Praktikumsbericht zum Programmierpraktikum im Hauptstudium

Implementation einer Installationsroutine zum GiST-Framework für Oracle DB

14. August 2006

Lehrstuhl für Verteilte Informationssysteme
Prof. Dr. Harald Kosch

Betreuer: Dr. Mario Döller
Praktikantin: Kerstin Renner

Inhaltsverzeichnis

1 Das GiST-Framework

(nach [Doe02] und [Kos05])

Das Oracle Datenbank-System bietet durch seine Data Cartridge-Technologie gute Möglichkeiten, verschiedene Teile des Systems, z.B. das Typsystem oder die Indexierung der Daten, zu erweitern. Das GiST-Framework (Generalized Search Trees) in seiner in diesem Praktikum verwendeten Weiterentwicklung ist eine solche Data Cartridge und dient zur Erweiterung des Indexing Systems.
Die MDC (Mutlimedia Data Cartridge) besteht aus drei Komponenten: dem GistService, dem GistWrapper und der Erweiterung der Datenbank selbst. Letztere führt die eigentliche Indexierung aus und liegt in drei Varianten vor: als Standard GiST-Index, als Index für Audio-Dateien und als Index für CLOBs (Character Large Object)

Das ursprüngliche GiST (siehe [Hel99]) wurde an der University of California, Berkeley entwickelt. Es ermöglicht dem Benutzer, jeden beliebigen balancierten Baum als Indexstrukur aufzubauen, indem lediglich die Methoden zum Einfügen, Löschen und Suchen implementiert werden. In der aktuellen Version sind bereits Implementationen für den R-, R*-, SS-, SR-, SP- und B-Baum enthalten.

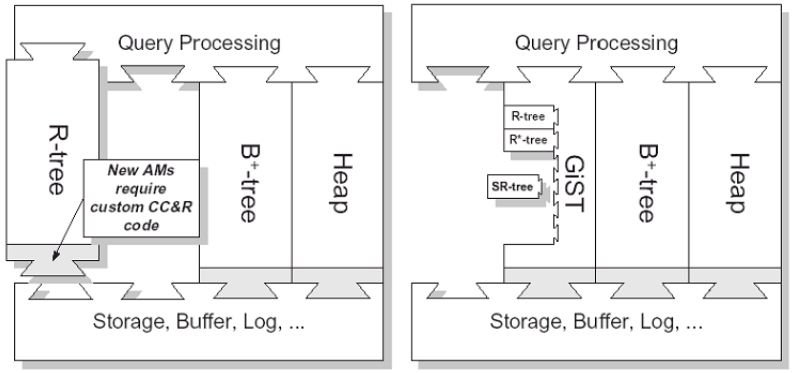

Abbildung 1: *Diese Grafik verdeutlicht, wie viel aufwändiger es ohne das GiST-Framework ist, eine neue Indexstruktur in eine Datenbank einzubauen. Ohne das Framework muss man sich mit den umfangreichen Schnittstellen der Datenbank auseinandersetzen und diese bei der Implementation berücksichtigen. Arbeitet man mit dem Framework, wurde diese Arbeit bereits erledigt, man implementiert nur noch die Datenstruktur an sich, und muss nur die wesentlich kleinere Schnittstelle zu GiST berücksichtigen.*

1.1 Der GistService

Der in C++ implementierte GistService läuft als Windows-Service und ist der Hauptbestandteil des Frameworks außerhalb der Datenbank. Er bildet dort die

Softwareumgebung für GiST. Der Serivce besteht aus zwei Teilen: dem GistCommunicator und dem GistHolder.
Der GistCommunicator basiert auf der COM-Technologie (Component Object Model) von Microsoft. Er bildet die Kommunikationsschnittstelle zwischen den Methoden des Services und dem GistWrapper. Die wichtigsten Methoden, die beiderseits zur Verfügung stehen, sind:

`gistCreate(char*, int)`	Erstellt einen Index mit bestimmtem Namen und Dimension.
`gistInsert(char*, char*, int)`	Einfügen eines Datums mit bestimmtem Schlüssel in den Baum mit einer bestimmten ID.
`gistClose(int)`	Schließt die Verbindung zu dem Baum mit der angegbenen ID.
`gistRemove(char*, int)`	Entfernt die Einträge, die die angebene Query zurückgibt, aus dem Baum mit einer bestimmten ID.
`gistOpen(char*)`	Öffnet die Verbindung zu dem Baum mit der angegebenen ID.
`gistIsEmpty(int)`	Prüft ob der angebenene Baum leer ist.
`gistQuery(char*, int, char*, int, int, int, int, char*)`	Führt eine Query mit den angegebenen Daten (Query-Name, Dimension, gesuchten Werten, benutzer Funktion und deren Argument, Anzahl der gewünschten Ergebnisse, Baumnummer) aus.

Diese neue Architektur ermöglicht im Gegensatz zum ursprünglichen GiST-Framework die Verwaltung von mehreren Indexbäumen gleichzeitig.
Dies wiederum wird durch den GistHolder ermöglicht. Er verwaltet die Bäume durch IDs, welche beim Zugriff auf einen Index durch einen Prozess mit angegeben werden. Die Bäume werden intern in einen Array gespeichert, jedoch kann dieser gegen jede gewünschte Datenstruktur ersetzt werden.
Eine schematische Darstellung des Services findet sich in Abbildung 2.

1.2 Der GistWrapper

Der GistWrapper ist eine DLL-Datei, die es der Datenbank ermöglicht, mit dem GistService zu kommunizieren. Er nutzt Oracles Möglichkeit, externen C/C++-Code über gemeinsame Programmbibliotheken auszuführen. Zum einen bietet der Wrapper den Prozeduren der Datenbank Zugriff auf den Service und zum anderen konvertiert er die Ein- und Ausgaben so, dass sie für die jeweils andere Seite (Datenbank oder Service) brauchbar sind.

1.3 Die Erweiterung der Datenbank

Die Erweiterung der Datenbank ist aufwändig und besteht aus mehreren Schritten. Sie werden im Abschnitt 2.4.4 erläutert, da das Installationstool diese Schritte nachvollziehen muss.

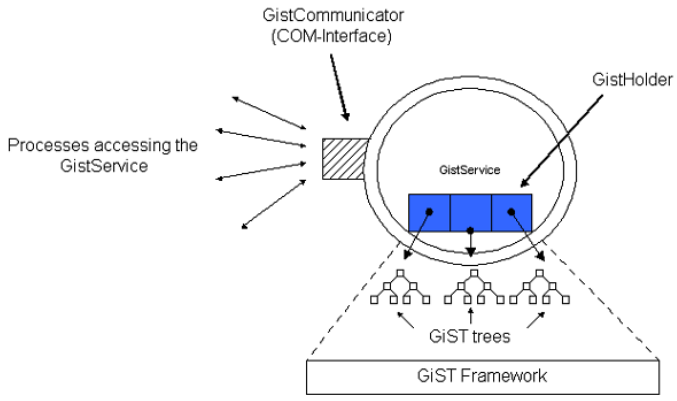

Abbildung 2: *Aufbau des GistServices*

2 Das Installationstool

Das Ziel des Praktikums war es, eine Windows-Installationsroutine für das in Kapitel 1 beschriebene Framework zu erstellen. Sie wurde in Java (JDK 1.5.0) implementiert und besteht aus den folgenden Schritten.

2.1 Eingabe des Installationspfades

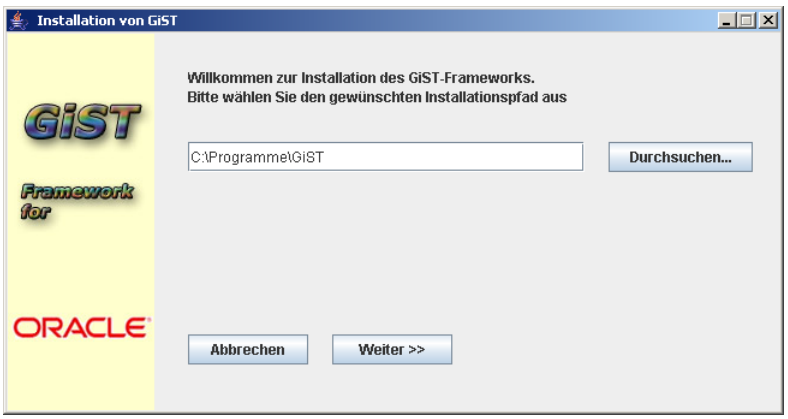

Abbildung 3: *Das Fenster zur Eingabe des Installationspfades*

Abbildung 3 zeigt den ersten Schritt der Installation, bei dem das Installationsverzeichnis anzugeben ist. Das Programm schlägt ein Standardverzeichnis vor, welches sich aus der ausgelesenen Windows-Umgebungsvariable `%PROGRAMFILES%` (auf deutschen Windows-Systemen ist dies standardmäßig `C:\Programme`) und dem Ordnername GiST zusammensetzt. Man kann das gewünschte Verzeichnis

direkt in das Textfeld eintragen oder es in einem durch „Durchsuchen...“ aufgerufenen Dialog auswählen.
Durch einen Klick auf „Weiter>>“ wird der Benutzer zum nächsten Fenster weitergeleitet, jedoch erst nach Validierung des angegebenen Pfades.
Es wird überprüft, ob er die Sonderzeichen `/ : * ? " < > | \` oder Leerzeichen enthält.
Ist dies der Fall, so wird eine Fehlermeldung angezeigt (siehe Abbildung 4), das gültige Standardverzeichnis erscheint wieder in dem dafür vorgesehenen Textfeld.
Ist die Pfandangabe gültig, wird der angegebene Ordner erstellt. Der Pfad wird an den nächsten Schritt übergeben, um später die benötigten Dateien hineinzukopieren, oder aber den Ordner wieder zu löschen, sollte die Installation abgebrochen werden.
Der Button „Abbrechen“ ruft - ebenso wie der Beenden-Button in der Titelleiste - die unter 2.6 beschriebene Sicherheitsabfrage auf. Diese Funktion erfüllt er auch in allen weiteren Fenstern, und wird deshalb nicht erneut erwähnt.

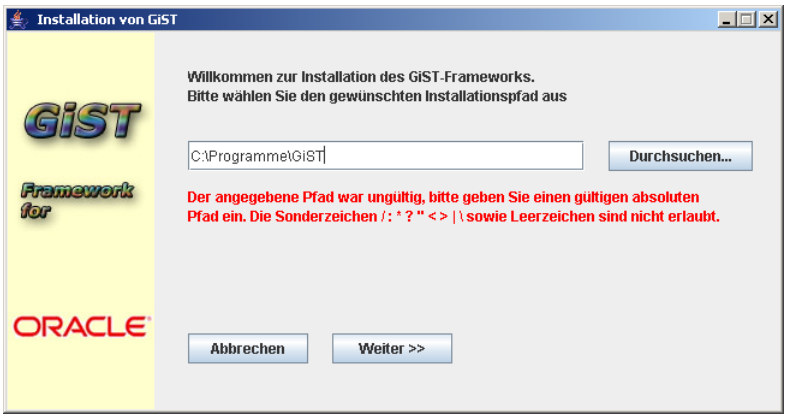

Abbildung 4: *Der eingegebene Pfad war ungültig*

2.2 Eingabe der Zugangsdaten für die Datenbank

In diesem in Abbildung 5 gezeigten Fenster muss der Benutzer die Zugangsdaten für die Datenbank angeben. Dazu gehören der Name der Datenbank, das SYS-Passwort, Username und Passwort des Users, für den das Framework installiert werden sollen. Es wird vorausgesetzt, dass der angegebene User bereits existiert und mindestens connect-Rechte besitzt.
Sind beim Betätigen des „Weiter>>“-Buttons die Daten ungütlig, läuft die Datenbank nicht, oder existiert der User nicht, so wird eine Fehlermeldung angezeigt (siehe Abbildung 6).
Durch den Button „<<Zurück“ kann der Benutzer zur Pfadeingabe zurückkehren, um diese zu verändern. Der bereits erstellte Ordner wird zu diesem Zweck

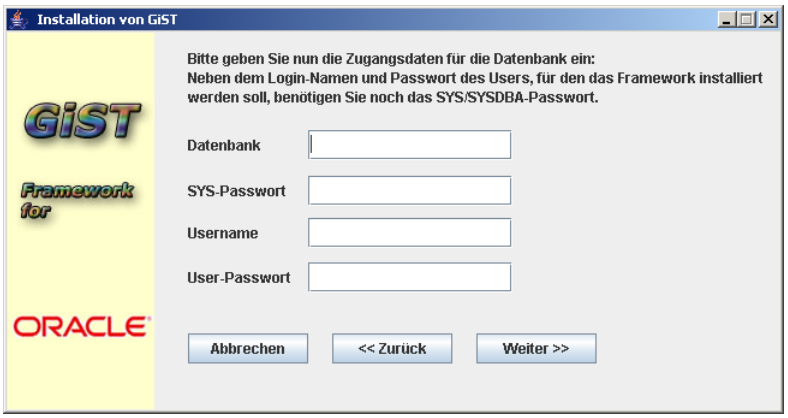

Abbildung 5: *Das Fenster zur Eingabe der Datenbank-Zugangsdaten*

wieder gelöscht.
Mit dem „Weiter>>“-Button startet nach Eingabe korrekter Zugangsdaten schließlich die eigentliche Installation.

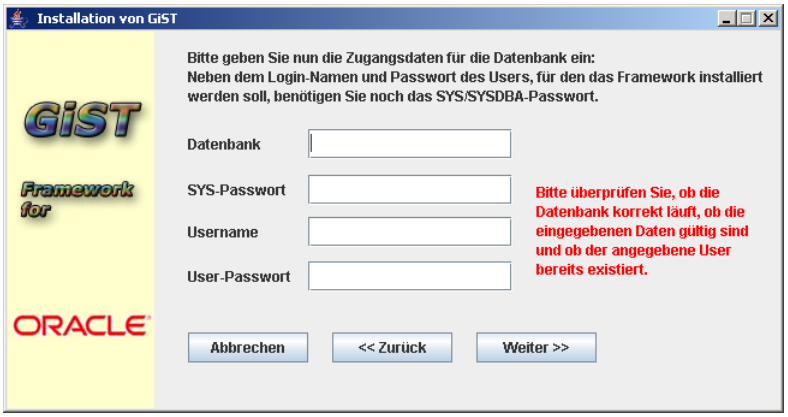

Abbildung 6: *Die eingegebenen Daten waren ungültig. Es besteht aber auch die Möglichkeit, dass die Datenbank nicht korrekt läuft.*

2.3 Auswahl der Indexart

Da mit dem Framework drei Indexarten zur Verfügung stehen, muss eine Auswahl stattfinden. Dies ist in diesem Schritt, der in Abbildung 7 zu sehen ist, der Fall. Die Auswahl erfolgt über ein Dropdown-Feld, dessen Wert bei einem Klick auf „Weiter>>“ ausgelesen und an das nächste Fenster übergeben wird.

Abbildung 7: *In einem Dropdown-Feld wird der gewünschte Index ausgewählt*

2.4 Fortschrittsanzeige der Installation

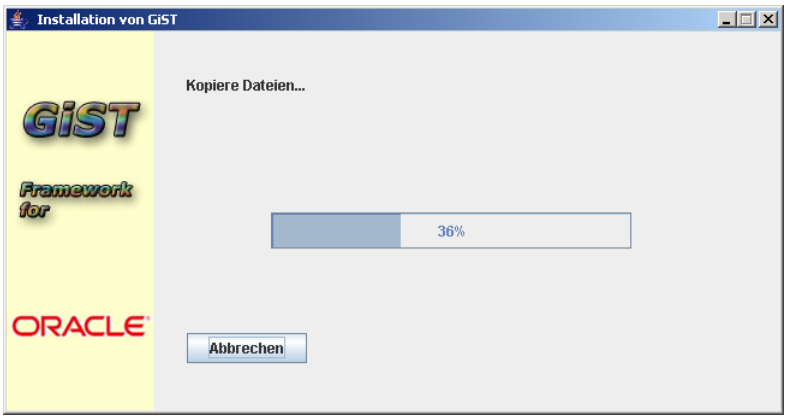

Abbildung 8: *Das Fenster mit dem Fortschrittsbalken zur Installation*

Während der Installation informiert, wie in Abbildung 8 zu sehen, eine JProgressBar den Benutzer über den Fortschritt. Da dies leider nicht automatisierbar ist, müssen die Prozentwerte manuell nach jedem Installationsschritt neu gesetzt werden. Dies funktioniert nur, wenn die ProgressBar mit den auszuführenden Befehlen in einem eigenen Thread läuft.
Folgende Schritte werden während der Installation ausgeführt:

2.4.1 Erstellen des benötigten Verzeichnisses

Der GistService enthält einen fest eincodierten Pfad `c:\temp\gist_service`, wo er neben einer Konfigurations- und einer Debug-Datei auch Abbilder der erstellten Indexbäume ablegt. Sie bleiben so auf der Festplatte gespeichert, auch wenn die Datenbank heruntergefahren wird. Die Dateinamen entsprechen dabei dem Schema `<username>.<indexname>`

Das oben erwähnte Verzeichnis wird in diesem Schritt durch die Installationsroutine erstellt.

2.4.2 Kopieren der Dateien

In diesem Schritt werden alle für die Ausführung des Services benötigten Dateien kopiert. Die Datei `gistwrapp.dll` wird in den in 2.4.1 erstellten Ordner kopiert, die ausführbaren Dateien `TestService.exe` (der GistService), `srvany.exe` (ein Hilfsprogramm, das es ermöglicht, jedes Programm als Windows-Service zu starten) und `ServiceClient.exe` (ein kleiner Test-Client, mit dem die Funktion des GistServices überprüft werden kann) werden in dem Installationsordner abgelegt, der im ersten Schritt (2.1) erstellt wurde.

2.4.3 Modifikation der Windows-Registry

Um den GistService als Windows-Service zu installieren, sind Änderungen an der Registry nötig. Um diese entsprechend zu modifizieren, wird eine Registrierungsdatei (*.reg) verwendet. Für diese Datei liegt ein Template vor, in welches der Installationspfad und der nach hexadezimal konvertierte Pfad zu der Datei `srvany.exe` eingefügt wird. Das so veränderte Template wird dann als valide Registrierungsdatei gespeichert und mit dem Konsolenbefehl `regedit /s gist-install.reg` in die Registry eingetragen.

2.4.4 Einspielen der SQL-Files in die Datenbank

Nach dem Service werden die für das GiST-Framework nötigen Funktionen, Prozeduren, Index-Typen etc. in der Datenbank erstellt. Die SQL-Dateien werden durch das Java-Programm eingelesen und jedes Statement einzeln ausgeführt. Dies ermöglicht einfaches Einfügen von Username und Passwort bei den Befehlen, wo sie benötigt werden.
Zunächst werden die sicherheitsrelevanten Aktionen vorgenommen. Dem User werden durch den SYSDBA die notwendigen File- und PolicyTablePermissions erteilt und anschließend durch den User selbst bestätigt.
Danach werden die Aktionen ausgeführt, die für alle drei Indexarten notwendig sind. Die GistWrapper-Bibliothek wird mit dem Befehl `CREATE LIBRARY` und einem Verweis auf die DLL-Datei eingebunden. Um das Funktionieren der Kommunikation zwischen Service und Datenbank zu gewährleisten, müssen dann zusätzlich Funktionen definiert werden, die ihre erhaltenen Eingaben wiederum an die Funktionen der DLL weiterleiten.
Darauf folgt das Erstellen der Elemente, die sich bei den drei Index-Varianten unterscheiden. Für jede Indexart existiert ein eigenes SQL-File, und je nach Auswahl in 2.3 wird die entsprechende Datei eingespielt. Alle drei dieser Dateien enthalten die Operatoren für die benutzten Baumstrukturen (z.B. in der Familie der R-Bäume Gleichheit oder Nächster Nachbar-Suche für d-dimensionale Punkte) und die Funktionen zu diesen Operatoren. Ferner den Typ und den dazugehörigen Type-Body für die ausgewählte Indexart. Der Typ verweist auf die Funktionen in der zu der Indexart gehörenden Java-Klasse. Zuletzt werden die Indextypen (R-, R*-, SR-, RR-, SS- und SP-Baum bei Standard- und CLOB-Index, ein auf Audio-Dateien zugeschnittener R-Baum beim Audio-Index) mit den Verweisen auf die zu benutzenden Operatoren erstellt.

Eigentlich sollten in diesem Schritt der Installation auch zwei JAR-Files mit dem Befehl `loadjava` eingespielt werden. Da dies aus unerklärlichen Gründen aus Java heraus weder als Konsolenbefehl noch durch ein Batch-File funktioniert, geschieht dies erst nach Ende der eigentlichen Installation (siehe 2.7).

2.4.5 Modifikation der Datei listener.ora

Um dem Datenbanksystem einen korrekten Zugriff auf die Wrapper-DLL zu gewährleisten, muss die Datei `listener.ora`, welche sich im Verzeichnis `ORACLE_HOME\NETWORK\ADMIN` befindet, um den Eintrag

```
(ENVS="EXTPROC_DLLS=ANY")
```

ergänzt werden.
Dazu ist es nötig, die Variable `ORACLE_HOME` auszulesen. Dies erwies sich als äußerst schwierig, da nicht automatisch eine Windows-Umgebungsvariable dafür angelegt wird. Die Information befindet sich in der Windows-Registry. Das Auslesen wäre mit Java möglich, jedoch ist der Registrierungsschlüssel, in dem sich diese Variable befindet, von der Oracle-Version anhängig. Da die Installation aber nicht nur mit der aktuellen Version 10g funktionieren soll, ist das Auslesen aus der Registry also auch nicht ohne weiteres möglich.
Unter [Mun04] wird eine unkonventionelle Art dargestellt, an den Inhalt von `ORACLE_HOME` zu gelangen. So erhält man in SQLPlus nach Eingabe des Befehls

```
@.[\%ORACLE_HOME\%]
```

die Fehlermeldung mit dem Fehlercode SP2-0310:

```
unable to open file ".[D:\oracle\product\10.1.0\Db_1].sql"
```

welche das `ORACLE_HOME`-Verzeichnis enthält. Speichert man diesen Befel in einer SQL-Datei und führt man diese über

```
Process p = Runtime.getRuntime().exec(
         "cmd /c sqlplus user/passwd@database @read_orahome.sql")
```

aus, ist es möglich die Konsolenausgabe durch `p.getInputStream()` einzulesen. Aus diesem eingelesenen String kann dann das gewünschte `ORACLE_HOME`-Verzeichnis, das in der Fehlermeldung in eckigen Klammern steht, extrahiert werden.
Damit ist der absolute Pfad zu der Datei `listener.ora` bekannt, und sie kann eingelesen und verändert werden.

2.4.6 Erstellen einer Log-Datei

Um eine spätere Deinstallation des Frameworks zu ermöglichen, müssen einige Werte (Installationspfad, Indexart, User, Userpasswort und Datenbank-Name) in einer Log-Datei gespeichert werden. Diese wird in diesem letzen Schritt der eigentlich Installation erstellt. Sie hat den absoluten Pfad `c:\temp\gist_service\install.log`. Dieser fest eincodierte Pfad des Services bietet sich zu diesem Zweck natürlich an.
Eine solche Log-Datei sieht dann wiefolgt aus:

```
**********************************
**DO NOT EDIT OR DELETE THIS FILE**
**********************************
[PATH]c:\Programme\GiST[/PATH]
[INDEX]Standard[/INDEX]
[USER]mdc_user[/USER]
[USERPASS]mdc_user[/USERPASS]
[DB]orcl[/DB]
```

User und Passwort sind in diesem Fall sofort erkennbar, was natürlich nicht wünschenswert ist. Es ließe sich durch einen Mehraufwand eine gewisse Codierung oder Verschlüsselung einbauen, jedoch muss diese dann auch bei der Deinstallation wieder rückgängig gemacht werden, was ebenfalls einen Mehraufwand bedeutet. Hier wurde allein der Einfachheit halber darauf verzichtet.

2.5 Abschluss-Fenster

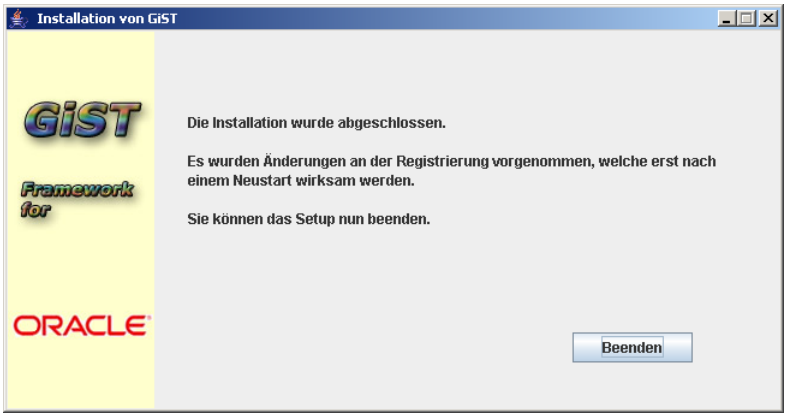

Abbildung 9: *Das letzte Fenster informiert über das Ende der Installation und bittet den User, den Rechner neu zu starten.*

Zuletzt gibt das Programm noch die Meldung aus, dass die Installation abgeschlossen wurde (siehe Abbildung 9). Durch einen Klick auf „Beenden“ schließt sich das Fenster und das Java-Programm wird beendet.
Anschließend ist ein Windows-Neustart nötig, damit die an der Registry vorgenommenen Änderungen greifen und somit der Service automatisch gestartet wird.

2.6 Sicherheitsabfrage

Abbildung 10: *Die Sicherheitsabfrage ist ein modaler Dialog, der unbeabsichtigtes Abbrechen der Installation vermeiden soll.*

Um ein versehentliches vorzeitiges Beenden der Installation zu verhindern, beenden weder das Beenden-Symbol in der Titelleiste noch der „Abbrechen“-Button das Programm sofort, sondern rufen lediglich die in Abbildung 10 gezeigte Sicherheitsabfrage auf. Erst wenn diese mit „Ja“ beantwortet wird, wird das Programm tatsächlich abgebrochen. Bei „Nein“ kehrt es wieder zum letzten Fenster zurück.
Dieser Dialog wurde im Gegensatz zu den anderen Fenstern nicht von der Swing-Klasse JFrame abgeleitet, sondern von JDialog, welche es erlaubt, den Dialog modal anzuzeigen, d.h. es ist nicht möglich auf das darunter liegende Fenster desselben Programms zuzugreifen, so lange nicht eine Option ausgewählt, bzw. ein Button geklickt worden ist.

2.7 Einspielen der JAR-Files

Wie in 2.4.4 beschrieben, ist es aus Java heraus nicht möglich, JAR-Files so in die Datenbank zu laden, dass diese auch funktionieren. Um dieses Problem zu umgehen, kann man sich mehrerer Batch-Dateien bedienen. Die erste dient nicht nur dazu, die beiden anderen aufzurufen, sondern ermöglicht nebenbei auch das Starten der Installation per Doppelklick.

```
cd Installer2
java -cp .;ojdbc14.zip Start
call load_mdc_common.bat
call load_mdc_indexlib.bat
```

Nach Beendigung der Java-Anwendung werden nacheinander die beiden anderen Batch-Dateien aufgerufen. Da diese auch die Zugangsdaten für die Datenbank enthalten müssen, werden sie erst während der Installation erstellt.
Der Versuch, beide `loadjava`-Befehle durch eine einzelne Batch-Dateien auszuführen, war gescheitert, deshalb wird nur jeweils ein solcher Befehl pro Datei aufgerufen.
Die JAR-Dateien enthalten zum einen Utility-Klassen, die eine Datenbankverbindung oder das Debuggen vereinfachen, zum anderen die Java-Klassen der verschiedenen Indexarten mit den dazugehorigen ODCI-Methoden. Letztere rufen über die in der Datenbank vorliegenden Funktionen wieder die Funktionen der Wrapper-DLL auf

3 Das Deinstallationstool

Zu einem Installationstool bietet es sich an, auch das entsprechende Gegenstück zu implementieren. Es ist ein eigenständiges Tool, der Code basiert jedoch zu großen Teilen auf dem der Installationsroutine.

3.1 Start-Fenster

Abbildung 11: *Das Start-Fenster informiert lediglich über die Funktion des Tools*

Die Fenster des Uninstallers haben denselben prinzipiellen Aufbau wie die des Installers. Das erste Fenster (siehe Abbildung 11) dient nur als Einstieg und erfüllt selbst noch keine Aufgabe.

Abbildung 12: *Das Start-Fenster mit Fehlermeldung. Die Log-Datei konnte nicht gefunden werden.*

Beim Klick auf „Weiter>>“ wird der ActionListener aktiv. Die bei der Installation erstellte Log-Datei wird ausgelesen und die Werte (Installationspfad, Indexart, Username, Passwort, Datenbank-Name) an das nächste Fenster übergeben, welches dann auch angezeigt wird.
Sollte die Log-Datei nicht existieren, so wird wie in Abbildung 12 eine Fehlermeldung angezeigt, die Deinstallation kann dann nicht fortgesetzt werden.
Der “Abbrechen“- und der Beenden-Button der Titelleiste ruft analog zum Installationstool eine Sicherheitsabfrage auf.

3.2 Fortschrittsanzeige der Deinstallation

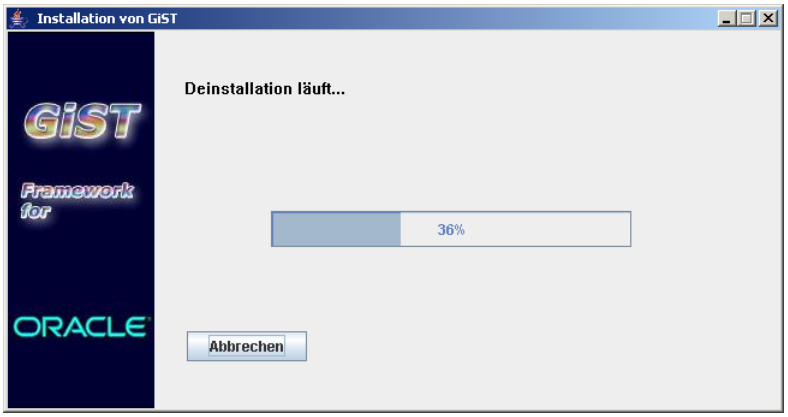

Abbildung 13: *Das Fenster mit dem Fortschrittsbalken zur Deinstallation*

Bei der Deinstallation in Abbildung 13 läuft wieder eine JProgressBar, die den Benutzer über den Fortschritt informieren soll. Folgende Schritte werden während der Deinstallation ausgeführt:

Zunächst wird der GistService durch den Konsolenbefehl `net stop GistService` beendet. Dies ist wichtig, damit die beiden aktiven Dateien (`TestService.exe`, `srvany.exe` und die erstellten Index-Dateien) zum Löschen freigegeben sind.
Anschließend werden die in der Datenbank erstellten Elemente (Index-Typen, Funktionen, Operatoren,...) durch die entsprechenden `drop`-Befehle gelöscht. Dies funktioniert wie bei der Installation über eine SQL-Datei und Java, wobei Statement für Statement eingelesen und ausgeführt wird.
Den nächsten Schritt bildet das Löschen der Dateien im Verzeichnis `c:\temp\gist_service` und im Installations-Ordner. Dies wird wieder durch Konsolenbefehle bewerkstelligt.
Analog zu 2.4.3, wird an Hand eines Templates anschließend wieder eine Registrierungsdatei erstellt, die die Einträge aus 2.4.3 wieder aus der Windows-Registry entfernt.
Zuletzt werden die nun leeren Verzeichnisse `c:\temp\gist_service` und der Installations-Ordner gelöscht. Damit ist die Deinstallation beendet.

3.3 Abschluss-Fenster

Das in Abbildung 3.3 gezeigte Abschluss-Fenster informiert nur noch über das Ende der Deinstallation. Mit dem „Beenden“-Button kann das Programm dann geschlossen werden.

Abbildung 14: *Das Abschluss-Fenster informiert über das Ende der Deinstallation.*

3.4 Sicherheitsabfrage

Abbildung 15: *Die Sicherheitsabfrage ist ein modaler Dialog, der unbeabsichtiges Abbrechen der Deinstallation vermeiden soll.*

Diese Sicherheitsabfrage funktioniert analog zu der aus der der Installation (2.6). Auch sie ist wieder ein modaler Dialog, der bei einem Klick auf „Ja“ das Programm beendet und bei „Nein“ zu dem darunterliegenden Fenster zurückkehrt.

Literatur

[Doe02] Enhancement of Oracle's Indexing Capabilities through GiST-implemented Access Methods; Mario Döller, Harald Kosch; 2002; Universität Klagenfurt

[Hel99] The GiST Indexing Project; Prof. Joe Hellerstein u.a.; 1999; `http://gist.cs.berkeley.edu`

[Kos05] Multimedia Database Systems: Where are we now?; Harald Kosch, Mario Döller; IASTED DBA 2005

[Mun04] Tipp des Monats Mai 2004; MuniQSoft GmbH; 2004; `http://www.muniqsoft.de/tipps/monatstipps/monattipp_tnsnames.htm`