

J2ME: Zusammenfassung

Ausarbeitung im Rahmen des Proseminars Mobile Java
Universität Ulm
SS 2004

Irene Graz
irene.graz@informatik.uni-ulm.de

Zuständige Betreuer: Martin Gumhold, Dr. Frank Kargl.

Inhaltsangabe:

1 Einleitung	3
2 Konfigurationen: CDC, CLDC und ihre Profile	3
CDC	4
Virtual Machine: CVM	4
Profile von CDC:	5
Foundation Profile (FP).....	5
Personal Basis Profile & Personal Profile (PBP & PP).....	5
CLDC	5
Virtual Machine: KVM	5
Profile:.....	5
MIDP (Mobile Information Device Profile)	5
3 Optionale Pakete	6
J2ME Web Services (JSR* -172).....	6
Mobile Media API (JSR-135)	7
Mobile 3D Graphics (JSR-184).....	8
Java TV API	9
Bluetooth API (JSR-82)	9
Wireless Messaging API (JSR-135) und JavaPhoneAPI	9
4 Sicherheitsmechanismen in J2ME (CLDC/MIDP/JSR-177)	10
5 Entwicklungsumgebungen und Programmierpraxis	10
6 Ausblick	10
Literaturverzeichnis:	11
Abbildungsverzeichnis:	11

1 Einleitung

Java 2 Plattform, Micro Edition, abgekürzt J2ME, ist eine Umsetzung der Programmiersprache Java für Geräte mit sehr limitierten CPU- und Speicher-Ressourcen, so genannte »embedded consumer devices«. Sie existiert seit Juni 1999 und ist hauptsächlich für den mobilen Bereich wie etwa Mobiltelefone oder PDAs entwickelt worden.

Bei der Durchsetzung des Mottos von Sun „compile once, run everywhere“ sind die Entwickler von J2ME auf ein großes Problem gestoßen. Die Geräte, auf denen die Programme laufen sollen, sind von ihrer Ausstattung her sehr unterschiedlich. Sie können eine Tastatur, eine Netzwerkanbindung haben oder auch nicht, ein farbiges oder schwarz/weißes Display enthalten. Außerdem haben sie sehr unterschiedliche Hardwarekapazitäten, z. B. Speicher zwischen 10 KByte bis zu 16 MByte. Die Geräte können nach den Merkmalen Systemarchitektur, Leistungsfähigkeit und Anwendungsfeld aufgeteilt werden.

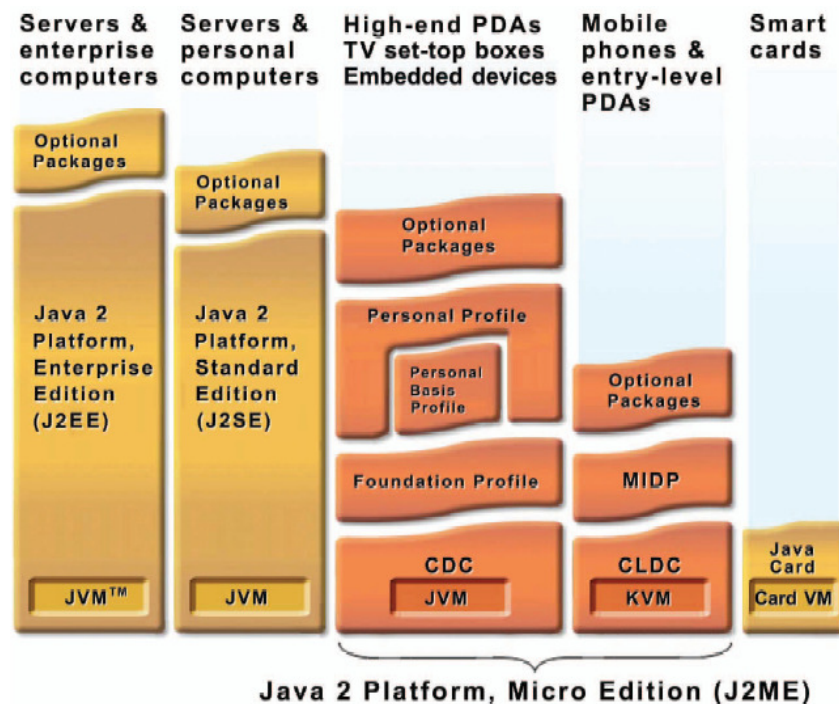


Abb. 1 Die Architektur von J2ME

Diese Problematik wurde durch die zusätzliche Einführung von Konfigurationen und Profilen innerhalb der Plattform gelöst, die je nach Bedarf angepasst werden können. Das Konzept ist, dass Java-Software auf beliebigen Geräten, die Java fähig sind, funktioniert, wenn sie der Spezifikation entspricht. Entwickler müssen Java-Software nicht jedes Mal anpassen, damit sie auf unterschiedlichen Endgeräten funktioniert. Die Programme laufen innerhalb einer "Virtual Machine" ab, die die spezifischen Unterschiede der Geräte vor der Software verbirgt.

