



Zusammenfassung und Programmdokumentation

der

Diplomarbeit im Fach Algorithmen und Datenstrukturen

„Ein Geodaten-Viewer mit automatischer Textplatzierung“

von

Hanspeter Buchegger

Betreuer: Prof. Stefan F. Keller, Hochschule für Technik Rapperswil

vorgelegt im August 2002 im Fachbereich Elektrotechnik



Fachhochschule Zürich

Inhaltsüberblick

| | | |
|---|------------------------------------|---|
| 1 | Einführung | 2 |
| 2 | Yagger - Ein Geodaten-Viewer | 2 |
| 3 | Fazit und Ausblick | 4 |
| 4 | Benutzerhandbuch (Tutorial) | 5 |
| 5 | Programmdokumentation | 8 |

1 Einführung

1.1 Zielsetzung

Ziel dieser Diplomarbeit war es, eine Plattform für die Erprobung von Plazierungsalgorithmen in digitalen Karten zu erproben und gleichzeitig vorhandene Verfahren auf punkt-, linien- oder flächenförmige Objekte anzuwenden.

Die Komplexität der Beschriftung von Landkarten wächst exponentiell mit der Anzahl zu beschriftender Objekte, es werden deshalb heuristische Ansätze verfolgt, dieses NP-vollständige Problem zu lösen.

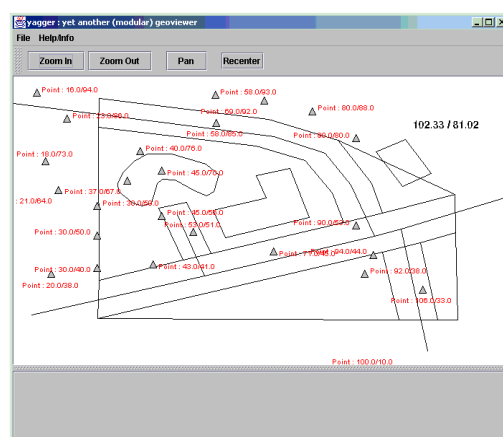
Es wurden aufgrund der Aufgabenstellung folgende Ziele festgelegt:

- Entwicklung eines Geodaten-Viewers für INTERLIS 2/XML-Daten auf der Basis eines Open Source-Bibliothek (dient zugleich auch als Technologiestudie)
- Entwicklung einer Schnittstelle für zur Laufzeit geladene ausführbare Programmodule (im folgenden „Plug-ins“ genannt)
- Entwicklung eines Plug-ins für die Beschriftung von Punktförmigen Geo-Objekten unter Verwendung eines Simulated Annealing-Algorithmus
- Entwicklung eines Plug-ins für die Beschriftung von linienförmigen Objekten unter der Verwendung von Simulated Annealing- oder Gradient Descent-Algorithmen.
- Erprobung des Laufzeitverhaltens des Linienbeschriftungsalgorithmus.

1.2 Erreichte Ziele

Die erreichten Ziele sind:

- Entwicklung eines Geodaten-Viewers für INTERLIS 2/XML-Daten als Erweiterung der Open Source-Bibliothek JaGo inkl. kleiner Erfahrungsbericht.
- Laden von ausführbaren Programmteilen (Plug-ins) zur Laufzeit.
- Prototyp eines INTERLIS 2/XML- nach GML-Interpreters/Umsetzers als ladbares Modul (Plug-in).
- Programmierung eines Punktbeschriftungsalgorithmus-Prototyps, der das Simulated Annealing-Verfahren als ladbares Modul (Plug-in) realisiert.
- Tests zum Laufzeitverhalten des Punktbeschriftungsalgorithmus' durchgeführt werden.
- Versuche mit verschiedenen Qualitäts- und Mutationsfunktionen.



Figur: Automatische Beschriftung von 25 Punkten in einer Testkarte.

2 *Yagger* - Ein Geodaten-Viewer mit JaGo

2.1 Funktionsumfang

Das Programm *yagger* (für „yet another geodata viewer“) kann ausführbare Module zur Laufzeit laden und ausführen. Diese Technik ist allgemein als ‚Plug-in‘ von verschiedenen, meist kommerziellen Softwareprodukten bekannt. Die Programmschnittstellen sind offen ausgelegt, es können weitere Plug-ins, die andere Funktionen abdecken, eingebunden werden.

