Einführung in Java

M. Münzenberger, A. Wieczorek[1]

1 Java und das Internet

Im April letzten Jahres wurde Java, eine Softwareentwicklung von Sun Microsystems, auf der 3.-WWW-Konferenz in Darmstadt der Öffentlichkeit vorgestellt und hat seitdem zu einer teilweise aufsehenerregenden Berichterstattung in den Medien geführt. Um den Hintergrund dieser Euphorie zu beleuchten, soll zunächst die Entwicklung des Internets, in dem Java vorwiegend zum Einsatz kommen soll, kurz dargestellt werden.

Die Entwicklung begann Ende der 60er Jahre mit der Förderung der Computervernetzung durch das amerikanische „Department of Defense (DoD)". Hauptgegenstand der Entwicklungen in dieser ersten Phase waren das Verstehen und Gestalten von Protokollen und einfachen Diensten. So entstand 1969 der Dienst telnet, es folgte ftp und 1971 E-Mail. Als darunter liegendes Protokoll etablierte sich TCP/IP, das seit 1983 ausschließlich verwendet wurde. Das Netz wird seitdem als Internet bezeichnet. Mitte der 80er Jahre weckte das Netz Interesse im Bereich wissenschaftlicher Institutionen und wurde 1989 vom DoD losgelöst. Mitte 1991 existierten weltweit bereits 500.000 Internet-Server, die praktisch nur durch Universitäten und Forschungsinstitutionen genutzt wurden.

Am CERN, dem Europäischen Zentrum für Teilchenphysik bei Genf, begann 1989 mit der Entwicklung eines Systems zum Auffinden von verschiedensten Informationen innerhalb der Institution die zweite Phase. Als bestmögliche Lösung dieses Problems wurde ein auf Client/Sever Architektur basierendes Hypertext-System vorgeschlagen und auf NeXT-Rechnern implementiert. Auf einem oder mehreren Servern liegen dabei die Dokumente in einem Hypertext-Format vor. Dieses ermöglicht es unter anderem, Verbindungungen (Hyperlinks) zwischen den einzelnen Dokumenten aufzubauen. Ausgehend von einem Dokument kann sich so ein Informationssuchender vielfältige Information aus anderen Dokumenten durch Navigation von Dokument zu Dokument in einer netzwerkförmigen Struktur erschließen. Die Navigation erfolgt mit einem Client, der in diesem Zusammenhang auch Browser genannt wird.

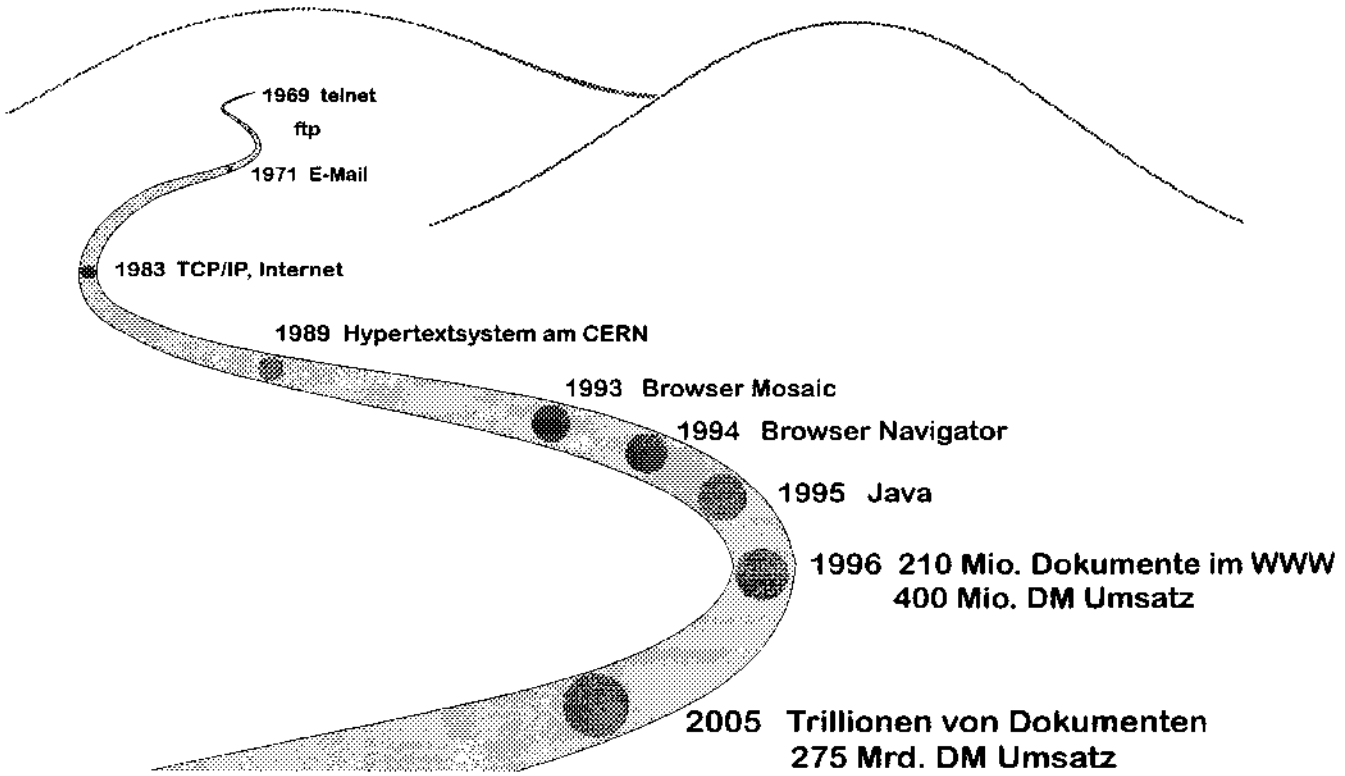

Bild 1: Entwicklung des Internets

[1] Die Autoren sind in einer Bundesbehörde tätig. Tel. 0611 - 554191

Das erstellte Hypertext-System wurde im Dezember 1991 in einem „CERN Newsletter" publiziert. Das Prinzip der navigierenden Erschließung von Dokumenten, die neben reinem Text auch audiovisuelle Information beinhalten können, ist nicht auf einen bestimmten Raum (Cyberspace) begrenzt. Die Idee, auf diese Art leicht zugänglich Informationen zu erschließen, wurde deshalb aufgenommen und auf einen neuen Dienst im Internet, dem World Wide Web (WWW), übertragen.

Im Dezember 1993 stellte Marc Andreessen vom National Center for Supercomputing Applications an der University of Illinois den von ihm entwikkelten Browser Mosaic vor; dieser fand im Internet eine schnelle Verbreitung. Zu dieser Zeit gab es im Internet etwa 50 öffentlich zugängliche WWW-Server, Anfang 1995 bereits 16000. Der Ausbau spiegelt auch das dann einsetzende kommerzielle Interesse wider.