2 Konfigurationen: CDC, CLDC und ihre Profile

Die Geräte werden zunächst ihrer Hardwaremöglichkeiten entsprechend einer der zwei Konfigurationen zugeteilt. Konfiguration von J2ME beinhaltet eine „Virtual Machine“ und einen Basissatz von Klassen, die Eingabe und Ausgabe, Vernetzung, Sicherheit und Internationalisierung ermöglichen. Zum Aufgabenbereich der Konfigurationen zählen nicht, sondern werden von den jeweiligen Profilen umgesetzt, Installation, Ausführung und Entfernen von Anwendungen, Benutzerschnittstelle und Ereignisbehandlung. Sie unterstützen die Grundfunktionalität der Geräte und legen die Mindestanforderungen an die Hardware der Geräte: Speicher-, Display-, CPU- und Netzwerk-Voraussetzungen fest. Zur Zeit existieren zwei Konfigurationen in J2ME: Connected Device Configuration und Connected Limited Device Configuration.

CLDC	CDC
Mindestens 160 KB Speicher	Mindestens 512 KB Speicher
Mindestens 32 KB Speicher für die Laufzeitumgebung	Mindestens 256 KB Speicher für die Laufzeitumgebung
Netzwerkverbindung basiert nicht auf TCP/IP	Netzwerkverbindung basiert auf TCP/IP
Instabile Netzwerkverbindung	Stabile Netzwerkverbindung
Endgerät wird von 16/32-Bit CPU betrieben	Endgerät wird von 32-Bit CPU betrieben
Endgerät darf mobile Stromversorgung haben	Endgerät hat keine mobile Stromversorgung
Verbindung zum Netzwerk muss vorhanden sein, eventuell drahtlos. Bandbreite darf unter 9600 Baud liegen.	

Abb. 2 Der Vergleich zwischen CLDC, CDC und J2SE

Die Einteilung in Konfigurationen reicht jedoch nicht aus, weil sich die Geräte auch innerhalb einer solchen Gruppe in vielen Punkten unterscheiden können. Zum Beispiel hat eine Waschmaschine und ein Handy die gleiche Hardwareausstattung, sie haben aber ganz unterschiedliche Funktionen. Um den Herstellern eine Möglichkeit zu geben Java 2 Micro Edition an ihre speziellen Eigenschaften anzupassen, existiert diese zusätzliche Profil-Ebene.

Ein Profil setzt auf einer Konfiguration auf und erweitert deren APIs. Man kann sagen, Profile dienen zur „vertikalen“ Unterteilung der einzelnen Marktsegmente. Der Hersteller kann im Laufe der Entwicklung eines Gerätes entscheiden, welche Profile von dem Gerät unterstützt werden sollen, muss dann aber auch alle dazugehörenden Merkmale umsetzen. Also wenn der Hersteller beschließt, die Hausfrauen zum Kauf seiner programmierbaren Waschmaschinen zu bewegen, indem er seine Geräte für das Telefonieren erweitert, muss er nur ein geeignetes Profil einbinden, damit die Software laufen kann.



Abb. 3 Aufteilung der Geräte auf die Konfigurationen

CDC

Die CDC ist für Geräte mit einer festen Verbindung über ein Netzwerk wie Bildtelefone, Set-Top-Boxen, Smart- Handys, größere PDAs, Autonavigationssysteme etc. gedacht. CDC ist einer Superklasse von CLDC und ist somit auch für CLDC Geräte einsetzbar. CDC beinhaltet die komplette Java 2 Plattform VM, nämlich die CVM (Customer Virtual Machine).

Virtual Machine: CVM

Die VM ist das zentrale Element, welches Java seine Plattformunabhängigkeit verleiht, indem sie als Schnittstelle zwischen Java Applications bzw. Applets und dem eigentlichen Betriebssystem fungiert. Die VM hat auf den Geräten der CDC Kategorie genug RAM für ihren Heap und eine genügende Prozessorleistung, die für eine flüssige und schnelle Befehlsverarbeitung sorgt.