Eine weitere Funktion des Programms ist, dass es Daten, die im INTERLIS 2/XML-Format vorliegen, lesen kann. Ein Plug-in liest Geodaten, die von einem INTERLIS 2-fähigen Geo-Informationssystem erzeugt werden und generiert GML-konforme, darstellbare Objekte. INTERLIS ist eine Schweizer de-jure Norm als Verbindung von UML zu XML (www.interlis.ch). GML (Geographic Markup Language) ist ein XML-Format, das vom amerikanischen OpenGIS Consortium als Industrie-Norm vorgeschlagen worden ist (www.opengis.org). Die INTERLIS 2/XML-Geodaten werden durch einen aus der JaGo-Bibliothek entnommenen Rendering-Service in einer grafischen Ausgabe dargestellt.

Um aufzuzeigen, wie ein Plazierungs-Plug-in aufgebaut sein kann, wurde ein Beschriftungsalgorithmus für Punktobjekte implementiert. Die notwendige programmtechnische Infrastruktur, um Linien- oder Flächenbeschriftungsalgorithmen zu realisieren ist implementiert worden.

Das Programm ist kein kommerzielles Produkt, vielmehr soll es als Experimentier-Plattform dienen, mit der Interessierte an das Thema ‚Geo-Informationssysteme‘ und das daraus resultierende Umfeld herangeführt werden. Der Sourcecode des Programms ist frei verfügbar und untersteht der sogenannten ‚BSD-OpenSource‘-Lizenz. Es wird vom Autor ausdrücklich gewünscht, dass die Ergebnisse einem breiteren, in GIS interessierten Publikum zur Verfügung gestellt werden.

2.2 Erreichte Qualität der Beschriftung

Der Beschriftungsalgorithmus berücksichtigt für die Bestimmung der Qualität die folgenden Kriterien:

- Überlappung der Beschriftung mit anderen Punktobjekten
- Gegenseitige Überlappung von Beschriftungen
- Position der Beschriftung in der Darstellung (Kartenränder)
- Positionsprioritäten der Beschriftung nach [Imhof 1962]
- Überdeckung von anderen Kartengrafikelementen

Neu gegenüber den Vorarbeiten [Barbeitos 2001] ist hier, dass der Kartenhintergrund für die Qualitätsberechnung herangezogen wird. Die dabei verwendeten Kriterien sind allerdings noch sehr einfach gehalten. Es bedarf weiterer Experimente und Überlegungen, um hier bessere Ergebnisse zu erreichen.

2.3 Laufzeiten

Die Berechnung der Qualität hat einen wesentlichen Einfluss auf die Laufzeiten. Insbesondere der Algorithmus, der die Überlappung von fremder Karteninformation durch Beschriftungstext berechnet, hat eine relativ grosse Laufzeit, da die Bewertung auf der Betrachtung einzelner Pixel der Darstellung beruht. Es müssen im Schnitt ca. 1000 bis 1200 Pixel einzeln verarbeitet werden. Hier könnten Verfahren aus der rechnergestützten Bildverarbeitung eine weitere Verbesserung bringen.

3 Fazit und Ausblick

3.1 Nicht erreichte Ziele

Die nicht erreichten Ziele sind:

- Ein umfassender, flexibler INTERLIS 2-Interpreter konnte nicht realisiert werden.
- Die Funktionalität (Plug-in) für Linienbeschriftung konnte nicht mehr realisiert werden.

3.1.1 Begründung für das Nichterreichen

Im Zusammenhang mit der verwendeten GIS-Bibliothek (JaGo, heute deegree, www.deegree.org) traten erhebliche Schwierigkeiten wegen fehlender Dokumentation auf (Beziehungen zwischen Klassen, Methoden und deren Semantik). Es war leider notwendig, komplexe Teile der vorhandenen Quellen zu analysieren, um auf die Funktionsweise zu schliessen. Dies führte insbesondere bei der Implementation des INTERLIS 2/XML-Ladeprogramms zu grossen Verzögerungen.