Der 22jährige Student Marc Andreessen gründete 1994 zusammen mit James H. Clark, außerordentlicher Professor an der Stanford Universität und Gründer der Computerfirma Silicon Graphics, das Softwarehaus Netscape. Dieses brachte den Browser „Navigator" heraus und stellt diesen bis heute kostenlos zur Verfügung. Der Browser hat heute einen Marktanteil von 75 Prozent und unterstützt ab der Version 2.0, die Anfang des Jahres erschien, Java. Die dynamische Entwicklung zeigt sich auch am Börsenwert dieses Softwarehauses, der zeitweise auf knapp 7 Milliarden Dollar kletterte, bei einem Umsatz von nur 80 Millionen Dollar.

Das Internet und speziell das WWW stellt sich damit als äußerst dynamisch wachsender Dreh- und Angelpunkt in unserer Informationsgesellschaft dar. Derzeit benutzen das Internet etwa 60 Millionen Menschen mit einem Zuwachs von 13% pro Monat, und es wird von einem Umsatz von ca. 400 Millionen DM bei 210 Millionen Dokumenten ausgegangen. Der eigentliche Massenmarkt wird erst in etwa 10 Jahren erwartet, nach Schätzungen soll es dann Trillionen (10**18) von Dokumenten mit „nützlicher" Information im Internet geben.

Nach Bild 2 treffen sich auf dem elektronischen Markt Informationsanbieter und Informationssuchende zum Abruf von Fakten oder spezialisierten Angeboten. Heute wird das Internet von Unternehmen bevorzugt für die Informationsbeschaffung und den Support verwendet. Durch die heterogene Netzstruktur des Internets ist die Plattformunabhängigkeit der Dienste eine unabdingbare Forderung für diesen Markt. Diese wird durch das Hypertext Transfer Protokoll und das darunterliegende TCP/IP Protokoll zwischen Client und Server erreicht. Die benötigte Software ist mittlerweile für viele Rechner und Betriebssysteme verfügbar.

Die Akzeptanz und der erwartete wirtschaftliche Nutzen werden im wesentlichen von zwei Kriterien abhängig sein:

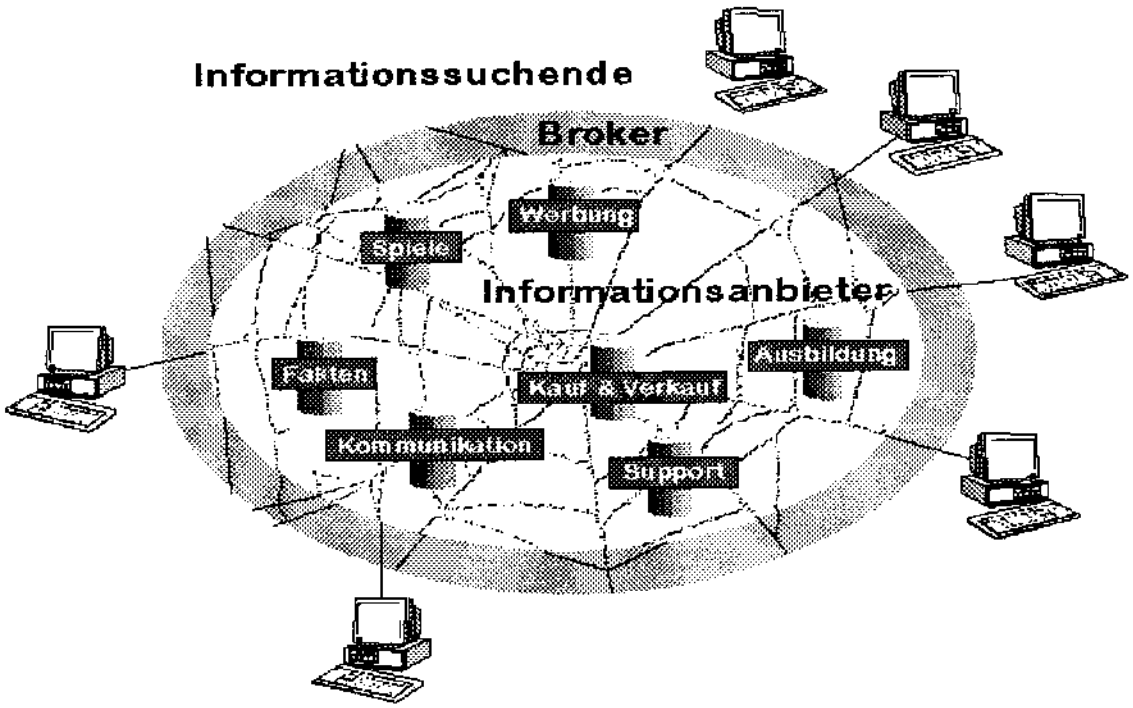

Bild 2: Elektronischer Markt im Internet

- Wie können die benötigten Informationen mit möglichst geringem Aufwand herausgefiltert werden?
- Wie können elektronische Dokumente möglichst interessant präsentiert werden und welche Interaktionen sind möglich?

Insbesondere für den letzten Punkt verspricht die Entwicklung von Java, wesentliche Akzente zu setzen. Ein Beispiel für eine attraktive Anwendung zeigt Bild 3. In einem Ticker laufen die aktuellen und die vorherigen Börsenwerte durch. In einer Grafik wird die Kursentwicklung ausgewählter Kurse angezeigt. Die Darstellungen folgen der aktuellen Börsenentwicklung. Ein Verweis führt auf ein Dokument, das den Kauf und Verkauf von Aktien ermöglicht.

kument. Die Grundidee dabei ist, daß ein Dokument aus Inhalt, Struktur und Layout (Darstellung) besteht. Die Struktur wird durch Markierungen (engl. tag) eingebracht. Dabei wird zwischen einer Anfangs- und einer Endemarkierung unterschieden. Sie haben die allgemeine Form:

Anfangsmarkierung:	<Name Attribut = Wert>
Endemarkierung:	</Name>

Jedes HTML-Dokument beginnt mit der Markierung <HTML> und endet mit </HTML>. Weitere Markierungen geben z.B. den Titel (<Titel>) eines Dokuments an, oder mit ihnen können mehrstufige Kapitelüberschriften (<H1>, <H2>, ..) gestaltet werden.