Profile von CDC:

Foundation Profile (FP)

Das Foundation Profile ist das grundlegendste von den Profilen und baut direkt auf der CDC auf. Es erweitert die J2SE APIs, die bereits im CDC enthalten waren. Die Kombination CDC/FP enthält eine mächtige Umgebung, um allgemeine Anwendungen zu erstellen. Allerdings können damit keine interaktiven Anwendungen geschrieben werden, weil das Abstract Window Toolkit (AWT) fehlt. Für einige Geräte, wie z.B. Router, ist es von Vorteil. Auf dem FP bauen das Personal Basis Profile und Personal Profile auf.

Personal Basis Profile & Personal Profile (PBP & PP)

Das Personal Basis Profile ermöglicht, zusätzlich zu den bereits im FP enthaltenen Möglichkeiten, die graphische Benutzeroberfläche und das Xlet Application Model mit der Inter-Xlet Communication (IXC) (Kommunikation zwischen den Xlets). Xlets ist ein anderer Name für die PBP Anwendungen. Sie sind den Applets sehr ähnlich. Wie Applets werden Xlets von der Software kontrolliert, die sie aufgerufen hat. Xlets haben keine *main*-Methode und implementieren immer das Interface Xlet. Sie besitzen einen Lebenszyklus, der durch verschiedene Methoden von einem Zustand in den nächsten gebracht werden kann. Mittels Inter-Xlet Communication können Xlets sogar untereinander kommunizieren.

Die Kombination CDC/FP/PBP macht interaktive Anwendungen möglich, die einen durch Xlets wohldefinierten Anwendungslebenszyklus haben. Sie wird hauptsächlich bei Geräten gewählt, die interaktives Fernsehen ermöglichen.

Das Personal Profile in Kombination mit CDC soll die Spezifikation Personal Java ersetzen. Personal Java war Suns erster Schritt Anwendungen für netzwerkfähige, mobile Geräte in Java zu ermöglichen. Anwendungen die für Personal Java geschrieben wurden können nun auf einfache Weise in Anwendungen für CDC/FP/PP umgeschrieben werden.

Des Weiteren erweitert es das PBP in einigen Punkten. So unterstützt PP das Applet Application Model und enthält eine umfangreichere Teilmenge des AWT. Somit ist die Kombination CDC/PP dem J2SE noch ähnlicher und enthält darüber hinaus das Xlet Application Model sowie die Inter-Xlet Communication.

CLDC

Die CLDC ist für Geräte, die mit sehr beschränkten Ressourcen auskommen müssen, gedacht. Dabei handelt es sich hauptsächlich um mobile Geräte wie Handys, PDAs, Pagers, etc. Die CLDC setzt außer der minimalen Speicheranforderung keine besonderen Hardware-Anforderungen voraus. Die Speicheranforderung besagt, dass die VM, die Konfigurationen, die Profile und die Applikationen innerhalb von 160-512kB Platz haben müssen. Daher benutzt CLDC eine abgespeckte Version der VM.

Virtual Machine: KVM

Aufgrund des eingeschränkten Speicherplatzes müssen Abstriche bei den verwendbaren Datentypen und den eingebundenen Klassen gemacht werden. Datentypen *float* und *double* werden nicht unterstützt, was nicht schlimm ist, denn in den meisten Geräten auch keine Gleitkomma-Recheneinheit (FPU) eingebaut ist. Weil die CDC nur die wichtigsten Klassen aus den Packages *java.lang*, *java.io* und *java.util* der J2SE beinhaltet, ist die Behandlung von Exceptions eingeschränkt. Bei nicht laufzeitbezogenen Fehlern wird entweder die Anwendung beendet oder das Gerät neu gestartet. Dann ist der *bytecode verifier* nicht mehr in der Virtual Machine integriert, weil der Prüfprozess zeit- und ressourcenaufwändig ist. Der Hauptteil der Überprüfung findet auf dem Rechner, auf welchem das Programm kompiliert wird, statt (Preverification). Die KVM prüft dann lediglich mit ihrem eigenen Verifier, ob die Classdatei vorüberprüft wurde und immer noch gültig ist.

Profil von CLDC:

MIDP (Mobile Information Device Profile)

Die CLDC wird von MIDP, dem *Mobile Information Device Profile*, erweitert. MIDP ist speziell auf Handys ausgerichtet. Das Profil definiert ein neues Application Model und neue Klassen für die Gestaltung von Benutzeroberflächen und der Speicherung von Daten in Form von Records.

Die Applikationen für MIDP hat Sun in Anlehnung an die Applets im Webbrowser *Midlets* getauft.

Nachfolgend befindet sich eine Tabelle der Pakete, die das MIDP enthält, mit der kurzen Beschreibung, um einen schnellen Überblick über die Möglichkeiten der Kombination CLDC/MIDP zu geben.

