



# deegree Web Catalogue Service v.2.1

## **lat/lon GmbH**

Aennchenstr. 19  
53177 Bonn  
Germany  
Tel ++49 - 228 - 184 96-0  
Fax ++49 - 228 - 184 96-29  
[info@lat-lon.de](mailto:info@lat-lon.de)  
[www.lat-lon.de](http://www.lat-lon.de)

Dept. of Geography  
Bonn University  
Meckenheimer Allee 166  
53115 Bonn

Tel. ++49 228 732098

## Change log

<b>Date</b>	<b>Description</b>	<b>Author</b>
2007-04-05	Initialisierung der deutschen Dokumentation (nur Harvester)	Markus Müller
2007-04-05	Übersetzung und Korrektur	Lyn Büsching
2007-04-10	Übersetzung und Korrektur	Hans Plum
2007-04-11	Redaktionelle Überarbeitung	Markus Müller, Lyn Büsching
2007-05-27	Fehlerbehebung in Harvesterrequest für einzelne Files	Hans Plum
2007-07-18	Abschnitt: Funktionsweise des Harvesters zugefügt Abschnitt FAQ hinzugefügt Abschnitt Testen des Harvesters um forceharvesting Methode erweitert Einige Typos bereinigt	Hans Plum
2007-07-26	Abschnitt: Synchronisation zweier Kataloge hinzugefügt	Lyn Büsching, Hans Plum

## Inhaltsverzeichnis

<b>1 Einleitung.....</b>	<b>5</b>
<b>2 Download / Installation.....</b>	<b>7</b>
<b>Voraussetzungen.....</b>	<b>7</b>
<b>deegree Web Catalogue Service release.....</b>	<b>7</b>
<b>Aufsetzen der Beispieldatenbank.....</b>	<b>7</b>
<b>Testen der Installation.....</b>	<b>8</b>
<b>3 Basiskonfiguration.....</b>	<b>15</b>
<b>Struktur der Konfigurationsdateien.....</b>	<b>15</b>
<b>4 Fortgeschrittene Konfiguration.....</b>	<b>16</b>
<b>Funktionsweise des Harvesters.....</b>	<b>16</b>
<b>Einrichten des Harvesters.....</b>	<b>16</b>
Einrichten der Datenbank.....	16
Aktivieren der Harvesting-Funktionalität.....	16
Testen des Harvesters.....	18
<b>5 Synchronisation zweier Kataloge.....</b>	<b>21</b>
<b>Installation des Triggers.....</b>	<b>21</b>
<b>Aktivierung des Triggers.....</b>	<b>21</b>
<b>Konfiguration des Triggers.....</b>	<b>21</b>
<b>Betrieb der Synchronisation.....</b>	<b>23</b>
<b>6 FAQ.....</b>	<b>24</b>

## **Tabellenverzeichnis**

Tabelle 1: Verzeichnisstruktur des CS-W.....7

## **Abbildungsverzeichnis**

Abbildung 1 Abhängigkeiten zwischen den Konfigurationsdateien.....15

Abbildung 2: Tomcat Konsole auf einem Windows System.....18

## 1 Einleitung

deegree ist ein Java Framework, das die Bausteine für Geodateninfrastrukturen (GDI) zur Verfügung stellt. Seine ganze Architektur wurde auf Basis von Standards des Open Geospatial Consortium (OGC) und des ISO Technical Committee 211 – Geographic information / Geoinformatics (ISO/TC 211) entwickelt. deegree umfasst außer OGC Web Services auch Clients. deegree ist Freie Software, wird durch die GNU Lesser General Public License (GNU LGPL) geschützt und steht über <http://www.deegree.org> zur Verfügung.

deegree2 ist das neue Release von deegree, das eine Anzahl von Funktionalitäten zur Verfügung stellt, zu denen deegree1 nicht in der Lage war. Diese Dokumentation beschreibt die Installation und Konfiguration von deegree Catalogue Service.

deegree's Web Catalogue Service Implementierung (Catalogue Service – Web profile, therefore CS-W) ist in der Lage gleichzeitig mehrere Metadatenformate auf Basis des selben physikalischen Datenspeichers zu bedienen. Dies ist möglich, da der deegree CS-W "Extensible Stylesheet Language Transformation" (XSLT) verwenden kann, um sowohl Anfragen als auch Antworten in das gewünschte Format zu transformieren. Der deegree CS-W hält die Daten nicht selbst vor, sondern gebraucht als Datenquelle einen OGC WFS (zur Zeit beschränkt auf einen deegree WFS). In Zukunft wird es möglich sein mit einem deegree CS-W jeden beliebigen OGC konformen WFS zu verwenden um Katalog Funktionen anzubieten. Der deegree WFS ist hinsichtlich der Unterstützung verschiedener Datenbankschemas sehr flexibel. Somit ist es möglich den deegree CS-W so einzurichten, das beinahe alle existierenden Metadatenformate unterstützt werden – ohne dass eine Datenreplikation nötig wird. Außer einem Catalogue Service umfasst deegree eine Anzahl anderer Dienste und Clients. Eine komplette Liste der deegree Komponenten ist einsehbar unter:

<http://www.lat-lon.de> → Produkte

Installationspakete bestimmter Komponenten können unter der folgenden URL gefunden werden:

<http://www.deegree.org> → Download

Der deegree CS-W bietet hinsichtlich der Konfiguration und der Ausgabeformate große Flexibilität. Er arbeitet mit einer großen Auswahl von physikalischen Datenquellen und Serverumgebungen. Die Konfiguration des CS-W ähnelt der anderer deegree Web Services; erforderlich ist das Editieren von XML-Dateien, die die Funktionalität des Services steuern.

Der deegree CS-W ist eine Implementierung der OpenGIS® Catalogue Services Specification 2.0 und des ISO19115/ISO19119 Application Profile for CSW 2.0 version 0.9.3 (OGC project document 04-038r2; ISO APP). Die folgenden Anfragen werden unterstützt:

- GetCapabilities
- DescribeRecord
- GetRecords
- GetRecordByID
- Transaction und
- Harvest

Der deegree CS-W erweitert die Harvest-Operation durch die Fähigkeit ganze Catalogue Services zu harvesten (ein Verhalten, das durch die ISO APP und CS-W 2.0 Spezifikation nicht festgelegt ist).