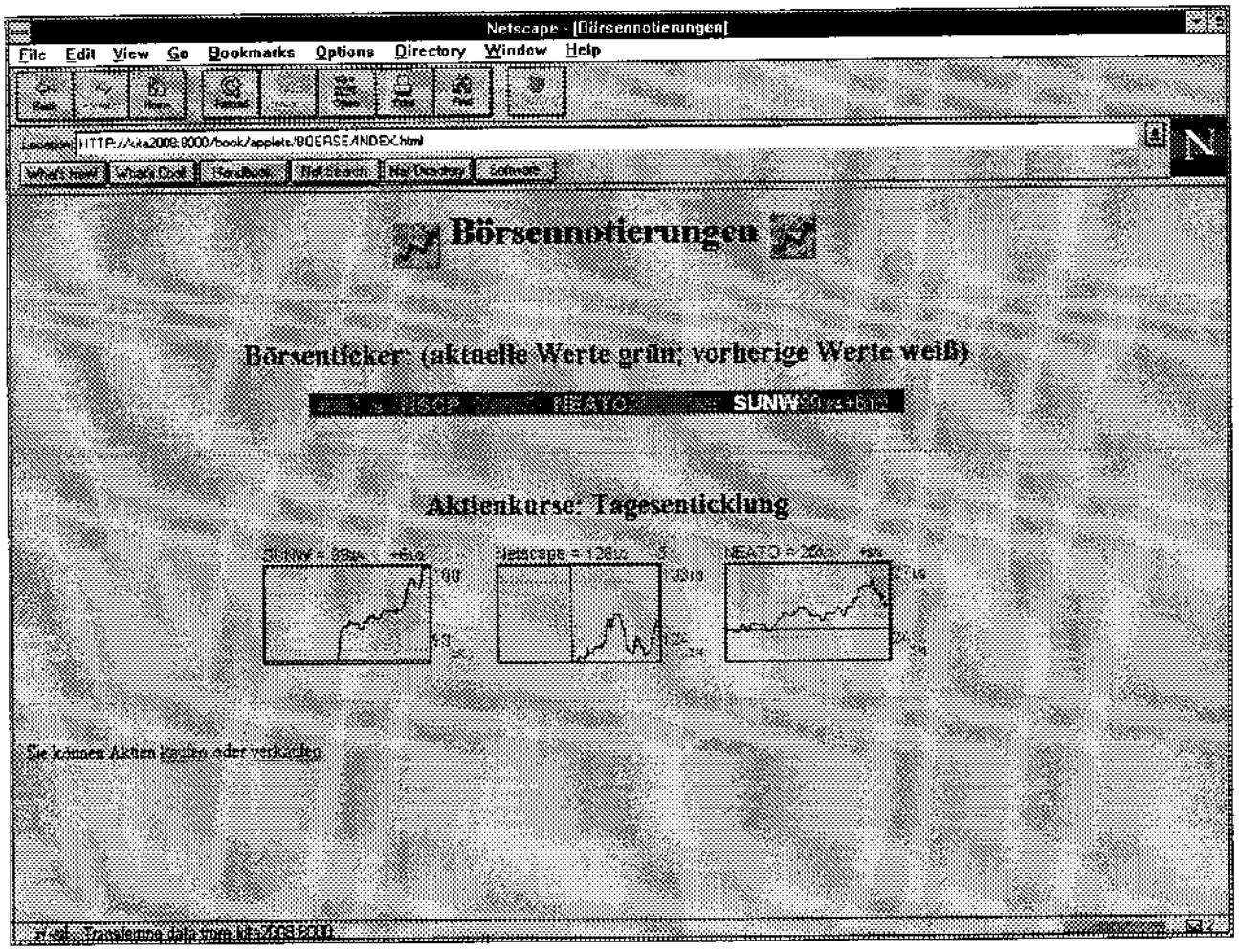

Bild 3: Börsenanwendung

2 Verwendung von Java

2.1 Integration in ein Dokument für das WWW

Dokumente im WWW werden mit Hilfe der Hypertext Markup Language (HTML) aufgebaut: man spricht deshalb auch von einem HTML-Do-

Von jedem WWW-Server können mit einem Browser HTML-Dokumente abgerufen und dargestellt werden. Eine Ressource, wie ein HTML-Dokument, kann im Internet weltweit eindeutig identifiziert werden. Diesen Identifikator bezeichnet man als Uniform Resource Locater (URL). So könnte sich zum Beispiel das HTML-Dokument für die fiktive Börsenanwendung aus Bild 3 unter folgender URL befinden:

http://www.boerse.com/boerse.html

Auch Bilder können durch eine URL referenziert und in ein HTML-Dokument mit der Markierung

```
<IMG SRC="url">
```

eingebunden werden. Für die Überschrift der Börsenanwendung ergibt sich folgender HTML-Code:

```
<CENTER>
<H1>
<img src="images/finance.gif" width=40      height=40
        align=absmiddle>
B&ouml;rsennotierungen
<img src="images/finance.gif" width=40      height=40
        align=absmiddle>
</H1>
</CENTER>
```

Die Überschrift ist zentriert, links und rechts der Überschrift sind Symbolbilder angeordnet, deren Größe angegeben ist und die durch eine URL im WWW identifiziert werden. Die Umlautdarstellung im Wort „Börsennotierung" entspricht der ISO-Norm 8879.

Mit Java ist es nun zusätzlich möglich, Programme in ein HTML-Dokument einzubinden. Diese werden als Applets bezeichnet und durch eine zusätzliche Markierung integriert. Die allgemeine Form einer Applet-Markierung besteht aus Standard-Attributen und applet-spezifischen Parametern. Diese Parameter bestehen immer aus zwei Attributen, wobei das erste den Namen des Parameters und das zweite dessen Wert spezifiziert. Der allgemeine Aufbau eines Applets ist:

```
<APPLET Standard-Attribute>
<PARAM NAME=Name-Attribut-1 VALUE=Value-
        Attribut-1>
...
...
<PARAM NAME=Name-Attribut-n VALUE=Value-
        Attribut-n>
</APPLET>
```

Für den StockTicker der Börsenanwendung ergibt sich folgendes Applet:

```
<APPLET CODE=StockTicker.class
        WIDTH=500 HEIGHT=20>
<PARAM NAME=data VALUE=tickerdata>
</APPLET>
```

Hier werden die drei notwendigen Standard-Attribute (CODE, WIDTH, HEIGHT) belegt. Der CODE gibt den Bezeichner des Applets an, die Größe des Applets soll dabei 500 * 20 Pixel betragen. Der applet-spezifische Parameter data gibt die Dateibezeichnung für aktuelle Börsennotierungen an.

Mit zusätzlichen Standard-Attributen kann z.B. die Positionierung des Applets im HTML-Dokument oder das Verzeichnis des Applets durch eine eigene URL angegeben werden. Standardmäßig wird als Verzeichnis des Applets das des HTML-Dokuments verwendet, auf dieses kann auch relativ Bezug genommen werden.

