PostGIS und UMN MapServer: Professionelle Geo-Software

Frank Koormann
Intevation GmbH

Web-Applications im Intra- und Internet sind eine der Stärken Freier Software im kommerziellen Einsatz. Mit MapServer und PostGIS lassen sich Anwendungen zur Visualisierung von Geo-Informationen auf GNU/Linux entwickeln, welche professionellen Anforderungen genügen.

1. Einleitung

Daten fallen in unserer heutigen Welt in vielfältigster Form und Zahl an. Um aus diesen Daten Informationen zu gewinnen, d.h. in einer Entscheidungssituation eine präzise Antwort auf eine Frage zu bekommen, ist es notwendig, die Daten mit all ihren Merkmalen zu betrachten.

Ein Großteil der Daten haben einen Raumbezug. Für viele Fragestellungen ist es unabdingbar, diese Raumkomponente einzubeziehen. Der Umgang mit dieser Raumkomponente ist jedoch nicht trivial, so dass bei zu hohem Aufwand der Wert der Information sinkt oder sie gar wertlos werden läßt.

Deshalb sind für einer effiziente Entscheidungsunterstützung einige Voraussetzungen zu erfüllen:

- Die Daten werden zentral von Experten im Umgang mit Geo-Daten verwaltet. Diese stellen auch die Werkzeuge zur Verfügung, mit denen
- Benutzer ohne Detailkenntnisse der darunterliegenden Methoden Fragestellungen bearbeiten und so Informationen gewinnen.
- Da neue Fragestellungen auch neue Werkzeuge benötigen können, soll deren Distribution ohne aufwendige Installation möglich sein.

Diese Anforderungen lassen sich im Intranet bzw. Internet hervorragend lösen. Web-Applications sind eine Domäne Freier Software, und auch für das Handling von Geo-Daten sind Komponenten verfügbar: PostGIS und UMN MapServer.

2. PostGIS

PostGIS (http://postgis.refractions.net) erweitert PostgreSQL (http://www.postgresql.org/), zu einer räumlichen Datenbank. PostGIS ist Freie Software unter der GPL, die Entwicklung wurde von Refractions Research Inc. (http://www.refractions.net) (Kanada) initiiert und auch massgeblich vorangetrieben. Zur Zeit wird PostgreSQL 7.1.x unterstützt, eine Portierung von PostGIS auf PostgreSQL 7.2.x läuft gerade.

2.1. Was sind räumliche Datenbanken?

Die Entwicklung räumlicher Datenbanken geht auf die 90er Jahre des letzten Jahrhunderts zurück. Bereits vorher war die Speicherung reiner Koordinaten als einfache Daten in einer Datenbank möglich. Räumliche Datenbanken gehen jedoch weiter, sie speichern Geo-Daten als Geo-Objekte und können diese wie andere Objekte in der Datenbank bearbeiten:

- Geo-Objekte beschreiben entweder eine Position oder eine Form und lassen sich auf die Grundtypen Punkt, Linie und Polygon zurückführen.
- Operanden werten Relationen zwischen Geo-Objekten aus: "Grenzt an", "Schneidet", "Innerhalb", ...
- Methoden generieren aus einem oder mehreren Objekten weitere Daten: Längen, Flächen, Entfernungen, Mengenbildung, ...

Die Basisanforderungen an eine solche räumliche Datenbank und deren Schnittstellen hat das OpenGIS Consortium (OGC) (http://www.opengis.org) in der Simple Feature Specification (für OLE/COM, CORBA und SQL) festgelegt. PostGIS orientiert sich an

dieser Spezifikation und es ist geplant, PostGIS auch entsprechend zertifizieren zu lassen.

Das OGC ist eine internationale Organisation, deren Ziel die Spezifikation einheitlicher Schnittstellen und Protokolle zwischen verschiedenen GIS-Anwendungen ist. Natürlich finden sich in diesem Konsortium die Hersteller proprietärer GIS-Werkzeuge, aber auch Behörden und Universitäten. In den einzelnen Gremien wirken durchaus auch Entwickler Freier Sofware mit. Neben der Simple Feature Specification hat die OGC auch eine Spezifikation für Web Mapping Server herausgegeben, dazu unten mehr.