Die Web Services von deegree sind als Javamodule implementiert, die von einem zentralen Servlet (dem "dispatcher") gesteuert werden. Dieses Servlet muss in einem entsprechenden Web Server / Servlet Engine veröffentlicht werden. Die meisten der verbreiteten Web Server unterstützen Servlettechnologie, womit deegree universell einsetzbar ist. Der Einsatz der Apache Tomcat 5.5 Servlet Engine wird aufgrund seiner großen Verbreitung und seines Status als Open Source-Produkt empfohlen .

Die vorliegende Dokumentation bezieht sich ausschließlich auf die Harvesting-Funktionalität von deegree CS-W. Eine vollständige Dokumentation liegt in englischer Sprache vor.

## 2 Download / Installation

### Voraussetzungen

Zur Installation von deegree Web Catalogue Service werden benötigt:

- Java (JRE or JSDK) version 1.5.x
- Tomcat 5.5.x
- PostgreSQL 8.0 + PostGIS 1.0.x

Zur Installation dieser Komponenten sollte die entsprechende Dokumentation auf [java.sun.com](http://java.sun.com) und [tomcat.apache.org](http://tomcat.apache.org) herangezogen werden.

### deegree Web Catalogue Service release

Der deegree CS-W kann von <http://www.deegree.org> heruntergeladen werden. Dieses Release ist als WAR-Archiv gepackt. Diese Datei wird in das Verzeichnis `$TOMCAT_HOME$/webapps` gelegt und der Tomcat neugestartet. Die Installation von deegree ist damit abgeschlossen.

**Note:** Es ist ebenfalls möglich das WAR-Archiv an einem beliebigen Ort auf dem Computer zu entpacken und den Tomcat darauf zu verlinken.

Aufgrund dieser Möglichkeit wird im Folgenden das Verzeichnis, in dem die Dateien entpackt wurden, mit `$csw_home$` (`= $TOMCAT_HOME$/webapps/deegree-csw` im Normalfall) bezeichnet.

Das Verzeichnis `$csw_home$` enthält die folgende Struktur:

Verzeichnis	Inhalt
<code>./WEB-INF</code>	Erforderlich für den Tomcat, enthält alle libraries, Konfigurations- und Datendateien
<code>./WEB-INF/conf/csw</code>	WFS Konfigurationsdateien

Tabelle 1: Verzeichnisstruktur des CS-W

### Aufsetzen der Beispieldatenbank

Im Normalfall wird eine PostgreSQL -Datenbank mit UTF-8 Encoding (siehe hierzu die PostgreSQL und PostGIS Dokumentation) erzeugt. Anschließend wird das entsprechende SQL-Script ausgeführt:

`create_database_no_constraints.sql` – dabei handelt es sich um das standardmäßig verwendete Skript.

`create_database_constraints.sql` – dieses Skript bietet mehr Kontrolle über die Richtigkeit von insert-, update- und delete-Transaktionen über Fremdschlüssel-constraints

Abschließend wird der Catalogue Service noch an die Datenbank über die Datei unter `WEB-INF/conf/csw/featuretypes/*.xsd` angebunden.

Eine Beispielkonfiguration für die Verbindung von Catalogue und Datenbank:

```
<xsd:appinfo>
```

```

<deegreewfs:Prefix>app</deegreewfs:Prefix>
<deegreewfs:Backend>POSTGIS</deegreewfs:Backend>
<deegreewfs:DefaultSRS>EPSG:4326</deegreewfs:DefaultSRS>
<JDBCConnection xmlns="http://www.deegree.org/jdbc">
  <Driver>org.postgresql.Driver</Driver>
  <Url>jdbc:postgresql://localhost:5432/csw</Url>
  <User>USERNAME</User>
  <Password>PASSWORD</Password>
  <SecurityConstraints/>
  <Encoding>iso-8859-1</Encoding>
</JDBCConnection>
<deegreewfs:SuppressXLinkOutput>>true</deegreewfs:SuppressXLinkOutput>
</xsd:appinfo>

```

## Testen der Installation

Die deegree CS-W Demo ist für den Zugriff auf eine PostGIS-Datenbank bereits vorkonfiguriert. Der Datenbank-Dump (siehe oben) enthält bereits einige beispielhafte Metadatensätze um Musteranfragen zu ermöglichen. Im folgenden Abschnitt werden einige Anfragen zur Verdeutlichung des prinzipiellen Verhaltens des Services erläutert.

### request 1:

<http://my.server.domain/deegree/services?SERVICE=CSW&ACCEPTVERSION=2.0.0&REQUEST=GetCapabilities>

### result:

```

<?xml version="1.0" encoding="UTF-8"?>
<csw:Capabilities updateSequence="0" version="2.0.0"
  xmlns:csw="http://www.opengis.net/cat/csw"
  xmlns:deegree="http://www.deegree.org/csw" xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:ows="http://www.opengis.net/ows" xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <ows:ServiceIdentification>
    <ows:ServiceType>CSW</ows:ServiceType>
    <ows:ServiceTypeVersion>2.0.0</ows:ServiceTypeVersion>
    <ows:Title>lat/lon CSW</ows:Title>
    <ows:Abstract>A catalogue service that conforms to the HTTP protocol
      binding of the OpenGIS Catalogue Service specification
      version 2.0.0.</ows:Abstract>
    <ows:Keywords>
      <ows:Keyword>CSW</ows:Keyword>
      <ows:Keyword>lat/lon</ows:Keyword>
      <ows:Keyword>geospatial</ows:Keyword>
      <ows:Keyword>catalogue</ows:Keyword>
    </ows:Keywords>
    <ows:Fees>NONE</ows:Fees>
    <ows:AccessConstraints>NONE</ows:AccessConstraints>
  </ows:ServiceIdentification>
  <ows:ServiceProvider>
    <ows:ProviderName>lat/lon GmbH</ows:ProviderName>
    <ows:ProviderSite xlink:href="http://www.latlon.de" xlink:type="simple"/>
    <ows:ServiceContact>
      <ows:IndividualName>Andreas Poth</ows:IndividualName>
      <ows:PositionName>Chief Developer</ows:PositionName>
      <ows:ContactInfo>
        <ows:Phone>
          <ows:Voice>+49-228-73-9666</ows:Voice>