2.2 Programmiersprache Java

Java geht auf das Design einer Programmiersprache für elektronische Konsumartikel zurück, die James Gosling und sein Team 1990 im Unternehmen Sun entwickelten. Die Anforderungen bezüglich der Zuverlässigkeit, Bedienbarkeit, Sicherheit und Kosten an elektronische Konsumartikel, wie Spülmaschine, Fernseher, Videorecoder usw., sind teilweise wesentlich höher als bei reinen Software-Programmen, da in einem Fehlerfall höhere Regreßansprüche gestellt werden können und im Extremfall die gesamte Modellreihe vom Markt zurückgenommen werden muß. Auf diesem Markt unterliegt die Hardware schnellen Innovationszyklen. Die zu entwickelnde Programmiersprache sollte deshalb hardwareunabhängig sein. Aufgrund dieser Anforderungen schied nach ersten Experimenten C++ als Basissprache aus. Die neu entwickelte Sprache wurde Oak, nach einer Eiche vor Goslings Zimmer, genannt. Später zeigte sich, daß aus urheberrechtlichen Gründen der Name Oak nicht verwendet werden durfte. Nach langen Diskussionen im Entwicklungsteam, die oft in der Cafeteria geführt wurden, kam dort die Inspiration auf, die Sprache entsprechend dem Kaffee „Java" zu nennen.

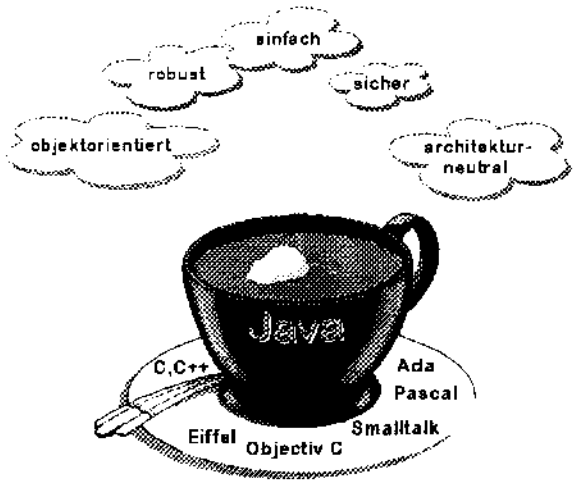

Bild 4: Fundament und Grundideen der Programmiersprache Java

Der erste Einsatz der Sprache erfolgte im Green Projekt, das zum Ziel hatte, neue Benutzungsschnittstellen für elektronische Konsumartikel zu ermöglichen. Aber die Entwicklung schlug fehl. Auch ein weiterer Versuch, das System bei Time Warner einzuführen, scheiterte. Als Mitte 1994 das WWW die ersten Erfolge verzeichnete, kam die Idee auf, Java als Internet-Programmiersprache zu erschließen. Die Konzepte von Java erschienen für

offene Systeme in einem unsicheren Netzwerk gut geeignet zu sein, da Java auf einem soliden Fundament bekannter Programmiersprachen aufbaut und konsequent nach den Grundideen wie Einfachheit, Robustheit, Sicherheit, Objektorientiertheit und Architekturneutralität entwickelt wurde. Die Sprache wurde um internetspezifische Anforderungen erweitert und ein Web-Browser mit Java entwickelt, der HotJava genannt wird und seit April 1995 verfügbar ist.

Die Syntax der Sprache wurde an C/C++ angelehnt, so daß diese für viele Programmierer vertraut erscheint. Einen Eindruck über die Syntax gibt Bild 5 mit einer Variante in Java für das bekannte „Hello World" Programm.

```
class HelloWorld {
 static public void main (String args [ ]) {
 system.out.println("Hello world");
 }
 }
```

Bild 5: C/C++ ähnliche Syntax am Beispiel des Programms „HelloWorld"

Viele Sprachkonstrukte von C/C++ werden jedoch häufig von Programmierern falsch angewendet oder führen zu einem unübersichtlichen oder gar fehlerhaften Programm. Aufgrund der Forderung nach Einfachheit, Robustheit und Sicherheit wurden viele Sprachelemente nicht übernommen. Deshalb ist Java auch eine einfach zu erlernende Sprache.

abstract	boolean	break	byte
byvalue *	case	cast *	catch
char	class	const *	continue
default	do	double	else
extends	final	finally	float
for	future *	generic *	goto *
if	implements	import	inner *
instanceof	int	interface	long
native	new	null	operator *
outer *	package	private	protected
public	rest *	return	short
static	super	switch	synchronized
this	throw	throws	transient
try	var *	void	voltile
while			

Java enthält in der Version 1 die im folgenden aufgeführten 57 reservierten Schlüsselwörter. Die Schlüsselwörter werden jedoch derzeit nicht alle benutzt, die nicht verwendeten sind mit einem * gekennzeichnet.

2.2.1 Datentypen und Operatoren

In Java werden folgende einfache Grunddatentypen unterschieden:

numerische Datentypen
Ganze Zahlen
- Byte: 8-Bit
- short: 16-Bit
- int: 32-Bit
- long: 64-Bit

Gleitkommazahlen
- float: 32-Bit
- double: 64-Bit

boolischer Datentyp
- boolean: Wertebereich {true,false}

Datentyp Zeichen
- char: 16-Bit Unicode

numerische Datentypen
Bei den numerischen Datentypen werden ganze Zahlen mit einer Breite von 8, 16, 32 und 64 Bit sowie Gleitkommazahlen mit 32 und 64 Bit unterschieden. Im Gegensatz zu der Sprache C ist bei Java die Anzahl der verwendeten Bits nur abhängig vom Datentyp, nicht von der eingesetzten Plattform.

boolischer Datentyp
In der Sprache C ist es gängige Praxis, einen boolischen Datentyp mit den Werten *true* oder *false* aus einem anderen Datentyp zu definieren. Dies führt oft zu Kompatiblitätsproblemen. Aufgrund der Anforderung nach Robustheit, Sicherheit und Architekturneutralität ist in Java ein boolischer Datentyp definiert, der nicht in einen anderen Datentyp gewandelt werden kann. Dazu kommt, daß auch der neue ANSI-Entwurf für C++ den boolischen Datentyp enthält.

Datentyp Zeichen
Die internationale Ausrichtung des Einsatzes der Sprache Java verlangt, daß neben Ziffern und dem Alphabet auch alle landessprachspezifischen Zeichen unterschieden werden können. Eine Repräsentation mit 8-Bit und damit 256 unterschiedlichen Zeichen, wie z.B. bei der Sprache C realisiert, ist dafür nicht ausreichend. Deshalb wird in Java der Unicode verwendet, der eine Breite von 16 Bit besitzt.

Die in C/C++ verwendete Praxis, Datendeklarationen in globalen Headerdateien vorzunehmen, um diese dann in vielen Daten zu inkludieren, ist in Java nicht möglich. Diese einfache Handhabung führt bei komplexen Programmen zu unübersehbaren Deklarationen sowie Redeklarationen von

Symbolen und ist deshalb aus Gründen der Robustheit und Sicherheit zu verwerfen. Ein Beispiel für eine Redeklaration mit fatalen Folgen ist:

#define private public

In diesem Fall würden alle folgenden mit dem Zugriffsschutz *private* definierten Bezeichner den Zugriffsschutz *public* erhalten.