Name	Beschreibung
<i>java.util</i> <i>java.lang</i>	eine Untermenge der entsprechend gleichnamigen Pakete des J2SE mit der bekannten Einschränkungen
<i>javax.microedition.rms</i>	Speicherung der Daten in Form von Records und deren Bearbeitung
<i>javax.microedition.midlet</i>	Beschreibung der Midlets und der Interaktionen
<i>javax.microedition.io</i>	Verbindungen über das HTTP und SLC
<i>javax.microedition.lcdui</i> , <i>javax.microedition.lcdui.game</i>	GUI und Entwicklung von Spielen, RGB Bilder
<i>javax.microedition.media</i> , <i>javax.microedition.media.control</i>	Abspielen von Tönen, Sequenzen und WAV Dateien
<i>javax.microedition.pki</i>	Sicherheit

Abb. 4 Pakete des MIDP und ihre Verwendung

3 Optionale Pakete

Wir haben bereits gesehen, dass mit Hilfe von Konfigurationen und Profilen ziemlich genau auf die Möglichkeiten eines Geräts eingegangen werden kann. Manche Geräte nutzen aber sehr spezielle Technologien. Daher können sich Geräte, welche dieselbe Kombination Konfiguration/ Profile verwenden trotzdem noch in diesen Funktionen unterscheiden. Zum Beispiel gibt es Handys, die Bluetooth unterstützen, andere tun dies nicht.

Optional Packages bilden eine Schnittstelle, um auf solche Funktionen eines Gerätes zuzugreifen. Sie sind nicht in den Profilen enthalten, um einen unnötigen Speicherverbrauch zu vermeiden.

J2ME Web Services (JSR* -172)

Die wesentlichen Aufgaben der Spezifikation sind die Beschreibung von Schnittstellen und der Nachrichtenaustausch. Hierzu wurden SOAP, WSDL und UDDI als drei Kerntechnologien entwickelt. Sie basieren alle auf XML und nutzen die Internetprotokolle HTTP und SMTP.

SOAP (Simple Objekt Access Protocol) ist das grundlegende Transportmittel für Web Services. Die SOAP-Nachrichten werden über ein Trägerprotokoll, wie z.B. HTTP, transportiert. Eine SOAP-Nachricht besteht aus einem Umschlag (ENVELOPE), der sich in Kopf (HEADER), der Anwendungsdaten wie XML-Namespaces enthält, und den Rumpf (BODY) für die eigentliche Nachricht unterteilt.

Web Services Description Language (WSDL) wurde in XML Format zur Beschreibung der SOAP Nachrichten eingeführt. Zu beachten ist hierbei, dass alles was in WSDL beschrieben ist abstrakt ist, d.h. es werden keinerlei Funktionen oder Dienste bereitgestellt, nur beschrieben.

UDDI ist eine Registrierungsdatenbank, die programmatische Beschreibung von Internet-Diensten enthält, weiterhin programmatische Beschreibungen von Unternehmen und den Diensten, die sie unterstützen. Diese Registrierungsdatenbank ist im Internet öffentlich zugänglich sein.

Zweck des UDDI ist die Zusammenarbeit von Firmen über Ihre Internet-Dienste zu beschleunigen, und das Angebot von Internet-Diensten zu fördern. Dies sollte durch eine Standardisierung von Beschreibung, Auffindung und Integration von Geschäften/ Unternehmen für das Internet erreicht werden.

*JSR ist die Abkürzung für Java Specification Request

Sun hat zur Erweiterung der Java 2 Micro Edition Plattform die J2ME Web Services API (JSR-172) herausgebracht um Web Services auch bei mobilen Applikationen zu unterstützen. Die API besteht aus zwei zusätzlichen Packages, und zwar sind es die JAX-RPC und JAXP.

JAX-RPC steht für Java API for XML-based Remote Procedure Call und ist für die Erzeugung von Web Services und Clients zuständig, die RPCs nutzen. Ein RPC Mechanismus ermöglicht den Aufruf von Methoden auf anderen Systemen und wird in JAX-RPC als XML-basiertes Protokoll, wie z.B. SOAP, repräsentiert.

JAXP steht für Java API for XML Processing, ein XML Parsing System auf mobilen Geräten.

J2EE Client Provisioning

Ein Provisioning Server ist eine Web Seite, die den Clients verschiedene Sachen zum Herunterladen anbietet, z.B. Fotos, Programme oder Lieder.

J2EE Client Provisioning Server stellen einen zentralen Behälter (Repository) für verschiedene Inhalte, von Client Applikationen bis hin zu Multimedia Inhalten, bereit.