```



```

        <ows:Facsimile>+49-228-73-2153</ows:Facsimile>
    </ows:Phone>
    <ows:Address>
        <ows:DeliveryPoint>Meckenheimer Allee 176</ows:DeliveryPoint>
        <ows:DeliveryPoint>Raum 120</ows:DeliveryPoint>
        <ows:City>Bonn</ows:City>
        <ows:AdministrativeArea>NRW</ows:AdministrativeArea>
        <ows:PostalCode>53115</ows:PostalCode>
        <ows:Country>GERMANY</ows:Country>
        <ows:ElectronicMailAddress>poth@latlon.de</ows:ElectronicMailAddress>
    </ows:Address>
    <ows:OnlineResource
        xlink:href="mailto:poth@latlon.de" xlink:type="simple"/>
    </ows:ContactInfo>
</ows:ServiceContact>
</ows:ServiceProvider>
<ows:OperationsMetadata>
    <ows:Operation name="GetCapabilities">
        <ows:DCP>
            <ows:HTTP>
                <ows:Get
                    xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
                <ows:Post
                    xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
                </ows:HTTP>
            </ows:DCP>
        </ows:Operation>
        <ows:Operation name="DescribeRecord">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Get
                        xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
                    <ows:Post
                        xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
                    </ows:HTTP>
                </ows:DCP>
                <ows:Parameter name="typeName">
                    <ows:Value>csw:Record</ows:Value>
                </ows:Parameter>
                <ows:Parameter name="outputFormat">
                    <ows:Value>text/xml</ows:Value>
                </ows:Parameter>
                <ows:Parameter name="schemaLanguage">
                    <ows:Value>XMLSCHEMA</ows:Value>
                </ows:Parameter>
            </ows:Operation>
            <ows:Operation name="GetRecords">
                <ows:DCP>
                    <ows:HTTP>
                        <ows:Get
                            xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
                        <ows:Post
                            xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
                        </ows:HTTP>
                    </ows:DCP>
                    <ows:Parameter name="TypeName">
                        <ows:Value>csw:Record</ows:Value>
                    </ows:Parameter>
                    <ows:Parameter name="outputFormat">
                        <ows:Value>text/xml</ows:Value>
                        <ows:Value>text/html</ows:Value>
                        <ows:Value>text/plain</ows:Value>
                    </ows:Parameter>
                    <ows:Parameter name="outputSchema">
                        <ows:Value>DublinCore</ows:Value>
                    </ows:Parameter>
                </ows:Operation>
            </ows:OperationsMetadata>
        </ows:ServiceMetadata>
    </ows:Service>
</deegree:ServiceCatalogue>

```

```

</ows:Parameter>
<ows:Parameter name="resultType">
  <ows:Value>HITS</ows:Value>
  <ows:Value>RESULTS</ows:Value>
  <ows:Value>VALIDATE</ows:Value>
</ows:Parameter>
<ows:Parameter name="ElementSetName">
  <ows:Value>brief</ows:Value>
  <ows:Value>summary</ows:Value>
  <ows:Value>full</ows:Value>
</ows:Parameter>
<ows:Parameter name="CONSTRAINTLANGUAGE">
  <ows:Value>Filter</ows:Value>
</ows:Parameter>
</ows:Operation>
<ows:Operation name="GetRecordById">
  <ows:DCP>
    <ows:HTTP>
      <ows:Get
xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
      <ows:Post
xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
    </ows:HTTP>
  </ows:DCP>
  <ows:Parameter name="ElementSetName">
    <ows:Value>brief</ows:Value>
    <ows:Value>summary</ows:Value>
    <ows:Value>full</ows:Value>
  </ows:Parameter>
</ows:Operation>
<ows:Parameter name="service">
  <ows:Value>CSW</ows:Value>
</ows:Parameter>
<ows:Parameter name="version">
  <ows:Value>2.0.0</ows:Value>
</ows:Parameter>
</ows:OperationsMetadata>
<ogc:Filter_Capabilities>
  <ogc:Spatial_Capabilities>
    <ogc:Spatial_Operators>
      <ogc:BBOX/>
      <ogc:Equals/>
      <ogc:Intersect/>
      <ogc:Crosses/>
      <ogc:Contains/>
      <ogc:Overlaps/>
    </ogc:Spatial_Operators>
  </ogc:Spatial_Capabilities>
  <ogc:Scalar_Capabilities>
    <ogc:Logical_Operators/>
    <ogc:Comparison_Operators>
      <ogc:Simple_Comparisons/>
      <ogc:Like/>
      <ogc:Between/>
      <ogc:NullCheck/>
    </ogc:Comparison_Operators>
    <ogc:Arithmetic_Operators>
      <ogc:Simple_Arithmetic/>
    </ogc:Arithmetic_Operators>
  </ogc:Scalar_Capabilities>
</ogc:Filter_Capabilities>
</csw:Capabilities>

```

## request2:

http://my.server.domain/deegree/services?SERVICE=CSW&ACCEPTVERSION=2.0.0&REQUEST=GetCapabilities&Sections=OperationsMetadata