Zur Strukturierung von komplexen Datenstrukturen werden Klassen verwendet. So werden zum Beispiel Zeichenketten und auch Arrays durch Klassen implizit realisiert. Dadurch können die Bereichsgrenzen überwacht werden, und es ist sichergestellt, daß sie nicht überschritten werden. Aufgrund des Sicherheitsaspekts ist die in der Sprache C übliche und fehleranfällige Adressierung von Arrays und anderen Datenstrukturen durch Zeiger nicht möglich. Die Realisierung einer Klasse wird in Abschnitt 2.2.3 noch näher erläutert. Java verwendet im wesentlichen die gleichen Operatoren wie die Sprache C.

2.2.2 Kontrollstrukturen

Die den Ablauf des Programms steuernden Kontrollstrukturen umfassen den von C oder anderen dritten Generationssprachen gewohnten Umfang mit geringen Modifikationen.

Zur Auswahl von Sequenzen gibt es die *if-then-else* und die *switch* Struktur. Bei der *if-then-else* Struktur ist der Hauptunterschied zu C, daß in Java die zu testende Bedingung einen boolischen Wert ergeben muß. Numerische Werte, wie null oder eins, sind nicht erlaubt. Die *switch* Struktur ist mit der in C identisch.

Für die Kontrollstruktur „Wiederholung" gibt es verschiedene Formen. Es werden wie in der Sprache C zwei *while* Schleifen und eine *for* Schleife unterschieden. Bei der Initialisierung der *for* Schleife können ergänzend auch Variablen definiert werden. Diese Variablen existieren nur innerhalb der Schleife.

Gegenüber C gibt es zusätzlich die Möglichkeit, mit den Statements *continue* bzw. *break* in Verbindung mit einer Marke (Label), mehrfach geschachtelte Schleifen oder Auswahlstrukturen definiert zu verlassen oder fortzusetzen. Von vielen C-Programmierern wird zur Realisierung einer solchen Konstruktion das *goto* Statement eingesetzt, mit dem jedoch auch andere Sprungziele erreicht werden können. Dies führt zu einer fehlerträchtigen Situation. Das *goto* Statement ist in der Version 1 von Java nicht realisiert, obwohl es sich um ein reserviertes Schlüsselwort handelt.

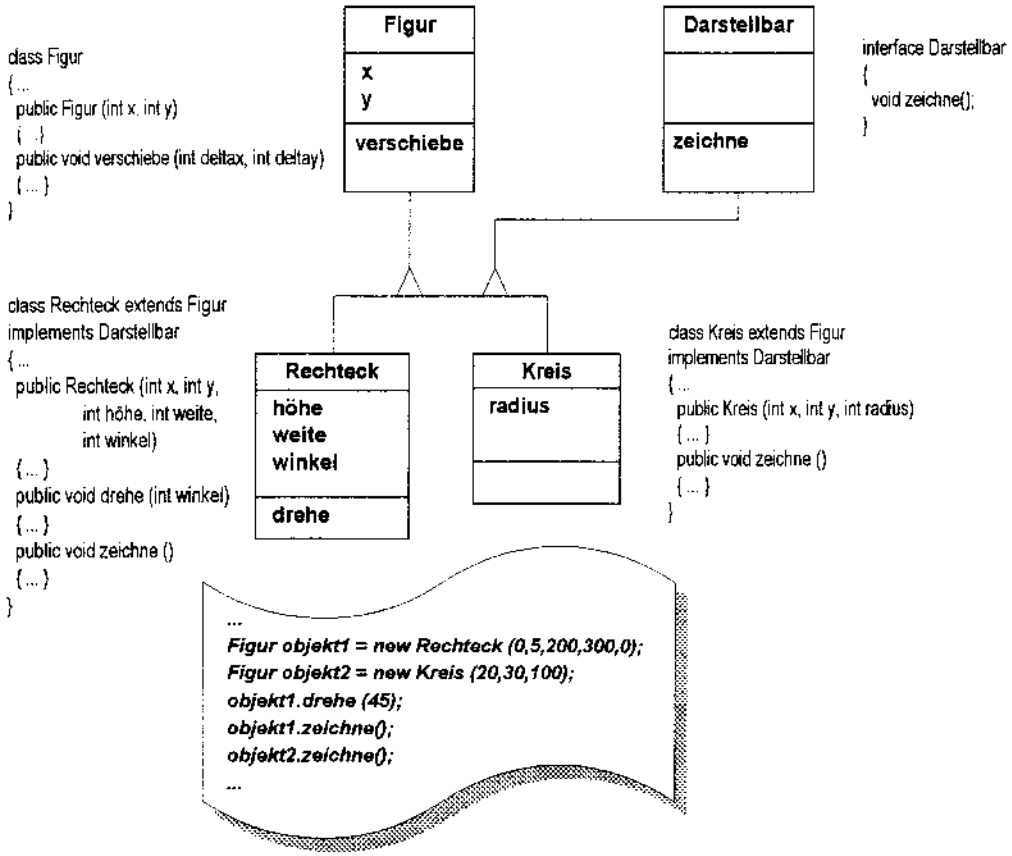

Bild 6: Beispiel für einen objektorientierten Entwurf und die Implementierung in Java

Das wichtigste Konzept zur Modularisierung in objektorientierten Systemen ist die Klassenbildung. Eine Klasse definiert die Attribute und die Operationen ihrer Objekte. In einem Grafikprogramm gibt es zum Beispiel Klassen für unterschiedliche grafische Figuren, wie Rechtecke, Kreise usw. Die Lage jeder Figur ist unabhängig von ihrer konkreten Ausprägung und läßt sich in einer eigenen Klasse *Figur* durch eine x- und y-Koordinate beschreiben. Die beiden Koordinaten bezeichnet man als Attribute der Klasse *Figur*. Die Klasse *Rechteck* besitzt zusätzlich die drei Attribute *höhe* , *weite* und *winkel* bzw. die Klasse *Kreis* das Attribut *radius*. Die Anwendung kann die Verschiebbarkeit der Figuren erfordern. Dafür enthält die Klasse *Figur* die Methode *verschiebe*. Die Orientierung eines Rechtecks kann durch die Methode *drehen* geändert werden. Die Attribute und die Methode der Oberklasse *Figur* werden an die beiden Unterklassen, *Rechteck* und *Kreis*, vererbt. Zusätzlich zu diesen Methoden besitzen alle Klassen einen Konstruktor, um Objekte - das heißt Instanzen der jeweiligen Klasse - zu erzeugen. Der Konstruktor besitzt den gleichen Namen wie die Klasse.