3.1.2 Vorgesehene Verbesserungen

Die Komplexität der verwendeten GIS-Bibliothek (JaGo) wurde erheblich unterschätzt. In einem Projekt dieses Umfangs sind fremde, nicht selbst entwickelte Software-Bibliotheken angesichts der zur Verfügung stehenden Zeit allerdings unverzichtbar. Es wurde von Seiten des Autors aber der lückenhaften Dokumentation zu wenig Aufmerksamkeit geschenkt. Die Hoffnung, dass sich die Bibliothek wegen offener Quellen trotz schlechter Dokumentation gut verwenden lässt, erwies sich in diesem Falle als Fehlannahme.

3.2 Ausblick: Erweiterungen

3.2.1 Verbesserung des INTERLIS 2/XML-Imports

Die aktuell implementierte INTERLIS 2/XML-Transferformat-Interpretation ist noch nicht vollständig. Hier ist noch ein weites Betätigungsfeld für zukünftige Arbeiten zu finden. Derzeit werden nur die INTERLIS-Primitive COORD2 (= Punkt-Geometrietyp) und Polyline (= Linien- Geometrietyp) korrekt in OpenGIS-Objekte umgesetzt.

3.2.2 Qualitätsfunktionen

Die Betrachtung des Kartenhintergrundes ist ein weitaus komplexeres Problem als zunächst angenommen. Es gilt, eine Funktion zu finden, mit der die überdeckten Kartengrafikelemente gut beschrieben werden. Dazu wurden eigene Überlegungen angestellt.

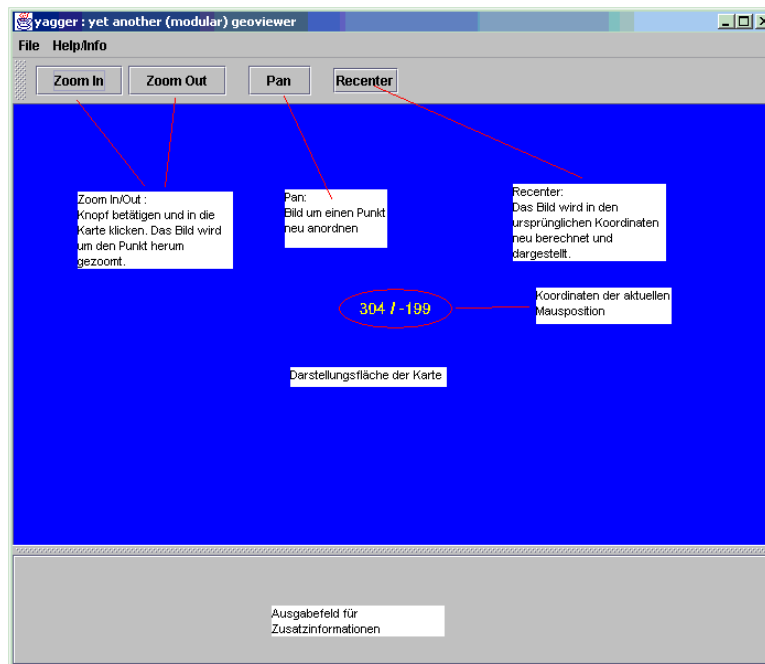
3.2.3 Platzierungsalgorithmen für Linien- und Flächenobjekte

Das vorliegende Programm soll als Basis für neue Platzierungsalgorithmen dienen. Im Zusammenhang mit Punktobjektbeschriftungen wurden neue Ansätze zur Beurteilung der Qualität verfolgt. Es wurden auch Experimente mit hybriden Platzierungsalgorithmen gemacht.

4 Benutzerhandbuch (Tutorial)

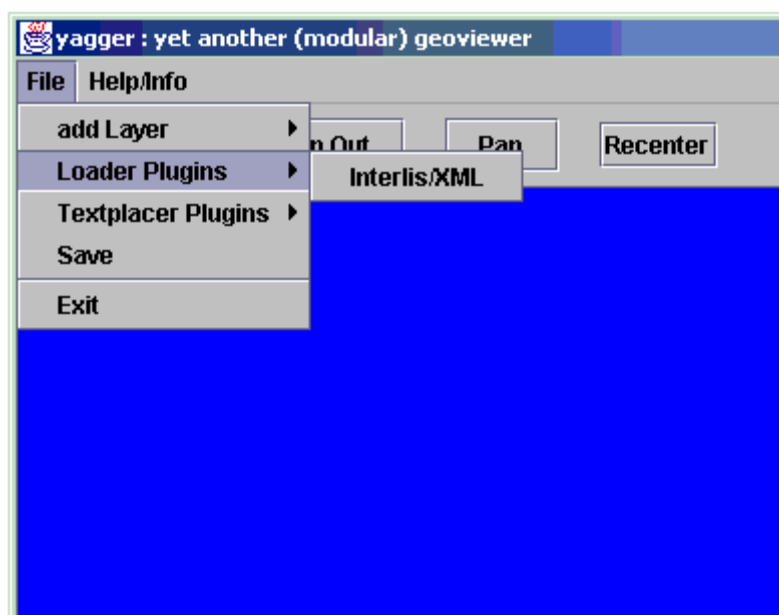
Diese Anleitung beschreibt nur die zum Laden von INTERLISdaten und dem anschliessenden Aufruf des Plazierungs- Plug-ins notwendigen Schritte.