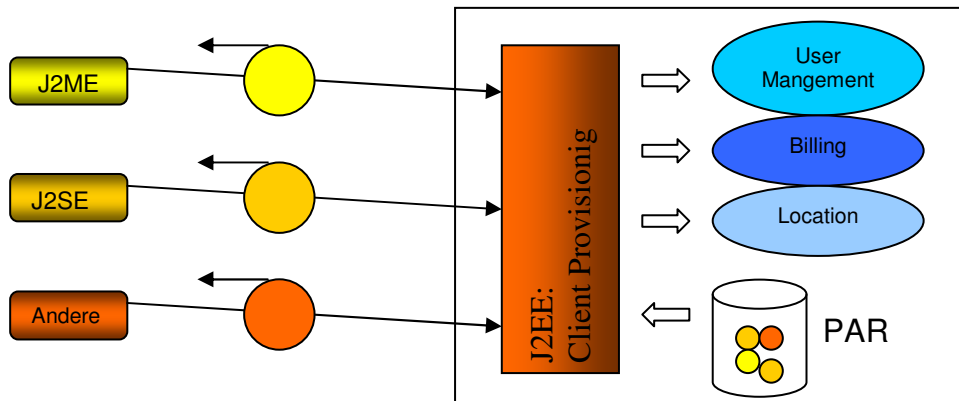


Abb. 5 Funktionalität des Client Provisioning Servers

Die Grafik zeigt die Interaktion zwischen Clients und dem Provisioning Server. Die Clients suchen aus dem Repository etwas aus und erhalten dann ihre Wahl. Es lassen sich auch Inhalte hoch laden und der Server ist mit Systemen für Zahlungen (Billing) und einer Benutzer Verwaltung (User Management) ausgestattet.

Die zentralen Elemente eines J2EE Client Provisioning Servers sind zum einen die Konfiguration Dateien *devices.xml*, *adapters.xml* und *matchers.xml* und zum anderen das Provisioning Archiv (PAR). *device.xml* definiert die unterstützten Geräte, *adapters.xml* ist für die unterstützten Plattformen zuständig und *matchers.xml* enthält Regeln zur Überprüfung von gewissen Voraussetzungen oder ob Geräteeinschränkungen eingehalten wurden, bereit. PAR ist ein Archiv, das alles lagert, was vom Client heruntergeladen wird und was der Client als Ganzes hoch laden kann.

Die J2EE Client Provisioning Architektur benötigt für die Kommunikation mit dem Server einen Client. In Java sollte dieser die Protokolle MIDP OTA für das Herunterladen von MIDP Applikationen für mobile Geräte oder JNLP für das Herunterladen von J2SE Applikationen unterstützen. Dem Client stehen dann drei Hauptfunktionalitäten zur Verfügung: Discovery, Delivery und Stocking.

Discovery ist eine Funktion des Provisioning Servers, bei der der Client den Server nach Inhalten absucht, die er Herunterladen kann.

Nach der Erforschung des Angebotes steht nun die Versendung des gewünschten Inhaltes an (Delivery).

Wenn der Entwickler fertig mit der Erstellung des Provisioning Portals, der Lagerstätte für die Repository's, ist und der J2EE AppServer mit den Provisioning Server hochgefahren wurde, müssen nun die PAR Archive hoch- und runtergeladen werden, damit andere Clients den Dienst auch nutzen können. Dies nennt man Stocking.

In dem Client Provisioning zeigt sich der Wandel der IT- Orientierung in die wirtschaftliche Richtung.

Mobile Media API (JSR-135)

Die Audio- /Videowiedergabe und ihre Übertragung bei mobilen Geräten sind durch ihre geringe Leistungsfähigkeit eingeschränkt. Die Mobile Media API ist ein optionales Package für J2ME, das die Probleme möglichst geschickt lösen soll. Deswegen wurde ein Konzept entworfen, das reibungslos

und relativ unabhängig von Übertragungsart und Datenquelle arbeitet. Dieses Konzept der Mobile Media API baut sich auf den Komponenten: DataSource, Player, Controls und Manager.

Die Klasse DataSource wird benutzt, um die Daten von ihrem Ursprungsort hin zum Player zu transportieren. Die Übertragungsart ist frei wählbar (HTTP, RTP usw.). Das Interface Player erledigt das Abspielen von sämtlichen Arten zeitbasierter Multimedia. Es besitzt die Methoden, um die Wiedergabe zu starten, stoppen oder an eine beliebige Stelle des Datenflusses zu springen und Controllable als SuperInterface. Dieses wird benutzt, um einem Player die jeweils anwendungsspezifischen Control Interfaces anzuhängen (RecordControl, VideoControl, TempoControl, VolumeControl usw.). Es ist auch möglich eigene Controls zu implementieren. DataSource und Player werden über Manager miteinander verknüpft.

Die API wurde zu einem sehr großen Teil von dem Java Media Framework abgeleitet und auf eine Größe von 20KB reduziert, was eingeschränkte Funktionalität zur Folge hat.

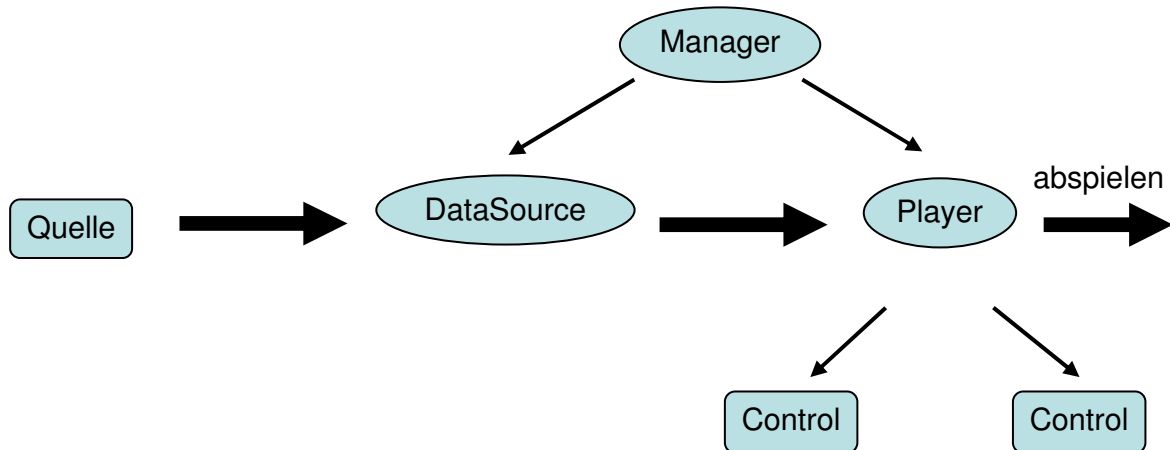


Abb. 6 Zusammenspiel der Komponenten der Mobile Media API

Mobile 3D Graphics (JSR-184)

Um den zahlreichen Anforderungen gerecht zu werden unterstützt die Mobile 3D Graphics API je nach Bedarf Scene Graph API oder Immediate Mode API oder beide zur gleichen Zeit.

Ein Java3D Programm erschafft Java3D Objekte und verstaut diese in einen "Scene Graph", das ist eine baumartige Datenstruktur, die unter anderem aus verschiedenen 3D Objekten besteht und sowohl den Inhalt des "Virtual Universe" (virtueller Raum, in dem die Objekte platziert werden) als auch dessen Rendering beschreibt. Im Fall der J2ME hat die Klasse „World“ die Wurzelfunktion. Die World kann keine Knoten über sich haben und das bedeutet, dass es kein „Universe“ wie in vielen anderen 3D APIs gibt.

Die 3D Objekte werden auch hier von der Klasse Graphics3D gezeichnet und zwar mit der Methode *render()*, die in vier Varianten existiert, um allen Ansprüchen gerecht zu werden.

Trotz der Wichtigkeit des geringen Speicherverbrauchs und trotz geringer Prozessleistung, kann es auch im mobilen Bereich Fälle geben, in denen höherer Speicherverbrauch und langsamere Darstellung in Kauf genommen werden, um eine höhere Qualität zu erzielen. Die Klasse Graphics3D bietet drei verschiedene Möglichkeiten dazu an, und zwar: Antialiasing, Dithering und True Color Rendering. Es handelt sich dabei um statische Variablen der Klasse, die an- bzw. ausgeschaltet werden können.

Die wichtigsten Konkurrenten der API sind SVG Mobile und OpenGL ES. Aber die Mobile Graphics3D API gewinnt mit der Vielfalt an Möglichkeiten, „high-level scene graph“, Kompatibilität zu OpenGL.

Java TV API

Set-Top Boxen und digitales Fernsehen, die das Java TV API unterstützen, bieten dem Verbraucher interaktive TV-Inhalte, zum Beispiel Enhanced Television, Video-On-Demand (VOD), Electronic Program Guides (EPGs) und interaktive Sportübertragungen mit mehreren Kameraeinstellungen. Die API verwaltet darüber hinaus Funktionen wie Audio/Video Streaming, Conditional Access und Zugang zu In-Band und Out-of-Band Daten-Kanälen.

Zusätzlich wird eine unabhängige, zwischen verschiedenen Betriebssystemen und Mikroprozessoren portierbare Software-Plattform für den Zugriff auf spezielle Hardware-Eigenschaften von TV-Geräten angeboten (z.B. Tuner-Kontrolle für den Programmwechsel und Grafik auf dem Bildschirm).

Der größte Vorteil, den die Java TV API dem Fernsehmarkt bringt, ist die Fähigkeit eine bedeutend bessere Übertragung zu ermöglichen. Beim digitalen Fernsehen kommt das Bild in der gleichen Qualität auf dem Bildschirm an, in der es vom Sender abgeschickt worden ist oder, falls die Störungen nicht mehr kompensiert werden können, dann überhaupt keins.

Bluetooth API (JSR-82)

Die Bluetooth API erlaubt es, eine im Endgerät vorhandene Bluetooth-Infrastruktur über Java anzusprechen. Im Folgenden sind einige Kernkonstrukte zur Kommunikation via Bluetooth erläutert. Sie baut sich auf Protokollen und Profilen.

Schichten	Protokolle	Beschreibung
Bluetooth Kernprotokolle	Baseband	- physikalische Verbindung - Übertragung
	SDP	- Kommunikation zw. mehreren Geräten - Abfrage der geräte- und verbindungsspezifischen Informationen
	L2CAP	
	LMP	- Verbindungsaufbau - Sicherheit - Einteilung der Paketgrößen
Cable Replacement Protocol	RFCOMM	
Telephony Control Protocols	TCS-Binary	- Übertragung von Sprache
	AT-commands	- Datenübertragung
Aufgesetzte Protokolle	PPP, OBEX, WAP, vCard, vCal, IrMC, WAE	
	UDP/TCP/IP	- Einbindung von LANs oder WLANs
Profile:		
Generic Access Profile(GAP)		- Grundlage für alle anderen Profile - Gegenseitiges Erkennen zw. den Geräten
Serial Port Profile (SPP)		- Ersetzen von seriellen Kabelverbindungen
Service Discovery und Application Profile (SDAP)		- Definition von eigenen Protokollen
Generic Object Exchange Profile (GOEP)		- Standard für den Austausch von Objekten

Abb. 7 Protokolle und Profile der Bluetooth Technologie

Für die Zusammenarbeit mit diesen Protokollen und Profilen sind in der API Klassen und Packages definiert. Aufgrund der Architektur der API ist die Weiterentwicklung jederzeit möglich.

Wireless Messaging API (JSR-135) und JavaPhoneAPI

Die APIs unterstützt das Versenden und Empfangen von Mitteilungen wie Kurznachrichten (SMS) und ab Version 2.0 multimediale Mitteilungen (MMS) und telefonieren mit mobilen Geräten.

4 Sicherheitsmechanismen in J2ME (CLDC/MIDP/JSR-177)

Bei den vielen Möglichkeiten, die J2ME anbietet, wird die Sicherheitsfrage aktuell. Die Plattform bedarf eines eigenen Sicherheitsmodells, weil für sie wegen der Speicherbeschränkungen das übliche Modell nicht geeignet ist.

Das CLDC Sicherheitsmodell kennt drei grundsätzlich unterschiedliche Sicherheitslevels. Auf dem ersten *Low level security* wird die semantische Korrektheit der Programme überprüft. Auf dem *Application level security* wird sichergestellt, dass das Programm klar definierte Zugriffsrechte auf die Bibliotheken und Systemressourcen hat. *End-to-end security* garantiert schließlich, dass jede Datenübertragung, die vom mobilen Gerät ausgelöst wird, auf dem kompletten Übertragungspfad vom mobilen Gerät bis zum Übertragungspartner (zum Beispiel einen Internetserver) und zurück sicher ist.

Des Weiteren existiert noch eine optionale *Security and Trust Services API (SATSA)*. Ihre Aufgabe, wie ihr Name schon sagt, ist die Sicherheit und Vertraulichkeit zu garantieren. Mit der SATSA wird ein Sicherheitselement, das so genannte *Security element (SE)* eingeführt. Ein SE ist eine Komponente in einem Java 2 ME Gerät (z. B. *SIM Card*), das sichere Speicherung der vertraulichen Daten, verschlüsselte Übertragung und eine sichere Ausführungsumgebung verspricht.

Basierend auf verschiedenen Software- und Hardwarecharakteristiken der verschiedenen Sicherheitselementen werden in der SATSA folgende Funktionen für Java 2 ME Plattformen angeboten: *Smart Card* Kommunikation (SATSA-APDU), Services für digitale Signaturen und das Userzertifikatesmanagement (SATSA-JCRMI, SATSA-PKI) und eine Bibliothek zur Verschlüsselung von Daten (SATSA-CRYPTO).

Damit werden unter anderem die Identifikation des Users, Banking und Payment ermöglicht.

Forscher und Hacker warnen seit langem vor Viren und Würmern für Handys. Insbesondere seitdem sehr viele Geräte Java-Applikationen ausführen können und Smartphones an die Leistungsfähigkeit von PDAs herankommen, ist es nur eine Frage der Zeit, bis die Schädlingsschwelle auf den Mobilfunkbereich überschwappt.

5 Entwicklungsumgebungen und Programmierpraxis

Zum bequemen Programmieren werden mehrere Entwicklungsumgebungen angeboten. Es gibt freie Versionen wie z.B. Borland JBuilder 8 plus Mobile Set oder Sun ONE Studio 4 update 1, Mobile Edition [SunONE], Eclipse. Zum anderen gibt es kostenpflichtige kommerzielle IDEs (Integrated Development Environments), wie z.B. IBMs WebSphere Studio Device Developer [WSDD]. WSDD ist eine Erweiterung von Eclipse und unterstützt auch die Entwicklung von MIDlets.

Im Folgenden ein Paar Bemerkungen zu der Programmierpraxis. Die J2ME ist nur mit den nötigsten und unerlässlichen Befehlen ausgestattet, auf vieles wurde aus Platzgründen verzichtet. Dies fällt schnell auf, z.B. gibt es keine vorgefertigte `Math.random()` Methode, stattdessen muss eine eigene Methode mit Hilfe des Random-Objektes implementiert werden. Und auch sonst stößt der Programmierer schnell an die Grenzen der J2ME. Doch die Eingeschränktheit bietet durchaus auch ihre Vorteile, ist das MIDP äußerst übersichtlich, was den Einstieg erleichtert.

6 Ausblick

Bisher hat beinahe jeder Hersteller von Geräten mit beschränkten Ressourcen eine eigene Entwicklungsplattform für die Applikationen benutzt. Die Einführung offener, Java basierender Plattformen wie JavaCard und J2ME für mobile Geräte wird die Welt der mobilen Anwendungen grundlegend verändern. Applikationen können über die Luftschnittstelle (OTA) entweder auf die SIM Karte oder das Handy geladen werden. Java ebnet den Weg für jede Form von neuen, interaktiven, grafischen und sicheren Anwendungen.

Die Kombination von SIM Karten mit J2ME Handys erlaubt verteilte Applikationsmodelle wie z.B. Flugplan-Anwendungen, bei denen elektronische Tickets, Kreditkarteninformationen und Loyalty Punkte auf der SIM Karte gespeichert werden und die Benutzeroberfläche auf dem Handy liegt.

Um die besondere Bedeutung von J2ME zu zeigen sind einige praktische Anwendungsszenarien zu nennen, wie z. B. die Routenplanung, bei der die Planungssoftware Wetter- und Stauinformationen als Web Service nachfragen und in die Planung einbeziehen kann. Oder ein Arzt kann auf bei einem Noteinsatz Patientendaten zugreifen. Gesamtstatus der Anlage, letzte Kalibrierdaten etc. können vom Service-Techniker während der mobilen Anlagewartung herangezogen werden. Vertreter auf

Kundenbesuch haben aktuelle Rabattsätze, Außenstände parat. Um die heutzutage besonders wichtige Unterhaltungsbranche nicht auszulassen, wäre hier die Interaktion mit mehreren anderen Benutzern beim Spielen zu erwähnen.

Unter der URL <http://wireless.java.sun.com/device>, lässt sich nachlesen welche Geräte J2ME unterstützen. An der Menge der Einträge lässt sich die Wichtigkeit und schnelle technologische Entwicklung seit der ersten Spezifikation 1999 erkennen.

Literaturverzeichnis:

Die Ausarbeitung stützt sich hauptsächlich auf die Vorträge der übrigen Teilnehmer des Proseminars.

Außerdem:

Die offizielle J2ME Seite:

<http://java.sun.com/j2me/>

Eine Arbeit über J2ME an der Uni Erlangen:

<http://www2.informatik.uni-erlangen.de:8083/Lehre/SA-DA/Themen/SA-abgeschlossen/tonka.pdf>

Ein Vortrag über J2ME:

http://informatik.hsr.ch/Content/Gruppen/Doz/jjoller/ISeminare/SeminarWS2001_2002/J2ME/

Abbildungsverzeichnis:

Abb. 1 Die Architektur von J2ME	3
Abb. 2 Der Vergleich zwischen CLDC, CDC und J2SE	4
Abb. 3 Aufteilung der Geräte auf die Konfigurationen.....	4
Abb. 4 Pakete des MIDP und ihre Verwendung.....	6
Abb. 5 Funktionalität des Client Provisioning Servers	7
Abb. 6 Zusammenspiel der Komponenten der Mobile Media API	8
Abb. 7 Protokolle und Profile der Bluetooth Technologie.....	9