Weiterhin erfordert die Anwendung eine grafische Visualisierung. Je nachdem, ob ein Rechteck oder ein Kreis darzustellen ist, werden andere spezialisierte Ausgabemethoden benötigt. Beide Klassen müssen deshalb eine Methode zur Darstellung enthalten. Im Gesamtsystem sind nicht nur Kreise und Rechtecke darzustellen, sondern z.B. auch Schriften. Um jedoch einen gemeinsamen Bezug zu ermöglichen, ist eine weitere Oberklasse *Darstellbar* einzuführen. Wie beschrieben erfolgt die Implementierung der Methoden dieser Klasse in der Unterklasse. Eine solche abstrakte Klasse wird in Java als Interface bezeichnet. Sie ermöglicht Mehrfachvererbung in einem bewußt eingeschränkten Rahmen. Dieses Konzept wurde von Objective C übernommen.

Die allgemeine Mehrfachvererbung wird in Java nicht unterstützt, da sie vielfach nur schwer zu verstehen und fehleranfällig ist. Existieren z.B. in beiden Oberklassen gleichnamige Attribute bzw. Methoden, so ist bereits der Vererbungsmechanismus nicht eindeutig.

Inhaltlich zusammengehörige Klassen, die gegebenenfalls über mehrere Dateien verteilt sind, können in einer Klassenbibliothek zusammengefaßt werden. Diese bezeichnet man wie in der Sprache Ada als *package*. Andere Klassen können dann einfach auf solche Pakete Bezug nehmen. Eine Reihe von Paketen sind in Java bereits verfügbar, etwa für die Ein- und Ausgabe, für die Gestaltung einer grafischen Benutzungsoberfläche, für mathematische Operationen und Netzwerkfunktionen.

2.2.4 Weitere Konzepte

Multithreading
Interaktive multimediale Anwendungen erfordern, daß viele Aufgaben quasi gleichzeitig vom Rechner bearbeitet werden. Zu einer Animation soll zum Beispiel gleichzeitig eine Audiowiedergabe erfolgen, während die Bildschirmseite weiter bewegt und eine Datei von einem Server heruntergeladen wird. Zur Lösung dieser Aufgabe ist ein Mechanismus notwendig, mit dem aus der Anwendung heraus mehrere Teilprozesse erzeugt und synchronisiert werden können; dies wird als Multithreading bezeichnet. Ein Paket zur Unterstützung dieses Mechanismus ist bereits in Java enthalten und die notwendigen Konstrukte gehören zum Sprachumfang.

Speichermanagement
Java tätigt das Speichermanagement vollständig automatisch. Es stellt dem Programmierer keine Funktionen zur expliziten Reservierung und Freigabe von Speicherplätzen zur Verfügung. Es können nur Daten und Objekte mit den vorgestellten Konstrukten bereitgestellt werden. Es existieren also nur Konstrukte zum Anlegen von Daten und Objekten, die implizit Speicherplatz belegen. Das automatische Freigeben von nicht mehr benötigtem Speicherplatz (Garbadge Collection) ist integraler Bestandteil von Java und seinem Laufzeitsystem. Die Freigabe erfolgt in einem Thread mit niedriger Priorität, wobei hierzu insbesondere die in einer interaktiven Anwendung häufig vorhandenen Pausen genutzt werden. Das automatische Speichermanagement vereinfacht die Programmierung und eliminiert einen ganzen Bereich von Fehlermöglichkeiten.

Ausnahmebehandlung
In vielen Programmiersprachen der dritten Generation werden Laufzeitfehler auf unterschiedliche Art behandelt, wobei die konkrete Realisierung noch zusätzlich vom Programmierer abhängig ist. Zum Beispiel wird beim Zugriff auf die Adresse null entweder die Anwendung mit einer Systemfehlermeldung abgebrochen oder aber das gesamte System gerät in einen unerlaubten Zustand und stürzt ab, je nachdem, welches Betriebssystem genutzt wird. Weiterhin wird meist zur Signalisierung bestimmter Ausnahmesituationen in der Anwendung ein bestimmter Rückgabewert zwischen aufrufender und aufgerufener Funktion vereinbart. Diese Beispiele zeigen, daß die Behandlung von Ausnahmesituationen nicht konsistent erfolgt und oft zu einem unübersichtlichen Quellcode

führt. Wie in der Sprache C++ wird die Behandlung von Ausnahmesituationen durch Sprachelemente (try, catch, throw) in Java unterstützt. Dies führt zu einer übersichtlichen und einheitlichen Vorgehensweise sowie zu dem geforderten robusten Verhalten.

Architekturneutralität
Die problemlose und sofortige Verfügbarkeit eines Java-Programms auf den unterschiedlichsten Plattformen durch einfaches Herunterladen von einem Server erfordert in hohem Maße eine Architekturneutralität. Die implementierte Unabhängigkeit im Sprachumfang, wie zum Beispiel des Datentyps, wurde bereits erwähnt. Die Forderung nach sofort verfügbarer Ausführung auf unterschiedlichen Systemarchitekturen erfordert einen interpretierenden Modus. Bekanntlich ist aber das Antwortzeitverhalten bei einer Anwendung, die mit einer zu interpretierenden Sprache implementiert ist, schlecht. Um dieses Manko zu umgehen, übersetzt der Java-Compiler das Programm in einen Bytecode. Vereinfacht beschrieben (s. Abschnitt Sicherheit), wird der Bytecode durch eine virtuelle Maschine ausgeführt. Die Möglichkeit, Java-Programme auf einer beliebigen Plattform auszuführen, wird damit nur durch die Verfügbarkeit der virtuellen Maschine auf der Zielplattform beschränkt. Der Bytecode ist zum einen architekturneutral und eröffnet damit eine neue Form der Software-Distribution. Zum anderen ist der Code effektiv zu interpretieren. Nach Angaben der Firma Sun verlängert sich die Ausführungszeit etwa um den Faktor 10 bis 20 gegenüber einer reinen C-Code Entwicklung. Mit einer Optimierung des Java-Compilers wird erwartet, daß die Ausführungszeit beschleunigt wird. Zusätzlich ermöglicht eine Option, den Bytecode in Maschinencode zu übersetzen, der - abgesehen von den durchgeführten Laufzeitüberprüfungen (z.B. Arraygrenzen) - fast so schnell ist wie bei einer reinen C-Entwicklung. Eine weitere Option ermöglicht auch das Einbinden von C-Bibliotheken; die Architekturneutralität geht bei Verwendung dieser Option allerdings verloren.

Sicherheit
Aufgrund der Ausrichtung von Java auf das Einsatzgebiet in einem unbekannten vernetzten, heterogenen Umfeld sind viele Sicherheitsmechanismen Bestandteil von Java. Gemäß des Sicherheitskonzeptes sind zwei Ebenen zu unterscheiden. Zum einen die Sprachebene, die es nicht ermöglicht, auf unerlaubte Ressourcen zuzugreifen. Insbesondere der Verzicht auf Zeiger ist hier zu nennen. Zum anderen ist die externe Ebene zu betrachten. Sie gibt an, wie die Programme von einem Internet-Server herunter in den eigenen Computer, den Client, geladen werden sowie in welcher Umgebung und mit welchen Zugriffsrechten die Programme dort ausgeführt werden. Für die erste Sicherheitsüberprüfung werden drei Bereiche unterschieden: der eigene Rechner, eine spezifizierte Grenze (Firewall eines Unternehmens) und, wie im Bild 7 dargestellt, das gesamte Internet. Mit dieser Abgrenzung werden die Bereiche spezifiziert, auf die während der Ausführung eines Java-Programms zugegriffen werden kann. Diese Grenze kann durch den Programmierer, den Anwender oder den Systemadministrator festgelegt werden.

Für den Quellcode einer beliebigen Klasse ergibt sich damit folgendes Bild: Der Quellcode wird auf einem Rechner erstellt und als Datei mit der Extension *java* gespeichert, diese wird auf dem gleichen oder einem anderen Rechner compiliert und dann als Bytecode in einer Datei mit der Extension *class* abgelegt. Diese Datei kann in einen Rechner mit einer Java-Laufzeitumgebung geladen und ausgeführt werden. Bei der Laufzeitumgebung kann es sich um einen Java-Interpreter oder einen java-kompatiblen Browser handeln. Im ersten

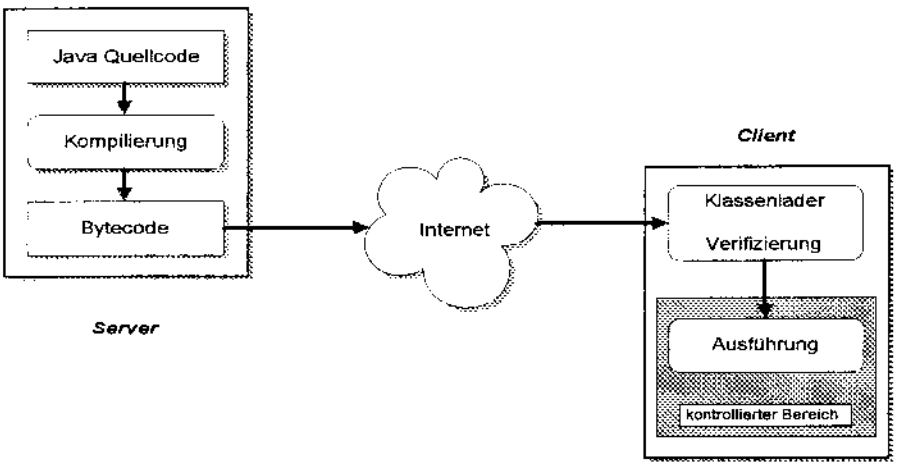

Bild 7: Verifikationsprozeß beim Laden und Ausführen eines Programms

Fall wird als Bytecode ein selbständiges Programm erwartet und ausgeführt, dagegen bei der Ausführung im Browser ein Applet. Der Unterschied besteht darin, daß der Programmrahmen des Browsers bei einem Applet verwendet wird, bei einem Programm dagegen explizit vorhanden sein muß. Ein Programm bzw. ein Applet wird klassenweise in die Laufzeitumgebung geladen. Dabei gilt mit den definierten Sicherheitsgrenzen folgende Regel: Eine unsichere Klasse kann eine sichere Klasse nicht überschreiben. Das heißt, existiert die benötigte Klasse auf dem eigenen Rechner, wird stets diese verwendet. Konnte der angeforderte Bytecode einer Klasse geladen werden, so wird er dem Verifizierer zugeführt. Dabei wird der Bytecode hinsichtlich seiner Orginalität verifiziert. Es wird überprüft, ob der Bytecode entweder mit einem Java-Compiler, der die Orginalspezifikationen erfüllt, oder mit einem modifizierten Compiler erstellt wurde, der gegebenfalls bestimmte Sicherheitsmechanismen der Sprachebene verletzt.

Die Überprüfung erfolgt anhand der Stackverwaltung, der Zugriffsrechte (private, public, protected), der symbolischen Namen von Variablen und Funktionen sowie deren Typ. Bei Java werden aus Sicherheitsgründen alle Typinformationen berücksichtigt, sogar über das für die Interpretation notwendige Maß hinaus. Erst danach erfolgt die Transformation auf Programmadressen, so daß nach der Verifikation der Zugriff, ähnlich wie bei einem herkömmlichen Programm, effizient durch Adressen erfolgen kann. Das Programm arbeitet jedoch in einem zuvor kontrollierten und abgegrenzten Bereich.

Mit seiner Funktionalität stellt der Verifizierer den kritischen Punkt zur Gewährleistung der Sicherheit dar, seine Echtheit ist deshalb vom Anwender sicherzustellen. Für die Version 1.1 von Java ist eine Verifikation durch ein Public Key Krytoverfahren vorgesehen.

3 Resümee und Perspektiven

Die Etablierung von Java im Internet ist trotz seiner erst kurzen Geschichte in vollem Gange. Erst vor einem Jahr, im April 1995, wurde Java vorgestellt und einen Monat später, auf der SunWorld95, wurden Weiterentwicklungen angekündigt. Im Oktober erfolgte dann die Freigabe des Beta-Releases und Anfang dieses Jahres die der Version 1.0. Das Java Developer Kit ist für die Betriebssysteme Solaris, Windows 95 und Windows NT kostenlos unter Beachtung der Lizenzbedingungen auf dem Sun-Server zu erhalten. Zusatzprodukte wie visuelle Entwicklungswerkzeuge, Schnittstellen zu Datenbanken oder zur Common Object Request Broker Architecture (CORBA) sind von Sun und anderen Firmen bereits verfügbar bzw. angekündigt. Als Browser stehen HotJava von Sun, Navigator von Netscape und Internet-Explorer von Microsoft zur Verfügung. Die mittlerweile bekanntgewordenen Mängel in den Sicherheitsmechanismen von Java sind nicht auf Fehler in der Konzeption, sondern bei der Implementierung zurückzuführen. Diese konnten teilweise schon behoben werden.

Das Unternehmen Sun hat Anfang des Jahres die Firma JavaSoft mit ca. 70 Mitarbeitern für eine breitere Unterstützung und Markteinführung von Java gegründet. Die Einschätzung von praktisch allen Unternehmen der Informationstechnik über die Zukunft von Java sind sehr positiv. Führende Firmen haben Lizenzrechte an Java erworben. Das Lizenzmodell von Sun für Java ermöglicht Computerherstellern, Schnittstellen für Java in ihre Betriebssysteme zu integrieren. Die Lizenznehmer erhalten das „Java Development Kit (JDK)", einen Java-Interpreter und verschiedene Klassenbibliotheken. Unternehmensspezifische Erweiterungen der Programmierschnittstellen müssen bekanntgegeben und nach einer bestimmten Zeitspanne fremde Erweiterungen in die eigenen Java-Interpreter eingebaut werden. Zudem ist jeder Lizenznehmer verpflichtet, mit der eigenen Installation die Anforderungen der Test-Suite von JavaSoft zu erfüllen. Damit ist sichergestellt, daß die einzelnen Implementierungen immer untereinander kompatibel bleiben. So hat z.B. IBM eine Java-Schnittstelle bereits für das im Sommer erscheinende Betriebssystem Merlin, dem OS/2 Nachfolger, angekündigt. An Implementierungen für Windows 3.1 und erstaunlicherweise MVS wird bei IBM gearbeitet. Apple sieht entsprechende Schnittstellen unter anderm für das Betriebssystem Newton in dem Personal Digital Assistant vor. Weitere Hersteller sind: HP, Hitachi, Microsoft, Netscape, Novell, Silicon Graphics, SCO, Tandem u.a.

Neben dem „Java Development Kit" werden von verschieden Firmen Entwicklungsumgebungen für Java bereits angeboten bzw. entwickelt. Dabei handelt es sich meist um modifizierte C++ Entwicklungsumgebungen. So bietet Rogue Wave das Product Zapp Factory und Borland C++ 5.0 an. Die Entwicklungsumgebung von Borland enthält einen sogenannten Applet-Accelerator, dies ist ein Just-in-time-Compiler, der das Applet während des Download bereits kompiliert und in eine EXE-Datei oder eine Runtime-DLL für Windows 95 bzw. NT umsetzt. Damit soll eine zehnfache Steigerung der Ausführungszeit erreicht werden. Sunsoft bietet unter dem Namen Java Workshop ein in Java implementiertes Entwicklungswerkzeug an. Die Fertigstellung ist im vierten Quartal 1996 vorgesehen. Die Firma O/Space hat bei ihrer Entwicklung den Schwerpunkt auf eine objektorientierte

Architektur zum Aufruf verteilter Objekte in einer heterogenen Umgebung gelegt und Java in die Corba Technologie integriert. Auch Sun sieht die Integration von Java in ihre Corba-konforme NEO-Produktfamilie vor.

Mit Java wurde dem Internet eine neue Dimension verliehen. Zuvor standen nur statische oder mit geringer Interaktivität versehene WWW-Dokumente zur Verfügung. Nun können die Dokumente praktisch beliebig komplexe Anwendungen beinhalten, die über das Internet sicher bezogen und im Client ausgeführt werden. Während der Ausführung sind Rückantworten an den Server oder das Laden von weiteren Applets vom gleichen bzw. einem anderen Server möglich. Durch die praktisch universelle Anwendbarkeit ergeben sich aber auch Perspektiven für den innerbetrieblichen Einsatz, für die der Begriff Intranet-Anwendungen geprägt wurde. Hier führt der Einsatz dieser Technologie zu einer Basisausstattung, die aufgrund der absehbar hohen Verbreitung eine große Stabilität aufweist und kostengünstig bezogen werden kann. Zudem können Erweiterungen für eine erste Präsenz im WWW dann relativ einfach vollzogen werden. Dabei wird das Marktvolumen von Insidern für Intranet-Anwendung sogar wesentlich höher eingeschätzt, als für reine Internet-Anwendungen. Es wird auch damit spekuliert, daß der Einsatz von Standardsoftware revolutioniert werden kann. Der derzeitige Erwerb und die Installation von Standardsoftware soll durch den bedarfsabhängigen Bezug über das Internet und eine Nutzungsgebühr ersetzt werden. Eine weitere Optimierung kann durch den Einsatz von Java-Chips zur Interpretation und Ausführung des Bytecodes erreicht werden. Als erster Halbleiterhersteller präsentierte Mitsubishi Electronics America einen Multimedia-RISC-Mikroprozessor mit integriertem Java-Interpreter. Das System soll zur Konstruktion von Personal Digital Assistants und bei Navigationssystemen eingesetzt werden. Weitere Einsatzbereiche werden im Telekommunikationsbereich, im Bau von einfachen und billigen Netzcomputern bis hin zu hochwertigen 3D-Workstations gesehen.

Konkurrenzentwicklungen hat insbesondere Microsoft angekündigt. Dort wurden ähnliche Entwicklungen unter den Codenamen Blackbird, Sweeper und Internet Information Server betrieben. Ein neues Produkt, Active X, ist seit März dieses Jahres verfügbar. Leben in die statischen Web-Seiten wird ebenfalls durch die Shockwave-Technologie des Multimedia-Herstellers Macromedia gebracht.

Aufgrund der großen Anwendungsbreite, der guten Konzeption von Java sowie den bereits jetzt vorliegenden umfangreichen Klassenbibliotheken (packages) und nicht zuletzt durch die erwarteten hohen Chancen von multimedialen Anwendungen im Inter- und Intranet ist von einer schnellen Verbreitung auszugehen. Als einziger Engpaß werden die derzeitigen Bandbreiten im WWW gesehen. Mit der Perspektive des WWW als breiter Markt für elektronische Dienstleistungen hat Java das Potential, den Konsummarkt, für den Java ursprünglich entwickelt wurde, zu erobern. Dementsprechend euphorisch sind die Meldungen in der bisher kurzen Geschichte von Java im WWW. Die Erfolgsstory wird praktisch in allen Medien aufgegriffen, die Breite der Berichterstattung über eine „Programmiersprache" ist bisher einmalig.

Als Systemanalytiker oder Programmierer soll uns das winkende Maskottchen von Java, Duke, an die möglichst intuitive und spielerische Gestaltung der Interaktionen mit HTML-Dokumenten erinnern.

Literatur

Rumbaugh, J; Blaha, M.; Premerlani, W.; Eddy, F.; Lorensen, W. (1993): Objektorientiertes Modellieren und Entwerfen. Hanser Verlag München u.a.

Hoff von, Artur; Shaio, Sami; Starbuck, Orca (1996): Hooked on Java. Creating Hot Web Sites with Java Applets. CD-ROM include. Addison-Wesley Publishing. Bonn u.a.

Java Report. The Source for Java Development. SIGS Conferences. *Zweimonatliches Magazin.*

Java Spektrum. SIGS Conferences. *Zweimonatliches Magazin.*

Online-Information

http://www.sun.com
Firmenserver von Sun
http://www.javasoft.com
Firmenserver von JavaSoft
insbesondere:
The Java Language Environment
und: The Java Language Specification
http://www.netscape.com
Firmenserver von Netscape
http://www.java-world.com.
Online Java-Journal
http://www.gamelan.com
Applet-Sammlung
http://sunsite.informatik.rwth-aachen.de
/Mirror/java
Java-Site in Deutschland