Bedienungselemente



Figur 1. Bedienungselemente.

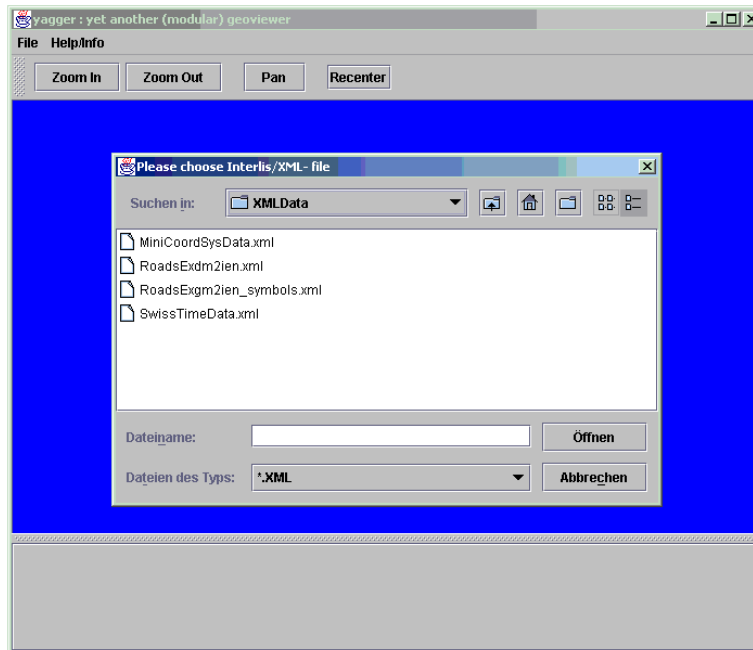
Wählen eines Loader-Plug-ins



Figur 2. Wählen eines Loader-Plug-ins.

Über das Menu ‚File‘ kann ein Loader-Plug-in angewählt werden. Derzeit wird nur nach einem Plug-in mit einem bestimmten Namen gesucht. Um hier eine Liste zu bekommen, muss dieser Algorithmus noch verbessert werden. Die Plug-ins, die hier angezeigt werden, geben bestimmen den in diesem Menu angezeigten Text selbst. Er kann frei gewählt werden, einzige Einschränkung ist derzeit, dass Plug-ins gleicher Funktionalität (also Loader oder Placer) eindeutige Namen haben müssen.

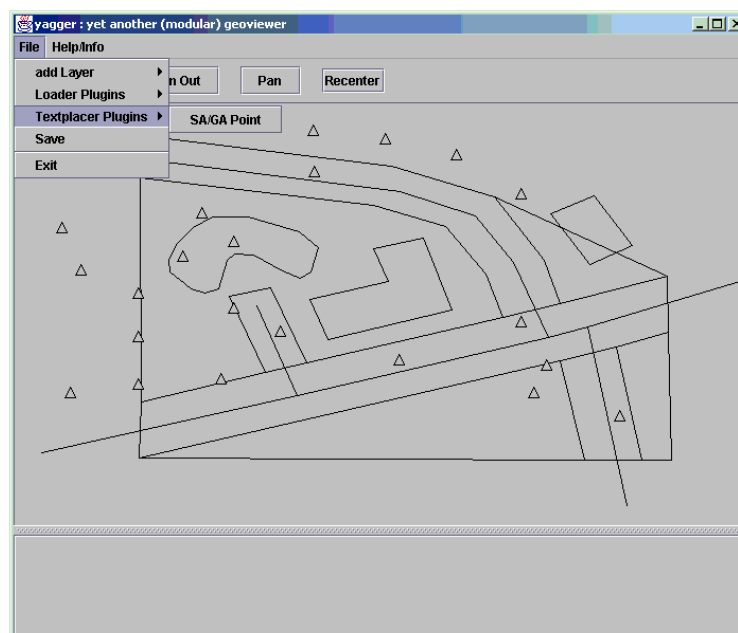
Wahl von INTERLIS-Daten



Figur 3. Wahl von INTERLIS-Daten.