Geo-Daten wurden (und werden auch heute noch) oft in proprietären Dateiformaten im Dateisystem abgelegt. Dabei kann eine Datenbank abhängig von den Anforderungen im produktiven Einsatz wesentliche Vorteile bieten: Transaktionen, BackUps, Integritätsprüfungen, Vermeidung von Redundanzen, Multi-User-Unterstützung, Sicherheit, Zugriffskontrolle, Locking. Für einen performaten Zugriff auf die Daten können desweiteren die Index-Fähigkeiten der Datenbank genutzt werden.

PostGIS bietet im Zusammenspiel mit PostgreSQL viele dieser Vorteile: Datenbestände lassen sich zentral administrieren und der Zugriff auf die Daten kontrollieren. Durch Indizierung lassen sich auch große Datenbestände mit mehreren Millionen Objekten effizient verwalten.

2.2. Beispiele für PostGIS

2.2.1. Anlegen einer Tabelle

Anlegen einer Tabelle für Richtfunkmasten in der Datenbank `raumnutzung_db`: Zunächst wird die Tabellen mit Attributen angelegt, dann eine Spalte der Geometrie-Informationen hinzugefügt:

```
CREATE TABLE richtfunk ( name VARCHAR );
AddGeometryColumn('raumnutzung_db', 'richtfunk', 'location', 2167, 'POINT', 2);
```

2.2.2. Einfügen eines Records

```
INSERT INTO richtfunk VALUES ('Hannover/NDR-Studio',
                        GeometryFromText('POINT(3550200 58035320)', 2167));
```

2.2.3. Abfragen

Namen und Entfernung aller Richtfunkmasten im Umkreis von 4000 Metern um einen Standort in Hannover:

```
SELECT name, distance(location, 'SRID=2167;POINT(3550200 5803520)')
    FROM richtfunk
    WHERE distance(location, 'SRID=2167;POINT(3550200 5803520)) < 4000;

               name              |     distance
---------------------+------------
 Rifu-Üst. "Hannover/NDR-Studio" | 447.213595499958
(1 row)
```

Namen aller Richtfunkmasten in Hannover und Entfernung von einen Standort. Dazu ist eine Verschneidung der Richtfunkstandorte mit den Gemeindegrenzen notwendig:

```
SELECT name, distance(location, 'SRID=2167;POINT(3550200 5803520)')
    FROM richtfunk
    WHERE location && (SELECT the_geom
                           FROM gemeinden
                           WHERE gemeindenummer=3201000)
    ;
```

oder:

```
SELECT name, distance(location, 'SRID=2167;POINT(3550200 5803520)')
    FROM richtfunk
    WHERE truly_inside((SELECT the_geom
                           FROM gemeinden
                           WHERE gemeindenummer=3201000),
        location)
    ;
                       name                       |     distance
--------------------------------+------------
 Rifu-Üst. "TELEPORT/EUROP"                       | 7279.69779592532
 Rifu-Üst. "Hannover/Allbank"                     |  7837.3783882112
 Rifu-Üst. "Kronsberg/Hotel"                      | 6899.76811204551
 Rifu-Üst. "Han./Hannoversche Allgemeine Zeitung" | 4817.14645822607
 Rifu-Üst. "Hannover/Groß Buchholz"               | 5465.32688773513
 Rifu-Üst. "Hannover/Mittelfeld"                  | 5233.81314148681
 Rifu-Üst. "Hannover/Hemmingen"                   | 4112.38373695841
 Rifu-Üst. "Hannover/NDR-Studio"                  | 447.213595499958
 Rifu-Üst. "Hannover/Leinhausen"                  |   5534.265985657
 Rifu-Üst. "Hannover/Stöcken"                     | 7655.88009310491
 Rifu-Üst. "Hannover/Ahlem"                       | 5886.22969310577
(11 rows)
```

3. UMN MapServer

Der UMN MapServer (http://mapserver.gis.umn.edu) ist Freie Software unter einer MIT-ähnlichen Lizenz. Die Entwicklung nahm ihren Anfang im Fornet Projekt an der Universität von Minnesota (UMN), St. Paul, in Kooperation mit der NASA. Die UMN ist immer noch Zentrum der Entwicklung, inzwischen gibt es aber auch signifikante Beiträge von anderen Gruppen/Firmen.

3.1. Grundlagen

Der MapServer zeichnet auf der Basis von Vorgaben und Parametern aus den vorliegenden Geo-Daten das Bild einer Karte, diese wird über das Netz zum Benutzer geschickt und dort im Browser angezeigt. Dieses Vorgehen erleichtert den Umgang mit teuren Vektor-Geo-Daten, da diese nicht direkt zum Benutzer geschickt werden sondern nur eine Sicht auf die Daten. Auch unter Performance-Gesichtspunkten hat dieses Konzept Vorteile: Benutzer agieren häufig mit relativ wenig Aktionen auf den Karten, so dass das durch die Grafiken transferierte Datenvolumen deutlich geringer ist als der Umfang der gesamten Datensätze.

3.1.1. Portabilität

Der UMN MapServer ist in C implementiert und bietet schon auf Grund seiner Konzeption eine hohe Portabilität: Der UMN MapServer läßt sich auf Servern unter GNU/Linux, diversen Unices und auch MS-Windows Servern einsetzen und arbeitet (abhängig von der Anwendungsart) mit nahezu allen Webservern zusammen. Dies ist ein signifikanter Vorteil gegenüber proprietären Herstellern, die zwar langsam auch GNU/Linux als Server-Plattform entdecken, sich aber dabei auf Intel-basierte beschränken und oft nur ausgewählte Webserver unterstützen.

3.1.2. Flexible Anwendung

Die Funktionalität des MapServers läßt sich serverseitig verschieden nutzen, die einfachste Form als reines CGI-Binary ist auf allen Web-Servern mit CGI-Support einsetzbar. Umfangreichere und komplexere Anwendungen können mit MapScript realisiert werden. Mittels SWIG wird hier das API des MapServers für Sprachen Perl, Python, Java oder Tcl/Tk bereitgestellt. Desweiteren gibt es inzwischen eine Schnittstelle für PHP. Mit dem MapServer implementierte Anwendungen arbeiten

zustandslos, alle Parameter werden in der URL gehalten. Damit kann der MapServer in die verschiedensten bestehenden Web-Application Umgebungen integriert werden.

3.1.3. OpenGIS WMS Kompatibilität

Auch für das Web-Mapping hat das OpenGIS Consortium Schnittstellen spezifiziert (WMS Specification (http://www.opengis.org/techno/specs/01-047r2.pdf)). Ein WebMapServer (WMS) erlaubt den Zugriff auf verschiedene Datenquellen und ermöglicht so ein Netzwerk von Servern, aus dem sich ein Benutzer variabel Karten zusammenstellen kann. Der UMN MapServer ist WMS kompatibel und kann sowohl als Client arbeiten (also Datenteile von anderen Servern abfragen und diese dann für eine Karte zusammenfügen), so wie als Server (auf Anfrage Datenteile liefern).

3.2. Funktionalität

3.2.1. Daten

Als Freie Software kann der UMN MapServer die Möglichkeiten der verschiedenen freien Bausteinen zum Handling von Geo-Daten nutzen und bietet damit Zugriff auf die gängigsten Datenformate:

- Vektor-Formate:

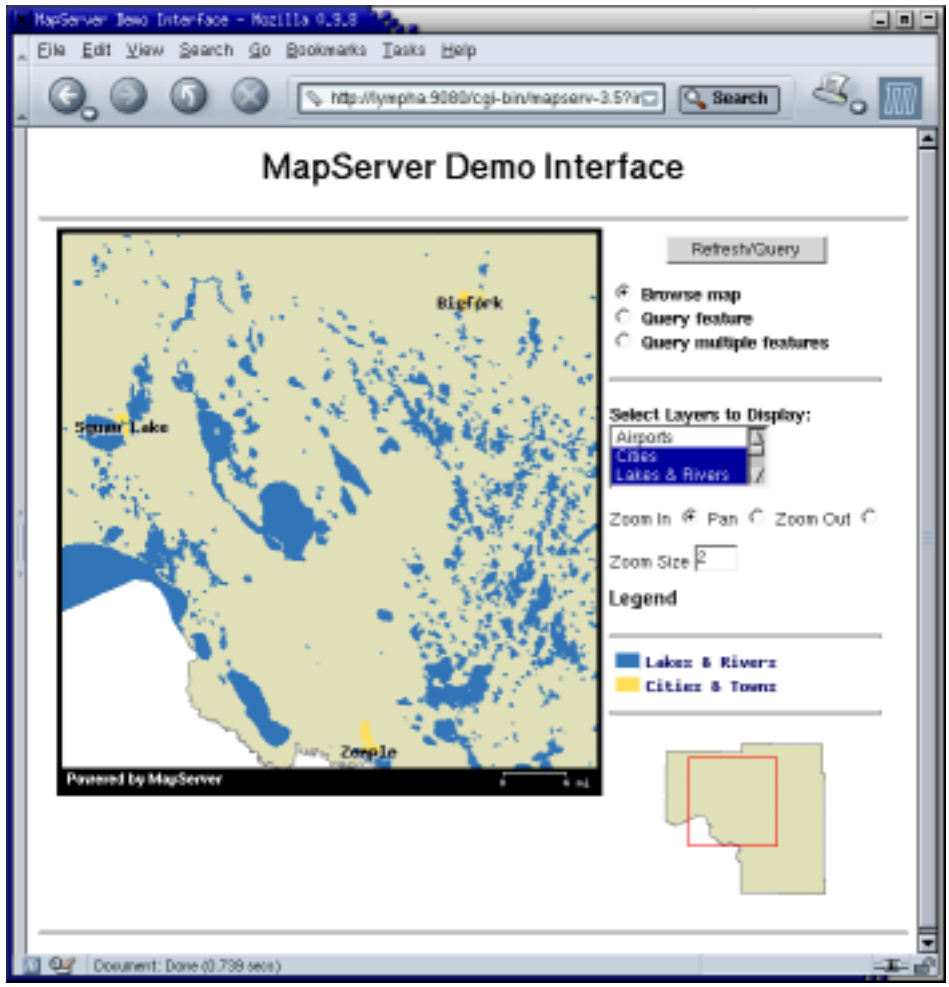
  ESRI Shapefile, Mapinfo File, Arc/Info Binary Coverage, Microstation DGN, UK .NTF, SDTS, U.S. Census TIGER/Line

- Raster-Formate:

  TIFF/GeoTIFF, PNG, ERDAS

- Datenbanken:

  PostGIS, Oracle Spatial, ESRI SDE



3.2.1. Konfiguration

3.2.1.1. Map-File

Die Basis-Konfiguration des MapServers wird für eine Anwendung in der MAP-Datei festgelegt: Über die Attribute von Objekten werden Aussehen, Zugehörigkeiten und auch Pfade spezifiziert. Die einzelnen Geo-Datensätze werden dabei in Datenebenen (Layern) organisiert:

```
LAYER
    NAME kreisgrenzen
    TYPE POLYGON
    STATUS ON
    DATA ../kreise
```

```
    MINSCALE 50000
    CLASS
        NAME "Kreisgrenzen"
        OUTLINECOLOR 0 0 0
  COLOR 230 230 230
    END
END # kreisgrenzen

LAYER
 NAME kreisstrassen
 TYPE LINE
 STATUS ON
    CONNECTIONTYPE postgis
    CONNECTION "user=maps password=**** dbname=kreise host=localhost port=5432"
 DATA "geometry FROM kreisstrassen"
 CLASS
  NAME "Kreisstrassen"
  SYMBOL 'solid-line'
  COLOR 128 128 0
  SIZE 5
 END
END # kreisstrassen
```

Einzelne Werte dieser Basiskonfiguration lassen sich on-the-fly für die nächste Karte in der URL ändern.

3.2.1.2. Templates

Die einfachste Variante der Anwendungsentwicklung basiert auf HTML-Templates im Zusammenspiel mit dem CGI-Binary des UMN MapServers: In den Vorlagen wird die Benutzer-Oberfläche mit einfacher Funktionalität vorgegeben, Zustandsvariablen werden durch eckige Klammern markiert (z.B. [scale] gibt den Massstab der dargestellten Karte an):

```
<p>
Zoom In  <input type=radio name=zoomdir value=1 [zoomdir_1_check]>
Pan      <input type=radio name=zoomdir value=0 [zoomdir_0_check]>
Zoom Out <input type=radio name=zoomdir value=-1 [zoomdir_-1_check]>
<p>
Zoom Size <input type=text name=zoomsize size=4 value=[zoomsize]>
<p>
```

(Ausschnitt aus einem Template: Navigationsmodi (ZoomIn, ZoomOut, Pan) per Radiobutton)

Auch in der template-basierten Variante lassen sich bereits Benutzeroberflächen mit Java-Applets für komfortablere Navigation realisieren. Entsprechende Applets (MApplet, Rosa-Applet) sind als Freie Software verfügbar.

3.2.1.3. MapScript

Wenn das durch die Template/CGI-Binary-Kombination vorgegebene Konzeption nicht passt, lassen sich Anwendungen durch MapScript realisieren. Für die gängigen Sprachen für CGI-Skripte steht damit ein API zur MapServer-Funktionalität zur Verfügung.

Z.B. Perl:

```
#!/usr/bin/perl
use mapscript;

$map = new mapObj('europe.map) or die('Unable to open mapfile.');
$img = $map->draw() or die('Unable to draw map');

mapscript::msSaveImage($img, 'europe.png',
         2, $map->{interlace}, $map->{transparent}, 95);

print header();
.....
```

Gleiches in PHP MapScript:

```
<?php

 dl('php_mapscript.so');
 $map_path="/var/www/html/ms/map_files/";
 $map = ms_newMapObj($map_path."europe.map");
 $image=$map->draw();
 $image_url=$image->saveWebImage(MS_PNG,1,1,0);
?>

<HTML>
 <HEAD>
  <TITLE>Example 1: Displaying a map</TITLE>
 </HEAD>
 <BODY>
  <IMG SRC=<?php echo $image_url; ?> >
 </BODY>
</HTML>
```

Das Beispiel macht bereits deutlich, dass beide MapSkript-Varianten nicht gleich sind: Während Module für Perl, Python, Java, etc. mittels SWIG direkt aus den MapServer-Sourcen generiert werden, ist PHP MapScript eine eigenständige Implementierung, die sehr nah an der bestehenden MapServer Entwicklung verläuft

3.2.2. Darstellung

Neben der reinen Darstellung der Karte als Grafik (PNG, JPEG, auch PDF) können weitere Elemente gegeriert werden: ein Massstabsbalken, eine Legende der

dargestellten Layer und eine Referenzkarte, auf der sich der dargestellte Kartenausschnitt besser räumlich einordnen läßt.

Der UMN MapServer kann automatisch einzelne Layer abhängig vom Massstab anzeigen. So können zu einem Thema (z.B. Strassen) verschiedene Layer konfiguriert werden, so dass in einer Übersicht nur die Hauptstrassen und erst bei grösseren Massstäben auch die Nebenstrassen zu sehen sind. Damit kann je nach Massstab eine optimale Darstellung erzielt werden (nicht zu detailiert, aber auch nicht zu grob).

Für Beschriftungen und auch Symbole für Punktinformationen bietet der UMN MapServer TrueType Unterstützung. Besonderheiten sind die optionale Ausrichtung der Beschriftung an Objekten und eine Kontrolle, um Überlagerungen zu vermeiden.

3.2.3. Analyse

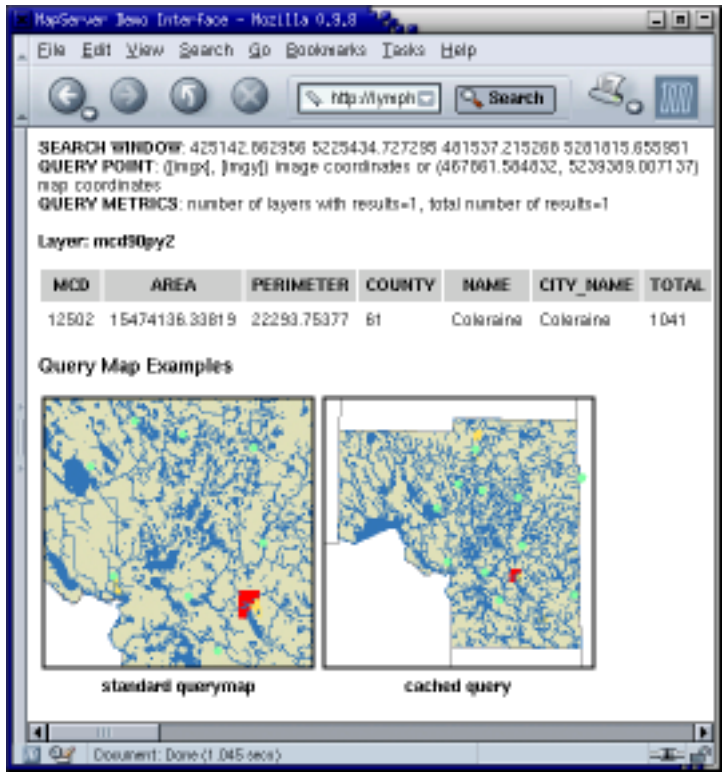
3.2.3.1. Thematische Karten

In den obigen Beispielen wurden je Layer nur eine Klasse benutzt. Für eine differenziertere Darstellung können zu einem Layer mehrere Klassen definiert werden. Die Zugehörigkeit der einzelnen Geo-Objekt läßt sich über reguläre und logische Ausdrücke kontrollieren:

```
...
 CLASSITEM "strassentyp"

 CLASS
  NAME "Autobahn"
  EXPRESSION /BAB/
...

...

 CLASS
  NAME "500 - 1000 / m^2"
  EXPRESSION ( [BEV_DICHTE] >= 500 AND [BEV_DICHTE] < 1000 )
...
```

3.2.3.2. Queries

Einzelne Objekte lassen sich durch "Anklicken" oder über Wertevorgaben selektieren. In dieser Auswahl besteht dann Zugriff auf alle Attribute der Objekte, so dass sich weitere Informationen abfragen lassen.



3.2.4. Optimierung

Der UMN MapServer bietet zwei Ansatzpunkte, an denen sich die Performance einer Anwendung optimieren läßt: Tile-Indices und Indexbäume

3.2.4.1. Tile-Indices

Mit einem Tile-Index lassen sich mehrer Daten-"Kacheln" zu einem grossen virtuellen Datensatz zusammenfassen: So könnten z.B. die Gemeindegrenzen in der Bundesrepublik Deutschland in Datensätzen für die einzelnen Länder zusammengefasst sein (ganz einfach weil unterschiedliche Vermessungsämter für diese Daten zuständig sind). Statt nun für eine deutschlandweite Anwendung 16 verschiedene Layer zu definieren, kann ein Tile-Index erzeugt werden, über den dann auf die verschiedene Datensätze zugegriffen wird.

Interessanter Nebeneffekt: Für das Rendering muss der UMN MapServer die Sichtbarkeit eines jeden Objekts bewerten, das sich in dem Datensatz befindet. Durch einen Tile-Index wird hier schon eine Vorauswahl getroffen, ist z.B. klar, dass nur

Bereiche aus dem Datensatz Niedersachen dargestellt werden müssen, reduziert sich die Zahl der zu bewertenden Objekte bereits beträchtlich.

3.2.4.2. Index-Bäume

Dieses Konzept läßt sich über Index-Bäume noch verfeinern: Diese indizieren die Geo-Objekte innerhalb eines Datensatzes. Der UNM MapServer kann diese Index-Bäume auswerten und somit die Sichtbarkeitsanalyse weiter auf den tatsächlich darzustellenden Bereich einschränken. Für Daten, die über eine Datenbankverbindung zu PostGIS bereitgestellt werden, übernimmt die Datenbank diesen Schritt.

3.3. Leistung

Der UMN MapServer ist ein performates Werkzeug für den Aufgabenbereich Web-Mapping. Da sich die Implementierung auf genau diese Aufgabenstellung konzentriert und kaum unnötigen Ballast zugunsten vermeintlicher Administratorenfreundlichkeit beinhaltet oder in komplexere Applikationsumgebungen eingebunden ist, sind Anwendungen sehr schnell. Vergleiche mit Konkurrenzprodukten beobachteten bis zu acht-facher Leistungssteigerung.

Der UMN MapServer skaliert sehr gut. Für eine einfache Anwendung mit einigen tausend Anfragen pro Tag ist ein kleiner Server ausreichend, es gibt aber auch Installationen, die auf Mehrprozessorsystemen über 100 000 Anfragen pro Tag bedienen.

4. Vortrag

Im Vortrag wird ein Auskunftssystem demonstriert, dass bei der Bezirksregierung Hannover im Einsatz ist und dort zu verschiedenen Fragestellungen Zugang zu den

Geo-Daten bietet. Damit können die für unterschiedliche Entscheidungssituationen notwendigen Informationen abgerufen werden.