Dialog angezeigt und gibt über den Zertifizierer und den Hersteller Auskunft (**"Authenticode"** von Microsoft). Dem Anwender wird mit im Zertifizierungsdialog aufgeführten Checkboxen jedoch angeboten, auf die Anzeige von Zertifikaten gänzlich zu verzichten. Fortan werden alle Steuerelemente, ob zertifiziert oder nicht, aus dem Internet auf den lokalen Computer geladen.

3.6 Ein Ausblick – Trident

Die ActiveX-Technologie ist zur Zeit noch plattformabhängig. Sie wird nur unter Windows 95 bzw. Windows NT und nur vom Microsoft Internet Explorer unterstützt. Der Netscape Navigator integriert ActiveX-Komponenten nur mit Hilfe eines Plugins der Firma NCompass. Allein mit den Mitteln des Code-Signings läßt sich auch das Sicherheitsproblem von ActiveX-Steuerelementen nicht lösen. Der Weg des Downloads der Applikationen auf den Client ist bei ActiveX-Komponenten nicht sicher genug. Microsoft hat sich des Problems angenommen, wobei abzusehen ist, daß im Internet-Umfeld mittelfristig der Trend von ActiveX-Komponenten auf der Client-Seite weggeht. Als Ersatz hat Microsoft die sogenannten Active-Server-Pages in Aussicht gestellt. Die steuernden Komponenten verbleiben auf dem Server, der lediglich "reinen" HTML-Code zum Client überträgt. **"Trident" alias "Dynamic HTML"** soll dies ermöglichen. Neben dem Sicherheitsgewinn hat dieses Konzept auch den Vorteil, daß die Web-Browser aller Hersteller entsprechende HTML-Dokumente verstehen.

Es bleibt anzumerken, daß Microsofts ActiveX-Technologie momentan bei den Web-Seiten-Designern und -Programmierern wenig Zuspruch findet und dementsprechend auf wenigen Web-Seiten integriert ist. Zum einem haben die ehemaligen OCXe überwiegend eine beachtliche Dateigröße – die Ladezeiten sind zu lang – und zum anderen ist ActiveX als proprietäre Microsoft-Technologie auf Windows beschränkt, wohingegen Java unter den verschiedensten Betriebssystemen lauffähig ist.

Quellenhinweise für Einsteiger

HTML

December, John; Ginsburg, Mark: HTML & CGI Unleashed – Professional Reference Edition, Macmillan Computer Publishing 1996.

Münz, Stefan: HTML-Dateien selbst erstellen, http://www.netzwelt.com/selfhtml/

Nefzger, Wolfgang; Münz, Stefan: HTML-Referenz 3.2, 2., völlig neubearb. Aufl., Franzis-Verlag 1996.

Ragget, Dave; Lam, Jenny; Alexander, Ian: HTML 3.2 – Neue Möglichkeiten für das Web-Publishing, Addison-Wesley 1996.

W3 Consortium (Hrsg.): HTML 3.2, 1997, http://www.w3.org/pub/WWW/MarkUp/Wilbur

Java und JavaScript

December, John: Java - Einführung und Überblick, Markt&Technik-Verlag 1996.

McGraw, Gary; Felten, Edward W.: Java Security – Hostile Applets, Holes and Antidotes, John Wiley and Sons 1996.

Mintert, Stefan: JavaScript – Grundlagen und Einführung, Addison-Wesley 1996.

Netscape Communications Corporation (Hrsg.): JavaScript Authoring Guide, 1996, http://home.netscape.com/eng/ mozilla/3.0/handbook/javascript/

Sun Microsystems (Hrsg.): Java Reaches New Heights, 1997, http://java.sun.com

Mandel, Andreas: Java-Tutorial zum systemtechnischen Workshop "Internet und WWW" an der Fachhochschule Furtwangen, http://www.fh-furtwangen.de/~mandel/wi5/www/aufbau.html

Kaffee & Kuchen – Die Java-Site, http://acc.de/java/

Kakao & Kekse – Die JavaScript-Site, http://acc.de/javascript/

Bank, David: The Java Saga, http://www.hotwired.com/wired/3.12/features/java.saga.html

König, Antje: Java Einblick, http://www.fhd-stuttgart.de/~ak02/java.htm

Klaiber, Simon: Java – Die besten Tips & Tricks der Profis, Sonderbeilage zur PC Professionell 12/1996.

ActiveX

ActiveX SDK unter http://www.activex.org. oder http://microsoft.com/intdev

Barlow, Chris: Scan for News, in: Visual Basic Programmers Journal, Nr. 9, August 1996.

Chaos Computer Club: Glasfaser bis in die Brieftasche, Online im Internet: URL: http://www.ccc.de/CRD/DRC280197.html [Stand: 8.4.1997]

Chappel, David: ActiveX und OLE verstehen, Verlag Microsoft Press, 1996.

Donnerhacke, Lutz: Vorsicht Falle – ActiveX als Füllhorn für Langfinger, http://www.ix.de/ix/artikel/1997/03/090/artikel.html

Grogan, Walter J.: Add ActiveX Controls to HTML Pages, in: Visual Basic Programmers Journal, Nr. 10 September 1996.

Maslo, Andreas: VBX, OCX und ActiveX: Technologie im Wandel, in BasicPro, 4/96, Beilage: Wildsite.

Meier, Bernd: OLE Custom Controls - Die Grundlage für Componentware, in: BasicPro, 1/96, S. 48-60.

Meyer, Hanns-Martin: Objekte integrieren mit OLE2, Springer-Verlag, Berlin u. a., 1994.

Mezick, D.: ActiveX Takes Center Stage, in: Visual Basic Programmers Journal, Nr. 12, Oktober 1996.

NCompass Labs ActiveX Plugin: http://www.ncompasslabs.com

Schmitt, H.-J.: Bewegliche Ziele – ActiveX: Microsofts Antwort auf Java, in: c't – Magazin für Computertechnik, 12/1996 S. 258 ff.

The ActiveX Gallery: http://www.microsoft.com/ie/controls

Bisher erschienen

Stand: Dezember 2000 – Den aktuellen Stand der Reihe erfahren Sie über unsere Web Site unter http//wi.uni-giessen.de

Nr. 1/1996	Grundlagen des Client/Server-Konzepts	Schwickert/Grimbs
Nr. 2/1996	Wettbewerbs- und Organisationsrelevanz des Client/Server-Konzepts	Schwickert/Grimbs
Nr. 3/1996	Realisierungsaspekte des Client/Server-Konzepts	Schwickert/Grimbs
Nr. 4/1996	Der Geschäftsprozeß als formaler Prozeß - Definition, Eigenschaften, Arten	Schwickert/Fischer
Nr. 5/1996	Manuelle und elektronische Vorgangssteuerung	Schwickert/Rey
Nr. 6/1996	Das Internet im Unternehmen - Neue Chancen und Risiken	Schwickert/Ramp
Nr. 7/1996	HTML und Java im World Wide Web	Gröning/Schwickert
Nr. 8/1996	Electronic-Payment-Systeme im Internet	Schwickert/Franke
Nr. 9/1996	Von der Prozeßorientierung zum Workflow-Management - Teil 1: Grundgedanken, Kernelemente, Kritik	Maurer
Nr. 10/1996	Von der Prozeßorientierung zum Workflow- Management - Teil 2: Prozeßmanagement und Workfflow	Maurer
Nr. 11/1996	Informationelle Unhygiene im Internet	Schwickert/Dietrich/Klein
Nr. 12/1996	Towards the theory of Virtual Organisations: A description of their formation and figure	Appel/Behr
Nr. 1/1997	Der Wandel von der DV-Abteilung zum IT-Profitcenter: Mehr als eine Umorganisation	Kargl
Nr. 2/1997	Der Online-Markt - Abgrenzung, Bestandteile, Kenngrößen	Schwickert/Pörtner
Nr. 3/1997	Netzwerkmanagement, OSI Framework und Internet SNMP	Klein/Schwickert
Nr. 4/1997	Künstliche Neuronale Netze - Einordnung, Klassifikation und Abgrenzung aus betriebswirtschaftlicher Sicht	Strecker/Schwickert
Nr. 5/1997	Sachzielintegration bei Prozeßgestaltungsmaßnahmen	Delnef
Nr. 6/1997	HTML, Java, ActiveX - Strukturen und Zusammenhänge	Schwickert/Dandl
Nr. 7/1997	Lotus Notes als Plattform für die Informationsversorgung von Beratungsunternehmen	Appel/Schwaab
Nr. 8/1997	Web Site Engineering - Modelltheoretische und methodische Erfahrungen aus der Praxis	Schwickert
Nr. 9/1997	Kritische Anmerkungen zur Prozeßorientierung	Maurer/Schwickert
Nr. 10/1997	Künstliche Neuronale Netze - Aufbau und Funktionsweise	Strecker
Nr. 11/1997	Workflow-Management-Systeme in virtuellen Unternehmen	Maurer/Schramke
Nr. 12/1997	CORBA-basierte Workflow-Architekturen - Die objektorientierte Kernanwendung der Bausparkasse Mainz AG	Maurer
Nr. 1/1998	Ökonomische Analyse Elektronischer Märkte	Steyer
Nr. 2/1998	Demokratiepolitische Potentiale des Internet in Deutschland	Muzic/Schwickert
Nr. 3/1998	Geschäftsprozeß- und Funktionsorientierung - Ein Vergleich (Teil 1)	Delnef
Nr. 4/1998	Geschäftsprozeß- und Funktionsorientierung - Ein Vergleich (Teil 2)	Delnef
Nr. 5/1998	Betriebswirtschaftlich-organisatorische Aspekte der Telearbeit	Polak
Nr. 6/1998	Das Controlling des Outsourcings von IV-Leistungen	Jäger-Goy
Nr. 7/1998	Eine kritische Beurteilung des Outsourcings von IV-Leistungen	Jäger-Goy
Nr. 8/1998	Online-Monitoring - Gewinnung und Verwertung von Online-Daten	Guba/Gebert
Nr. 9/1998	GUI - Graphical User Interface	Maul
Nr. 10/1998	Institutionenökonomische Grundlagen und Implikationen für Electronic Business	Schwickert
Nr. 11/1998	Zur Charakterisierung des Konstrukts “Web Site”	Schwickert
Nr. 12/1998	Web Site Engineering - Ein Komponentenmodell	Schwickert
Nr. 1/1999	Requirements Engineering im Web Site Engineering – Einordnung und Grundlagen	Schwickert/Wild
Nr. 2/1999	Electronic Commerce auf lokalen Märkten	Schwickert/Lüders
Nr. 3/1999	Intranet-basiertes Workgroup Computing	Kunow/Schwickert
Nr. 4/1999	Web-Portale: Stand und Entwicklungstendenzen	Schumacher/Schwickert
Nr. 5/1999	Web Site Security	Schwickert/Häusler
Nr. 6/1999	Wissensmanagement - Grundlagen und IT-Instrumentarium	Gaßen
Nr. 7/1999	Web Site Controlling	Schwickert/Beiser
Nr. 8/1999	Web Site Promotion	Schwickert/Arnold
Nr. 9/1999	Dokumenten-Management-Systeme – Eine Einführung	Dandl
Nr. 10/1999	Sicherheit von eBusiness-Anwendungen – Eine Fallstudie	Harper/Schwickert
Nr. 11/1999	Innovative Führungsinstrumente für die Informationsverarbeitung	Jäger-Goy
Nr. 12/1999	Objektorientierte Prozeßmodellierung mit der UML und EPK	Dandl
Nr. 1/2000	Total Cost of Ownership (TCO) – Ein Überblick	Wild/Herges
Nr. 2/2000	Implikationen des Einsatzes der eXtensible Markup Language – Teil 1: XML-Grundlagen	Franke/Sulzbach
Nr. 3/2000	Implikationen des Einsatzes der eXtensible Markup Language – Teil 2: Der Einsatz im Unternehmen	Franke/Sulzbach
Nr. 4/2000	Web-Site-spezifisches Requirements Engineering – Ein Formalisierungsansatz	Wild/Schwickert
Nr. 5/2000	Elektronische Marktplätze – Formen, Beteiligte, Zutrittsbarrieren	Schwickert/Pfeiffer
Nr. 6/2000	Web Site Monitoring – Teil 1: Einordnung, Handlungsebenen, Adressaten	Schwickert/Wendt
Nr. 7/2000	Web Site Monitoring – Teil 2: Datenquellen, Web-Logfile-Analyse, Logfile-Analyzer	Schwickert/Wendt
Nr. 8/2000	Controlling-Kennzahlen für Web Sites	Schwickert/Wendt
Nr. 9/2000	eUniversity – Web-Site-Generierung und Content Management für Hochschuleinrichtungen	Schwickert/Ostheimer/Franke

Bestellung (bitte kopieren, ausfüllen, zusenden/zufaxen)

Adressat:

Professur für BWL und Wirtschaftsinformatik
Fachbereich Wirtschaftswissenschaften
Licher Straße 70
D – 35394 Gießen
Telefax: (0 641) 99-22619

Hiermit bestelle ich gegen Rechnung die angegebenen Arbeitspapiere zu einem Kostenbeitrag von DM 10,- pro Exemplar (MwSt. entfällt) zzgl. DM 5,- Versandkosten pro Sendung.

Nr.	An
1/1996	
2/1996	
3/1996	
4/1996	
5/1996	
6/1996	
7/1996	
8/1996	
9/1996	
10/1996	
11/1996	
12/1996	

Nr.	An
1/1997	
2/1997	
3/1997	
4/1997	
5/1997	
6/1997	
7/1997	
8/1997	
9/1997	
10/1997	
11/1997	
12/1997	

Nr.	Anz
1/1998	
2/1998	
3/1998	
4/1998	
5/1998	
6/1998	
7/1998	
8/1998	
9/1998	
10/1998	
11/1998	
12/1998	

Nr.	Anz
1/1999	
2/1999	
3/1999	
4/1999	
5/1999	
6/1999	
7/1999	
8/1999	
9/1999	
10/1999	
11/1999	
12/1999	

Nr.	Anz
1/2000	
2/2000	
3/2000	
4/2000	
5/2000	
6/2000	
7/2000	
8/2000	
9/2000	

Absender:

_____
Organisation

_____
Abteilung

_____
Nachname, Vorname

_____
Straße

_____
Plz/Ort

_____
Telefon Telefax eMail

_____
Ort, Datum Unterschrift