Wenn der Loader keine Angabe hat, welche Datei er laden soll, öffnet er einen FileChooser-Dialog. Hier kann durch einfaches Anklicken und drücken der Taste ‚Öffnen‘ das File geladen werden. Nach dem Laden wird es automtatisch dargestellt.

Wählen eines „Placers“



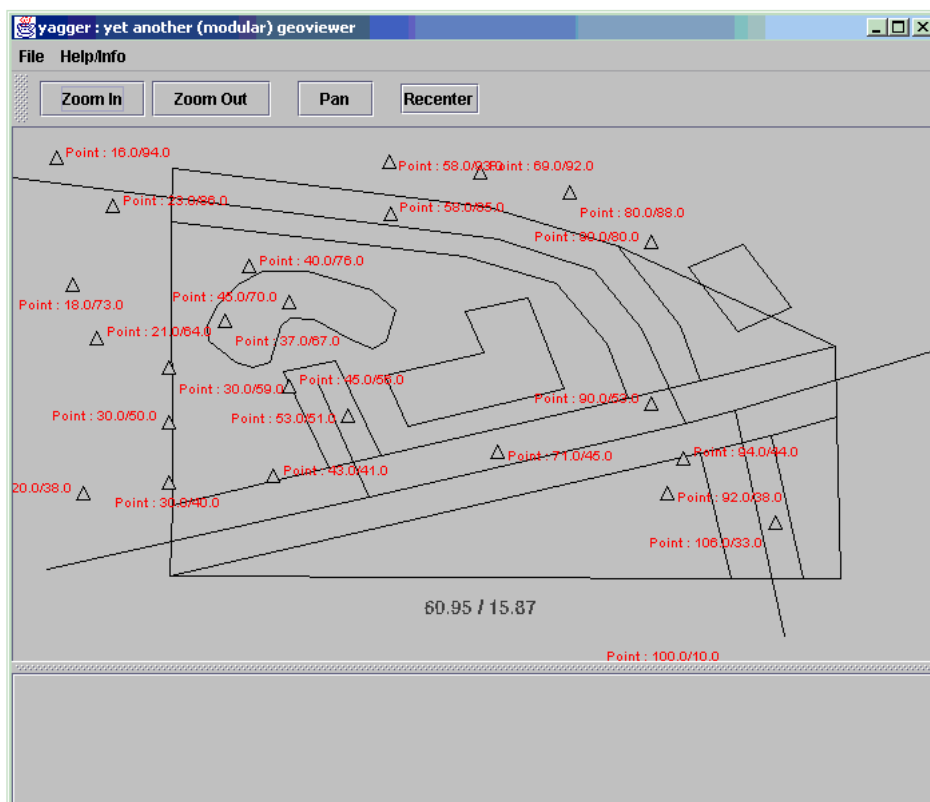
Figur 4. Wählen eines „Placers“.

Wenn die Daten geladen und dargestellt sind, kann ein Placer- Plug-in aufgerufen werden. Dies ist auch dann notwendig, wenn Zooming- und Panning- Funktionen verwendet werden, da der Placer nicht automatisch aufgerufen wird.

Ausgabe des Resultates

Die Ausgabe des Resultats erfolgt automatisch. In der aktuellen Version werden die Punktnamen noch nicht aus der INTERLIS/XML- Quelle gelesen sondern syntetisch erzeugt. Der ausgegebene Text wird aus den Koordinaten des Punktes erzeugt (Beispiel Point : 16/94 hat die Koordinaten 16 / 94)

In der derzeitigen Version existiert noch ein Fehler. Nach dem Plazieren kann es vorkommen, dass das Bild nicht korrekt aufgebaut und damit die plazierten Texte nicht alle sichtbar sind.



Figur 5. Ausgabe des Resultates.