### result:

```
<?xml version="1.0" encoding="UTF-8"?>
<csw:Capabilities updateSequence="0" version="2.0.0"
  xmlns:csw="http://www.opengis.net/cat/csw"
  xmlns:deegree="http://www.deegree.org/csw" xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:ows="http://www.opengis.net/ows" xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <ows:OperationsMetadata>
    <ows:Operation name="GetCapabilities">
      <ows:DCP>
        <ows:HTTP>
          <ows:Get
            xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
          <ows:Post
            xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
          </ows:HTTP>
        </ows:DCP>
      </ows:Operation>
    <ows:Operation name="DescribeRecord">
      <ows:DCP>
        <ows:HTTP>
          <ows:Get
            xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
          <ows:Post
            xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
          </ows:HTTP>
        </ows:DCP>
        <ows:Parameter name="typeName">
          <ows:Value>csw:Record</ows:Value>
        </ows:Parameter>
        <ows:Parameter name="outputFormat">
          <ows:Value>text/xml</ows:Value>
        </ows:Parameter>
        <ows:Parameter name="schemaLanguage">
          <ows:Value>XMLSCHEMA</ows:Value>
        </ows:Parameter>
      </ows:Operation>
    <ows:Operation name="GetRecords">
      <ows:DCP>
        <ows:HTTP>
          <ows:Get
            xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
          <ows:Post
            xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
          </ows:HTTP>
        </ows:DCP>
        <ows:Parameter name="typeName">
          <ows:Value>csw:Record</ows:Value>
        </ows:Parameter>
        <ows:Parameter name="outputFormat">
          <ows:Value>text/xml</ows:Value>
          <ows:Value>text/html</ows:Value>
          <ows:Value>text/plain</ows:Value>
        </ows:Parameter>
        <ows:Parameter name="outputSchema">
          <ows:Value>DublinCore</ows:Value>
        </ows:Parameter>
        <ows:Parameter name="resultType">
          <ows:Value>HITS</ows:Value>
          <ows:Value>RESULTS</ows:Value>
          <ows:Value>VALIDATE</ows:Value>
        </ows:Parameter>
      </ows:Operation>
  </ows:OperationsMetadata>
</csw:Capabilities>
```

```

</ows:Parameter>
<ows:Parameter name="ElementSetName">
  <ows:Value>brief</ows:Value>
  <ows:Value>summary</ows:Value>
  <ows:Value>full</ows:Value>
</ows:Parameter>
<ows:Parameter name="CONSTRAINTLANGUAGE">
  <ows:Value>Filter</ows:Value>
</ows:Parameter>
</ows:Operation>
<ows:Operation name="GetRecordById">
  <ows:DCP>
    <ows:HTTP>
      <ows:Get
xlink:href="http://localhost:8081/deegree/services?" xlink:type="simple"/>
      <ows:Post
xlink:href="localhost:8081/deegree/services?" xlink:type="simple"/>
    </ows:HTTP>
  </ows:DCP>
  <ows:Parameter name="ElementSetName">
    <ows:Value>brief</ows:Value>
    <ows:Value>summary</ows:Value>
    <ows:Value>full</ows:Value>
  </ows:Parameter>
</ows:Operation>
<ows:Parameter name="service">
  <ows:Value>CSW</ows:Value>
</ows:Parameter>
<ows:Parameter name="version">
  <ows:Value>2.0.0</ows:Value>
</ows:Parameter>
</ows:OperationsMetadata>
</csw:Capabilities>

```

### request3:

http://my.server.domain/deegree/services?request=DescribeRecord&version=2.0.0&outputFormat=text/xml&schemaLanguage=XMLSCHEMA&typeName=csw:Record&namespace=csw:http://www.opengis.org/csw

### result:

```

<csw:DescribeRecordResponse version="2.0.0"
  xmlns:csw="http://www.opengis.net/cat/csw">
  <csw:SchemaComponent schemaLanguage="XMLSCHEMA"
    targetNamespace="http://www.deegree.org/csw">
    <xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
      targetNamespace="http://purl.org/dc/elements/1.1/"
      xmlns="http://purl.org/dc/elements/1.1/"
      xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:annotation>
      <xs:documentation>Simple DC XML Schema, 2002-03-12
by Pete Johnston (p.johnston@ukoln.ac.uk),
Carl Lagoze (lagoze@cs.cornell.edu), Andy Powell (a.powell@ukoln.ac.uk),
Herbert Van de Sompel (hvdsomp@yahoo.com).
This schema defines terms for Simple Dublin Core, i.e. the 15
elements from the http://purl.org/dc/elements/1.1/ namespace, with
no use of encoding schemes or element refinements.
Default content type for all elements is xs:string with xml:lang
attribute available.</xs:documentation>
    <xs:appinfo>
      <rdf:RDF xmlns:dc="http://purl.org/dc/elements/1.1/"
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        <rdf:Description rdf:about="">

```

```

<dc:title>Simple DC XML Schema</dc:title>
<dc:date>2002-03-12</dc:date>
<dc:description>This schema defines terms for Simple Dublin
Core, i.e. The 15 elements from the
http://purl.org/dc/elements/1.1/ namespace, with
no use of encoding schemes or element refinements.
Default content type for all elements is xs:string with xml:lang
attribute available.</dc:description>
<dc:publisher>The Dublin Core Metadata Initiative</dc:publisher>
<dc:creator>Pete Johnston (p.johnston@ukoln.ac.uk)</dc:creator>
<dc:creator>Carl Lagoze (lagoze@cs.cornell.edu)</dc:creator>
<dc:creator>Andy Powell (a.powell@ukoln.ac.uk)</dc:creator>
<dc:creator>
    Herbert Van de Sompel (hvdsomp@yahoo.com)
</dc:creator>
<dc:relation rdf:resource="http://purl.org/dc/elements/1.1/">
</rdf:Description>
</rdf:RDF>
</xs:appinfo>
</xs:annotation>
<xs:import namespace="http://www.w3.org/XML/1998/namespace"
    schemaLocation="http://www.w3.org/2001/03/xml.xsd"/>
<xs:element name="title" type="elementType"/>
<xs:element name="creator" type="elementType"/>
<xs:element name="subject" type="elementType"/>
<xs:element name="description" type="elementType"/>
<xs:element name="publisher" type="elementType"/>
<xs:element name="contributor" type="elementType"/>
<xs:element name="date" type="elementType"/>
<xs:element name="type" type="elementType"/>
<xs:element name="format" type="elementType"/>
<xs:element name="identifier" type="elementType"/>
<xs:element name="source" type="elementType"/>
<xs:element name="language" type="elementType"/>
<xs:element name="relation" type="elementType"/>
<xs:element name="coverage" type="elementType"/>
<xs:element name="rights" type="elementType"/>
<xs:group name="elementsGroup">
    <xs:sequence>
        <xs:choice maxOccurs="unbounded" minOccurs="0">
            <xs:element ref="title"/>
            <xs:element ref="creator"/>
            <xs:element ref="subject"/>
            <xs:element ref="description"/>
            <xs:element ref="publisher"/>
            <xs:element ref="contributor"/>
            <xs:element ref="date"/>
            <xs:element ref="type"/>
            <xs:element ref="format"/>
            <xs:element ref="identifier"/>
            <xs:element ref="source"/>
            <xs:element ref="language"/>
            <xs:element ref="relation"/>
            <xs:element ref="coverage"/>
            <xs:element ref="rights"/>
        </xs:choice>
    </xs:sequence>
</xs:group>
<xs:complexType name="elementType">
    <xs:simpleContent>
        <xs:extension base="xs:string"/>
    </xs:simpleContent>
</xs:complexType>
</xs:schema>
</csw:SchemaComponent>

```

```
</csw:DescribeRecordResponse>
```

#### request4:

```
http://my.server.domain/deegree/services?request=GetRecords&version=2.0.0&outputFormat=text/xml&outputSchema=DublinCore&namespace=csw:http://www.opengis.org/csw&typeName=Product&elementSetName=full
```

#### result:

```
<csw:GetRecordsResponse xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:csw="http://www.opengis.net/cat/csw" version="2.0.0">
  <csw:RequestId>testDoService1116584574358</csw:RequestId>
  <csw:SearchStatus status="none" timestamp="2005-05-20T12:22:55.850+02:00"/>
  <csw:SearchResults requestId="testDoService1116584574358"
    recordSchema="http://127.0.0.1:8080/deegree/services?version=2.0.0&request=DescribeRecordType&typenames=DublinCore"
    numberOfRecordsMatched="1" numberOfRecordsReturned="1" nextRecord="0">
    <dc:metadata>
      <dc:format>shape</dc:format>
      <dc:relation>latlon.shp</dc:relation>
      <dc:spatial>
        <dcmiBox:Box name="Geographic" projection="">
          <dcmiBox:northlimit units="decimal degrees">52.0</dcmiBox:northlimit>
          <dcmiBox:eastlimit units="decimal degrees">9.0</dcmiBox:eastlimit>
          <dcmiBox:southlimit units="decimal degrees">51.0</dcmiBox:southlimit>
          <dcmiBox:westlimit units="decimal degrees">7.0</dcmiBox:westlimit>
        </dcmiBox:Box>
      </dc:spatial>
      <dc:publisher>M. Schneider</dc:publisher>
      <dc:source>empty</dc:source>
      <dc:title>deegree CSW 2.0 Test</dc:title>
      <dc:creator>M. Schneider</dc:creator>
      <dc:subject>deegree CSW 2.0 Test</dc:subject>
      <dc:identifier>OWS1</dc:identifier>
      <dc:date/>
      <dc:description>First entry ever</dc:description>
      <dc:contributor>lat/lon Inc.</dc:contributor>
      <dc:language>en</dc:language>
      <dc:type>dataset</dc:type>
      <dc:rights>owner</dc:rights>
    </dc:metadata>

    [...]

  </csw:SearchResults>
</csw:GetRecordsResponse>
```

### 3 Basiskonfiguration

#### Struktur der Konfigurationsdateien

Die folgende Abbildung zeigt die Beziehungen zwischen den verschiedenen Konfigurationsdateien, die ggf. angepasst werden müssen:

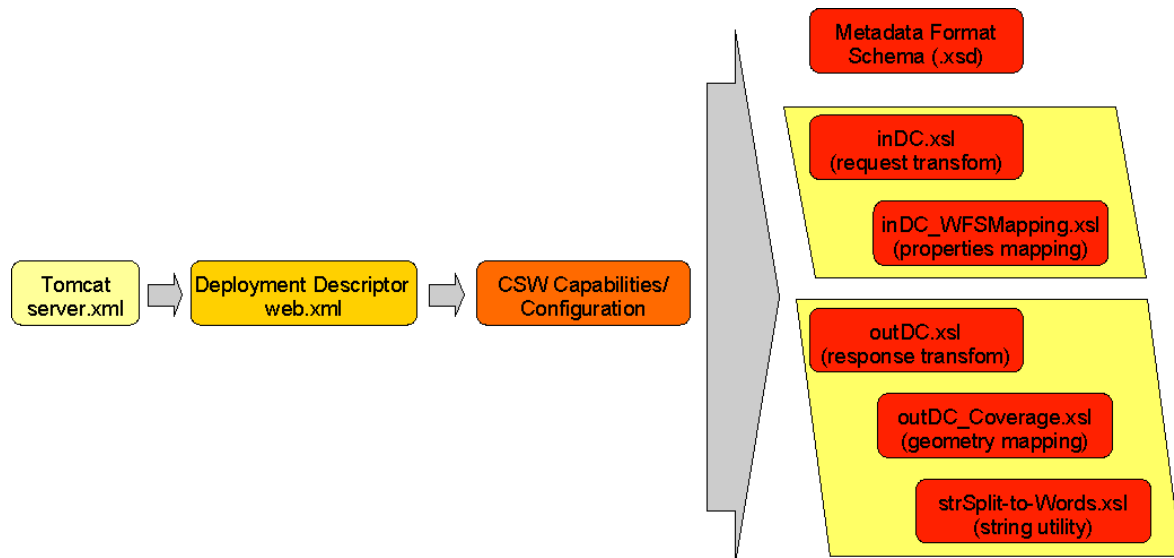


Abbildung 1 Abhängigkeiten zwischen den Konfigurationsdateien

## 4 Fortgeschrittene Konfiguration

### Funktionsweise des Harvesters

Nachdem ein Harvest-Request von einem Nutzer abgesetzt worden ist und dieser eine Bestätigung seines Auftrags, startet der Harvester einen ersten Zyklus, bei dem er die angegebene Resource via GetRecord abfragt. Diese Metadatendokumente schreibt er über einen Insert in sein eigenes Repository und speichert jeweils den Zeitstempel (Timestamp) und Fileidentifizier aus dem jeweiligen Metadatenatz ab.

Im nächsten Zyklus muß sich Harvester gegen die entsprechende Resource synchronisieren. Hierzu prüft er, ob der Datensatz mit dem Fileidentifizier noch vorhanden ist bzw. ob dieser geändert worden ist. Falls der Datensatz in der Quelle gelöscht worden ist, entfernt der Harvester diesen auch in seinem Repository; hat sich der Timestamp geändert, führt der Harvester ein Update gegen seinen eigenen Bestand aus.

### Einrichten des Harvesters

Der deegree CS-W unterstützt die harvest-Operation, wie es in der ISO APP definiert ist, zusätzlich ist das Harvesten ganzer Kataloge möglich. Die Installation und Konfiguration dieser Fähigkeit ist im Folgenden beschrieben.

### Einrichten der Datenbank

Bevor der Harvester aktiviert werden kann, müssen einige zusätzliche Tabellen zur Datenbank des CSW hinzugefügt werden. Dazu muß die Datei harvester.SQL in der CSW-Datenbank ausgeführt werden. Dadurch werden vier Tabellen erstellt, die zum Verwalten der Harvesting-Jobs und der Kontrolle der einzelnen Metadaten-Records des zugehörigen CSW verwendet werden können.

Danach muss der Harvester so konfiguriert werden, dass er auf diese Tabelle zugreift. Deswegen muss das entsprechende properties-file, das in deegree2.jar enthalten ist angepasst werden.

1. Entpacken das WEB-INF/lib/deegree2.jar in einem temporären Verzeichnis
2. Erstellen eines Verzeichnis mit dem Namen `$csw_home$/WEB-INF/classes/org/deegree/ogcwebservice/csw/manager`
3. Kopieren der Datei `org/deegree/ogcwebservice/csw/manager/harvestrepository.properties` (aus der entpackten jar-Datei) nach `$csw_home$/WEB-INF/classes/org/deegree/ogcwebservice/csw/manager` und hier Anpassen der Datenbankverbindung.

### Aktivieren der Harvesting-Funktionalität

Um die Harvesting Funktion des CSW zu aktivieren, muss die Datei WEB-INF/web\_harvester.xml verwendet werden. Hierzu wird sie in die Datei web.xml umbenannt (Achtung: Zuvor sollte eine Sicherungskopie der



ursprünglichen web.xml Datei erstellt werden!).

Folgender Eintrag unterscheidet eine Catalogue mit bzw. ohne Harvester Funktionalität in der Datei WEB-INF/web.xml:

```
<web-app>
  <context-param>
    <param-name>CSW.config</param-name>
    <param-value>
      /WEB-INF/conf/csw/csw_capabilities.xml
    </param-value>
  </context-param>
  <listener>
    <listener-class>
      org.deegree.enterprise.servlet.CSWHarvestingContextListener
    </listener-class>
  </listener>
  ...
```

Diese Parameter erzwingen für die Webapplikation des Catalogue einen besonderen Listener zu verwenden, der während der Initialisierung und des Herunterfahrens des Katalogservices aufgerufen wird.

Da der Harvester den Anbieter der Harvesting-Operation via eMail informiert, muss die SMTP Verbindung des Katalogservices konfiguriert werden. Dafür muss im Tomcat die folgende Eigenschaft gesetzt werden: "-DmailHost=SERVERNAME". Das Konfigurieren dieser Eigenschaft ist abhängig von der Installation des Tomcats und der Art wie er gestartet wird

**a) Tomcat ist als Service unter Windows installiert/gestartet:** Öffnen Sie die Apache Tomcat Konfigurations-Konsole und fügen Sie die mailHost Parameter zu den Java Optionen hinzu:

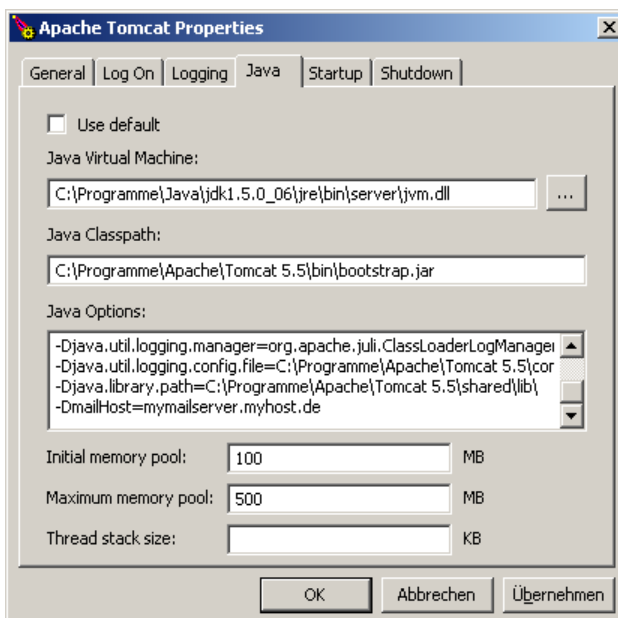


Abbildung 2: Tomcat Konsole auf einem Windows System

**b) Tomcat ist als Service unter Linux/Unix installiert/gestartet:** Setzen der JAVA\_OPTS in \$TOMCAT\_HOME/bin/catalina.sh

```
export JAVA_OPTS="-Xmx500m -Xms100m -DmailHost=SERVERNAME"
```

**c) Tomcat wird direkt gestartet (z.B. durch eine Batch-Datei):** Setze die Option im Java-Aufruf

```
java.exe -Xms100m -Xmx500m -jar -DmailHost=mymailhost.myhost.de
-Duser.dir="C:\Programme\Apache\Tomcat 5.5"
"C:\Programme\Apache\Tomcat 5.5\bin\bootstrap.jar" start
```

## Testen des Harvesters

### Senden einer Harvesting-Anfrage – Hinzufügen eines Harvesting-Jobs

Der folgende Request zeigt eine Anfrage, die das Harvesten eines ganzen Kataloges startet. Diese Funktion ist deegree-spezifisch und nicht durch die ISO APP definiert (nur das Harvesten von einzelnen Metadaten wird hier spezifiziert).

```
http://my.server.domain/deegree/services?request=Harvest&version=2.0.0&source=http://HOSTNAME:PORTNUMBER/aCatalogue/services&resourcetype=catalogue&resourceformat=text/xml&responsehandler=mailto:info@myemail.de&harvestinterval=PT2M
```

Wichtige Parameter:	Beschreibung
request=Harvest	
version=2.0.0&	
source=http://HOSTNAME:PORTNUMBER/aCatalogue/services&	Adresse des CSW der geharvestet wird
resourcetype=catalogue&	Dies ist ein vendor-specific Parameter: Da die CSW Spezifikation die Möglichkeit einen ganzen CSW zu harvesten nicht vorsieht, verwendet deegree hier diesen Parameter mit dem Wert "catalogue".
&resourceformat=text/xml&	
responsehandler=mailto:info@myemail.de&	Die eMail-Adresse zu der alle Benachrichtigungen für diese Anfrage gesendet werden.
interval=PT2M	Zeitintervall, übereinstimmend, mit der ISO8601 Period syntax.
<b>Siehe OGC CSW 2.0 Spezifikation für weitere Informationen.</b>	

*Tip:* Um im Rahmen eines Tests einen neuen Harvesting-Zyklus zu erzwingen, kann man in der Tabelle harvestsource im Feld forceharvesting für die jeweilige Adresse den Wert auf "1" setzen. Beim nächsten Lesezugriff des Harvesters auf die DB - im Regelfall innerhalb von 10 Sekunden – wird ein neuer Zyklus direkt gestartet. Der Harvester stoppt

den weitere Zyklen des Harvesting, wenn der Wert wieder auf "0" zurückgesetzt wird.

```
Bsp: update harvestsource set forceharvesting = 1 where id =  
ID_OF_HARVESTSOURCE
```

### *Löschen eines Harvesting-Jobs*

Die OGC CSW 2.0 Spezifikation definiert keine Operation zum Löschen eines Harvesting-Jobs. Es sind zwei Schritte notwendig um einen Harvesting-job zu löschen:

1. Lösche den Job aus den in Beziehung stehenden Harvester-Tabellen harvestsource, metadatabcache, jt\_source\_responsehandler, reponsehandler in einer Anfrage

Der folgende SQL Anweisung muss Schritt-für-Schritt auf die Harvester-Datenbank ausgeführt werden:

--execute alles in einer Transaktion

```
begin;
```

-- Finde die ID der gewünschten harvestsource

```
select id from harvestsource where source =  
'http://HOSTNAME:PORTNUMMBER/aCatalogue/services';
```

-- Finde alle in Beziehung stehenden ID's in responsehandler, die zu der ID der Quelle gehören

```
select fk_responsehandler from jt_source_responsehandler  
where fk_harvestsource = ID OF DESIRED HARVESTSOURCE;  
delete from jt_source_responsehandler where  
fk_harvestsource = SOURCE-ID;  
delete from responsehandler where id = <LIST OF  
fk_harvestsource>;
```

-- Finde alle Einträge mit ihren fileidentifiers im metadatabcache zugehörig zur harvestsource

```
select fileidentifier from metadatabcache where id = ID of  
harvestsource;  
delete from metadatabcache where id = ID of harvestsource;  
commit;
```

2. Über die Liste der fileidentifiers werden im Anschluss alle Datensätze aus der CSW Datenhaltung gelöscht. Über CSW Transaktionen auf der Basis des HTTP POST Protokoll werden untenstehende Anfragen gegen den Catalogue gesendet:

```
<?xml-stylesheet type="text/xsl"?>
<csw:Transaction service="CSW" version="2.0.0"
xmlns:csw="http://www.opengis.net/cat/csw"
xmlns:gml="http://www.opengis.net/gml"
xmlns:iso19115="http://schemas.opengis.net/iso19115full"
xmlns:iso19115brief="http://schemas.opengis.net/iso19115brief"
xmlns:smXML="http://metadata.dgiwg.org/smXML"
xmlns:ogc="http://www.opengis.net/ogc">
  <csw>Delete>
    <csw:Constraint version="1.0.0">
      <ogc:Filter>
        <ogc:PropertyIsLike wildCard="%" singleChar="_" escape="\ ">
          <ogc:PropertyName>./iso19115:fileIdentifier/
            smXML:CharacterString</ogc:PropertyName>
          <ogc:Literal>FILEIDENTIFIERS</ogc:Literal>
        </ogc:PropertyIsLike>
      </ogc:Filter>
    </csw:Constraint>
  </csw>Delete>
```

### *Aktualisierung der harvestsource Parameter*

Wenn die harvestsource bearbeitet werden soll müssen zunächst, wie zuvor beschrieben, alle zugehörigen Metadateneinträge gelöscht werden. Anschließend erfolgt die Bearbeitung wie folgt:

```
update harvestsource set source = 'NEW CATALOGUE URL' where id = 'ID OF THE
SOURCE TO BE CHANGED';
```

### *Aktualisierung der eMail-Adresse einer harvestsource*

#### SQL: Schritt-für-Schritt auszuführen

```
-- finde die dazugehörigen Fremdschlüssel für die jeweilige harvestsource in der
Verbindungstabelle
select fk_responsehandler from jt_source_responsehandler where fk_harvestresource =
ID;
update reponsehandler set address = 'NEW EMAILADDRESS' where id = fk_reponsehandler
and isEmailAddress = 1;
```

### *Harvesting eines Metadatenrecords aus einer Datei:*

Senden Sie einen harvestrequest mit der URL einer Datei an den Service:  
<http://my.server.domain/deegree/services?request=Harvest&version=2.0.0&source=http://HOSTNAME:PORTNUMMBER/aDirectory/aMetadataFile.xml&resourceType=csw:profile&resourceformat=text/xml&responsehandler=mailto:info@myemail.de&harvestinterval=PT2M>

Hierbei ist zu beachten, dass diese Datei einmalig geharvested wird und im Metadaten-Repository verbleibt, bis sie manuell durch eine CSW Transaktion gelöscht wird.



## 5 Synchronisation zweier Kataloge

Um eine hohe Verfügbarkeit eines Katalogdienstes zu gewährleisten, kann im Bedarfsfall auf ein 2. unabhängiges System zurückgegriffen werden. Um dies zu realisieren, soll jede schreibende Transaktion auf dem Hauptsystems (Master) auch gegen das Ausfallsystem (Slave) abgesetzt werden.

Die Umsetzung erfolgt hier über einen sogenannten Triggermechanismus. Die Konfiguration und Rahmenbedingungen, die zu beachten, werden im Folgenden beschrieben:

### Installation des Triggers

deegree CSW enthält in der Webapplikation die Funktionalität des Triggers. Die Installation erfolgt wie oben beschrieben. Weiterhin muß jedoch eine weitere Tabelle in die Datenbank des Mastersystems angelegt werden. Hierzu ist das SQL Skript FAILEDREQUESTS.sql gegen diese Datenbank auszuführen.

### Aktivierung des Triggers

Zur Konfiguration des Triggermechanismus muss eine XML-Datei namens triggerConfiguration.xml im Klassenverzeichnis des CSW, /WEB-INF/classes bereitgestellt werden (Auszug aus einer Beispieldatei siehe unten).

### Konfiguration des Triggers

Innerhalb dieser Datei können Trigger für unterschiedliche Services definiert werden. Relevant ist der Eintrag für den CSW. Folgende Informationen werden für die Konfiguration benötigt:

- CSW Adresse des Ausfallsystems (Parameter targetCSW1)
- Verbindungsdaten zur Datenbank, in der die Master-Slave Kommunikation verwaltet wird. Falls das Slave-System nicht erreicht werden kann, werden die Transaktionen in einer DB im Mastersystem zwischengespeichert und es wird zu einem späteren Zeitpunkt noch einmal versucht, das Ausfallsystem zu synchronisieren.
  - JDBC-Treiber der Datenbank (Parameter driver), bspw. org.postgresql.Driver. Bisher wird hier nur PostgreSQL/PostGIS unterstützt.
  - Verbindungsparameter zur Datenbank (Parameter url) wie: URL, Port und Datenbankname
  - Benutzername (Parameter user)
  - Passwort (Parameter password)
- Um den Administrator jeweils zu einer Störung zu informieren, versendet das Mastersystem eMails. Hierzu benötigt das Mastersystem Zugriff auf einen SMTP-Server (Simple Mail Transfer Protocol). Hierzu werden folgende Angaben benötigt:

- Adresse des SMTP-Servers (Parameter smtp)
- Absender- und Empfängeradresse (Parameter sender und receiver)
- Über den Parameter maxRepeat wird angegeben, nach wie vielen fehlgeschlagenen Versuchen, die Synchronisation für den jeweiligen Datensatz abgebrochen werden soll.

Beispielkonfiguration einer triggerConfiguration.xml Datei:

```
<dgTrigger:name>org.deegree.ogcwebservices.csw.CatalogueService</dgTrigger:name>
  <dgTrigger:method>
    <dgTrigger:name>doService</dgTrigger:name>
    <dgTrigger:preTrigger>
      <!-- add trigger definition(s) if required -->
    </dgTrigger:preTrigger>
    <dgTrigger:postTrigger>
      <dgTrigger:trigger>
        <dgTrigger:name>FormattingTrigger</dgTrigger:name>
      </dgTrigger:trigger>
    </dgTrigger:postTrigger>
  </dgTrigger:method>
<dgTrigger:performingClass>org.deegree.ogcwebservices.csw.manager.CSWsynchronization
Trigger</dgTrigger:performingClass>
  <dgTrigger:initParam>
    <dgTrigger:name>driver</dgTrigger:name>
    <dgTrigger:type>java.lang.String</dgTrigger:type>
    <dgTrigger:value>org.postgresql.Driver</dgTrigger:value>
  </dgTrigger:initParam>
  <dgTrigger:initParam>
    <dgTrigger:name>url</dgTrigger:name>
    <dgTrigger:type>java.lang.String</dgTrigger:type>
    <dgTrigger:value>jdbc:postgresql://localhost:5432/csw_lgb</dgTrigger:value>
  </dgTrigger:initParam>
  <dgTrigger:initParam>
    <dgTrigger:name>user</dgTrigger:name>
    <dgTrigger:type>java.lang.String</dgTrigger:type>
    <dgTrigger:value>postgres</dgTrigger:value>
  </dgTrigger:initParam>
  <dgTrigger:initParam>
    <dgTrigger:name>password</dgTrigger:name>
    <dgTrigger:type>java.lang.String</dgTrigger:type>
    <dgTrigger:value>postgres</dgTrigger:value>
  </dgTrigger:initParam>
  <dgTrigger:initParam>
    <dgTrigger:name>smtp</dgTrigger:name>
    <dgTrigger:type>java.lang.String</dgTrigger:type>
    <dgTrigger:value>mail.lat-lon.de</dgTrigger:value>
  </dgTrigger:initParam>
  <dgTrigger:initParam>
    <dgTrigger:name>sender</dgTrigger:name>
    <dgTrigger:type>java.lang.String</dgTrigger:type>
    <dgTrigger:value>buesching@lat-lon.de</dgTrigger:value>
  </dgTrigger:initParam>
  <dgTrigger:initParam>
    <dgTrigger:name>receiver</dgTrigger:name>
    <dgTrigger:type>java.lang.String</dgTrigger:type>
    <dgTrigger:value>buesching@lat-lon.de</dgTrigger:value>
  </dgTrigger:initParam>
  <dgTrigger:initParam>
    <dgTrigger:name>maxRepeat</dgTrigger:name>
    <dgTrigger:type>java.lang.Integer</dgTrigger:type>
    <dgTrigger:value>3</dgTrigger:value>
  </dgTrigger:initParam>
  <dgTrigger:initParam>
    <dgTrigger:name>targetCSW1</dgTrigger:name>
  </dgTrigger:initParam>
</dgTrigger>
```

```
<dgTrigger:type>java.net.URL</dgTrigger:type>
<dgTrigger:value>http://localhost:8080/deegree/services</dgTrigger:value>
  </dgTrigger:initParam>
</dgTrigger:trigger>
</dgTrigger:postTrigger>
</dgTrigger:method>
</dgTrigger:class>
```

## Betrieb der Synchronisation

Beim Betrieb der Synchronisation ist zu beachten, dass das Mastersystem im Falle eines Ausfalls des Slavesystems einer erhöhten Belastung ausgesetzt wird. Da jeder Request in diesem Falle zwischengespeichert und bis zu n-mal (im Parameter maxRepeat zu konfigurieren) versucht wird, diesen gegen das Slavesystem abzusetzen, muß neben den normalen Funktionen diese Mehrbelastung abgearbeitet werden. Dies kann vorübergehend zu einem verzögerten Betrieb führen.



## 6 FAQ

Beim Vergleich des Repositories des Harvesters und der Quelle fehlen Datensätze im Harvester. Was kann die Ursache sein?

- Der Harvester kann nur auf validen XML Dokumenten gegen das ISO Applikation Profile arbeiten.
- Die Fileidentifizierer müssen über alle zu harvestenden Datensätze eindeutig sein. Doppelte Fileidentifizierer werden abgewiesen.
- Falls der Harvester einen Datensatz über seinen Fileidentifizierer nicht erhält, löscht er diesen in seinem eigenen Repository. Bricht während eines Harvestingzykluses die Verbindung ab, ist für den Harvester nicht erkennbar, warum er den angefragten Datensatz nicht erhalten hat. In dieser Folge löscht er diesen auch aus seinem Repository. Bei sehr langsam antwortenden Cataloguen kann der Timeout für jeden GetRecord-Request in der Datei harvesterrepository.properties (in Millisekunden) höher gesetzt werden, um dieses Problem zu umgehen. Hierbei ist jedoch zu beachten, dass der Harvester ggf. langsamer arbeiten wird.