5 Programmdokumentation

5.1 Lieferumfang

‚yagger‘ wird als Standalone- Applikation mit allen Bibliotheken geliefert, die zum Betrieb notwendig sind. Der Quellcode wird ebenfalls mitgeliefert, es steht dem Benutzer frei, eigene Änderungen oder Anpassungen daran vorzunehmen.

5.2 Lizenzierung

Das zu ertellende Programm wird unter den Bedingungen der sog. BSD- Open- Source Lizenz der Allgemeinheit zur Verfügung gestellt. Die Originalfassung dieser Lizenz ist im Internet unter der Adresse <http://www.opensource.org/licenses/bsd-license.html> in englischer Sprache abrufbar. Der Vollständigkeit halber ist hier eine von der Hochschule Rapperswil übersetzte Fassung eingefügt:

BSD- Lizenz (Deutsche, übersetzte Fassung)

Dieses Werk bezieht sich auf die Lizenzbestimmungen der "Non-copyleft Free Software", oft auch "BSD-Lizenz" genannt. Dieses Werk beinhaltet demnach keine Einschränkungen für den - auch kommerziellen – Gebrauch und die Weiterverbreitung von Quellcode und Programmen bis auf die folgenden Bedingungen:

- a.) Es muss beim Werk immer untenstehender Verweis auf die Lizenzgeber und die BSD-Lizenzbedingungen vorhanden sein;
- b.) Die BSD-Lizenzbedingungen beinhalten auch einen Garantiausschluss;
- c.) Fehler in diesem Werk müssen periodisch der unten angegebenen Kontaktadresse gemeldet werden und ausserdem darf
- d.) in einem abgeleiteten Werk nicht mit dem Namen des ursprünglichen Lizenzgebers geworben werden, es sei denn, dieser erklärt sich ausdrücklich damit einverstanden.

Copyright (c) 2001, HSR und Autoren. All rights reserved.

Kontakt: <http://www.integis.ch>.

5.3 Installationsanleitung

Damit das Programm ‚yagger‘ funktioniert müssen folgende Voraussetzungen erfüllt sein:

- Ein Java-Runtime- Environment (JRE) oder ein Java Development Kit (JDK) der Version 1.3 oder höher. Das Programm wurde aufgrund des Umstandes, dass die Version 1.4 zum Zeitpunkt der Entwicklung noch im Betastadium war, nicht darauf entwickelt und auch nicht getestet.
- Das Java Advanced Imaging- API (JAI) Version 1.1 muss im JRE/JDK installiert sein.
- Das Java3D- API muss in der Version 1.3 in der JRE/JDK installiert sein.
- Die JaGo- Bibliothek muss in Form von Quellen (bei Verwendung eines JDK) oder als JAR- Archiv verfügbar sein.
- Verschiedene Bibliotheken müssen zur Laufzeit zugänglich sein. Dies sind:
 crimson.jar
 jago.jar
 jaxp.jar
 latlon.jar
 xalan.jar

Die entsprechenden Pakete sind für verschiedene Betriebssysteme (Windows, Linux, SunOS etc.) bei <http://java.sun.com> verfügbar.

Bei einem angenommenen Startverzeichnis ‚/Viewer‘ und einem JDK im Verzeichnis ‚/jdk1.3.1‘ ist der vollständige classpath wie folgt:

```
/Viewer/classes; /Viewer/crimson.jar; /Viewer/jago.jar; /Viewer/jaxp.jar;
/Viewer/latlon.jar; /Viewer/xalan.jar; /jdk1.3.1/jre/lib/ext/jai_codec.jar;
/jdk1.3.1/jre/lib/ext/jai_core.jar;
/jdk1.3.1/jre/lib/ext/mlibwrapper_jai.jar; /jdk1.3.1/jre/lib/ext/j3dcore.jar;
/jdk1.3.1/jre/lib/ext/j3dutils.jar; /jdk1.3.1/jre/lib/ext/vecmath.jar;
/jdk1.3.1/jre/lib/i18n.jar; /jdk1.3.1/jre/lib/jaws.jar;
/jdk1.3.1/jre/lib/rt.jar; /jdk1.3.1/jre/lib/sunrsasign.jar;
/jdk1.3.1/lib/dt.jar; /jdk1.3.1/lib/htmlconverter.jar;
/jdk1.3.1/lib/tools.jar
```

Die Struktur der CD- ROM ist wie folgt:

