FirstSpirit™
Your Content Integration Platform

FirstSpirit™ Search

FirstSpirit Version 4.x

Version	**1.3**
Status	**RELEASED**
Datum	**2011-06-24**
Abteilung	FS-Core
Autor/ Autoren	R. Voß, S. Rusch
Copyright	2011 e-Spirit AG
Dateiname	SEAR40DE_FirstSpirit_Modules_Search

e-Spirit AG

Barcelonaweg 14
44269 Dortmund | Germany

T +49 231 . 286 61-30
F +49 231 . 286 61-59

info@e-Spirit.com
www.e-Spirit.com

e-Spirit AG

Inhaltsverzeichnis

1 Einleitung

Die Dokumentation zu FirstSpirit™ Search beschreibt das lizenzabhängige FirstSpirit™ Modul zur Anbindung unterschiedlicher Suchmaschinen.

Dabei sind unterschiedliche Anwendungsmöglichkeiten vorstellbar:

- FirstSpirit™ Search bindet eine externe Suchmaschine oder eine andere Datenquelle (z. B. Datenbank) an die von FirstSpirit™ verwalteten Webanwendungen an (Vorschau, Generierung).
- FirstSpirit™ Search stellt eine interne Suchmaschine zur Verfügung.
- FirstSpirit™ Search kann bestehende Suchmaschinen um zusätzliche Funktionalitäten erweitern. Beispiele hierfür sind Mehrsprachigkeit, frei definierbare Kategorien und dynamische Filter.
- FirstSpirit™ Search kann in Verbindung mit FirstSpirit™ Personalisation Suchergebnisse für bestimmte Benutzer und Gruppen filtern. Es werden nur die Suchergebnisse ausgegeben, für die der jeweilige Benutzer (bzw. Gruppe) die entsprechenden Rechte besitzt.

Der Inhalt einer Web-Site kann relativ schnell wachsen. Damit der Benutzer schnell zu den gewünschten Informationen gelangt, ist der Einsatz von Suchmaschinen notwendig.

Ein Nachteil vieler verfügbarer Suchmaschinen ist die mangelnde Erweiterungsfähigkeit. Vielfach ist beispielsweise die Indizierung unbekannter Dokumentenformate, die uneingeschränkte Definitionsfähigkeit von Attributen (so genannte Metainformation) oder die Anbindung einer externen Datenquelle gewünscht. Diese Forderungen werden von vielen Suchmaschinen nur unvollständig bzw. unzureichend erfüllt.

Mit dem Modul FirstSpirit™ Search ist es möglich, wahlweise interne oder externe Suchmaschinen an FirstSpirit™ anzubinden. Über das Modul FirstSpirit™ Search kann eine bestehende Suchmaschine außerdem um weitere Funktionalitäten erweitert werden.

2 Bausteinprinzip

Das Modul FirstSpirit Search ist in Bausteinen untergliedert. Abhängig vom gewünschten Einsatzzweck können diese Bausteine miteinander kombiniert werden.

Grundsätzlich verfügt FirstSpirit Search über drei Bausteine:

- Engine (siehe Kapitel 2.1, Seite 5)
- Server (siehe Kapitel 2.2, Seite 8)
- Service (siehe Kapitel 2.3, Seite 8)

In den folgenden Unterkapiteln werden die einzelnen Bausteine und die dazugehörigen Begriffe aufgeführt und beschrieben.

2.1 Engine

FirstSpirit Search bietet die Möglichkeit eine interne oder eine externe Suchmaschine anzubinden. Um einer Webanwendung Suchfunktionen zur Verfügung zu stellen, muss eine so genannte Such-Engine verwendet werden. Unter dem Begriff (Such-)Engine wird eine Suchmaschine verstanden, die Standardfunktionen für das Durchsuchen von Dokumenten oder Datensätzen bereitstellt. Außerdem ist es möglich, über eine Such-Engine Informationen über die dahinter stehende Datenquelle zu erhalten.

2.1.1 Implementierung / Adapter

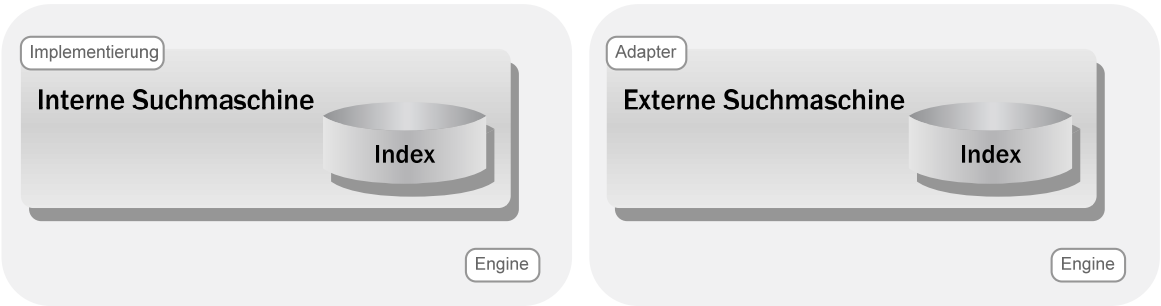

Abbildung 2-1: Engine-Implementierung und -Adapter

Die Anbindung einer internen Suchmaschine wird als Implementierung bezeichnet. Bei der Implementierung einer Schnittstelle (hier: die Suchmaschine) werden Funktionen selbstständig zur Verfügung gestellt, ohne dabei von anderen Komponenten abhängig zu sein.

Momentan steht als Implementierung die Spider-Engine (siehe Kapitel 4.5.4.2 Seite 37) zur Verfügung.

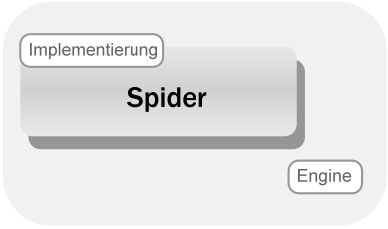

Abbildung 2-2: Engine-Implementierungen

Im Gegensatz dazu, wird die Anbindung einer externen Suchmaschine als Adapter bezeichnet. Ein Adapter ermöglicht es auf inkompatible Schnittstellen zuzugreifen.

Als Engine-Adapter stehen momentan Lucene (siehe Kapitel 4.5.4.3, Seite 46), und JDBC (siehe Kapitel 4.5.4.4, Seite 48) zur Verfügung.

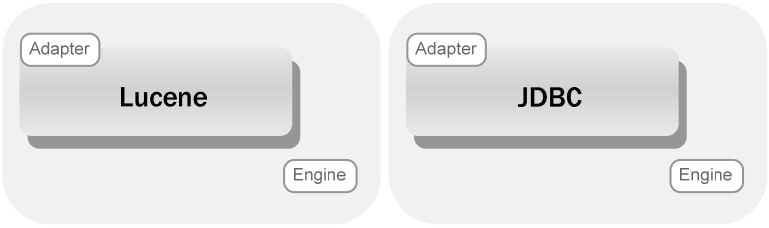

Abbildung 2-3: Engine-Adapter

2.1.2 Proxy

Eine weitere Funktion, die der Baustein Engine erfüllt, ist die Bündelung von Suchmaschinen. Außerdem können Suchmaschinen um fehlende Funktionalitäten erweitert werden, z. B. Kategorisierung und Sprachzuweisung.

Ein Proxy delegiert Aufrufe an einzelnen Engines. Zusätzlich kann er auf die Aufrufparameter und Rückgabewerte Einfluss nehmen. In FirstSpirit Search wird ein Proxy verwendet, um Engines zu bündeln und um weitere Funktionen zu erweitern. Ein Beispiel für die Erweiterung der Suchmaschinenfunktionen ist die Definition eines Filters, um die Ergebnismenge einzuschränken.

Beim Zusammenschluss von Engines zu einem Verbund ist es unerheblich, ob es sich bei den einzelnen gebündelten Engines um eine Implementierung (siehe Kapitel 2.1.1, Seite 5), einen Adapter (siehe Kapitel 2.1.1, Seite 5) oder um einen weiteren Proxy handelt.

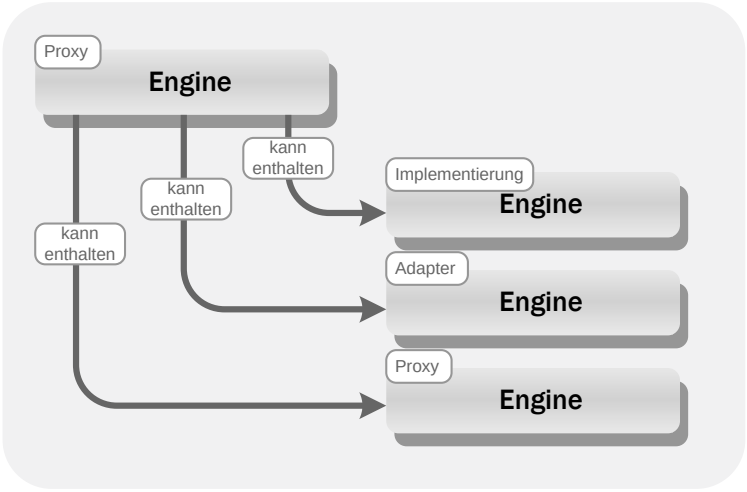

Abbildung 2-4: Engine-Proxy

Momentan sind folgende Engine-Proxys verfügbar (siehe Abbildung 2-5):

- einfacher Filter (Kapitel 4.5.4.5, Seite 51)
- Mehrsprachigkeit (Filter) (Kapitel 4.5.4.6, Seite 53)
- Kategorisierung (Filter) (Kapitel 4.5.4.7, Seite 53)
- einfacher Engine-Verbund (Kapitel 4.5.4.8, Seite 57)
- Mehrsprachigkeit (Engine-Verbund) (Kapitel 4.5.4.9, Seite 59)
- Kategorisierung (Engine-Verbund) (Kapitel 4.5.4.10, Seite 61)
- Personalisierung (Kapitel 4.5.4.11, Seite 63)
- Session-Management (Kapitel 4.5.4.12, Seite 66)

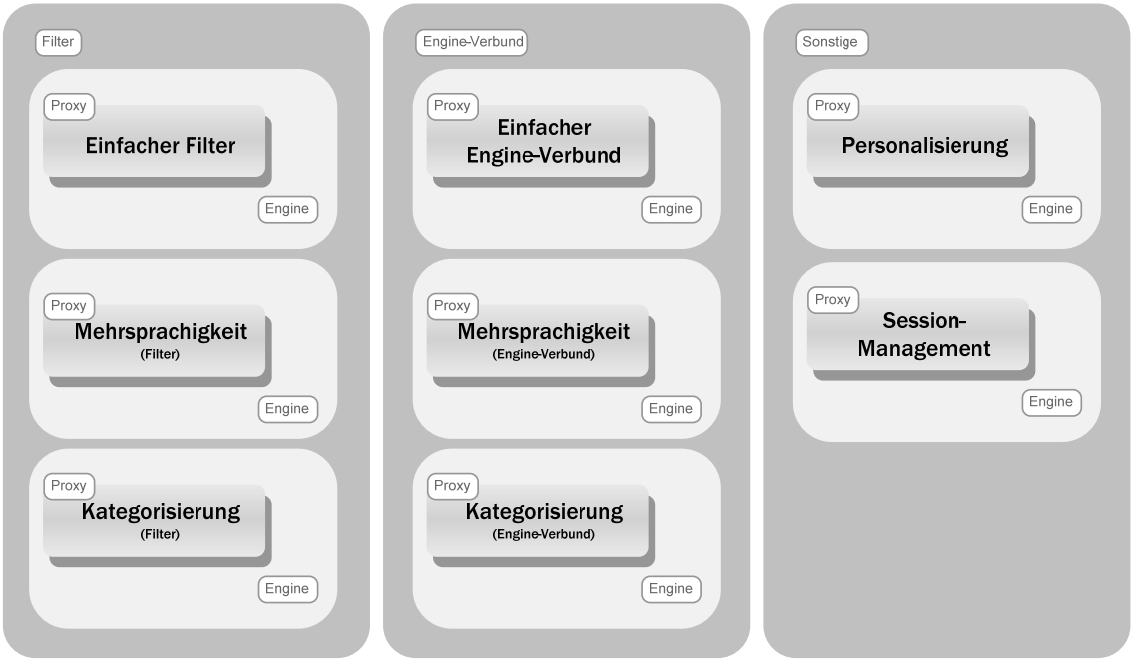

Abbildung 2-5: Engine-Proxys

2.2 Server

Ein weiterer Baustein des Moduls FirstSpirit Search ist der (Such-)Server. Über einen (Such-)Server werden alle definierten Engines (siehe Kapitel 2.1, Seite 5) verwaltet. Der Server dient als Einstiegspunkt in das Suchsystem und wird bei der Konfiguration der Web-Komponente "FS SEARCH" angegeben (siehe Kapitel 3.3 Seite 12).

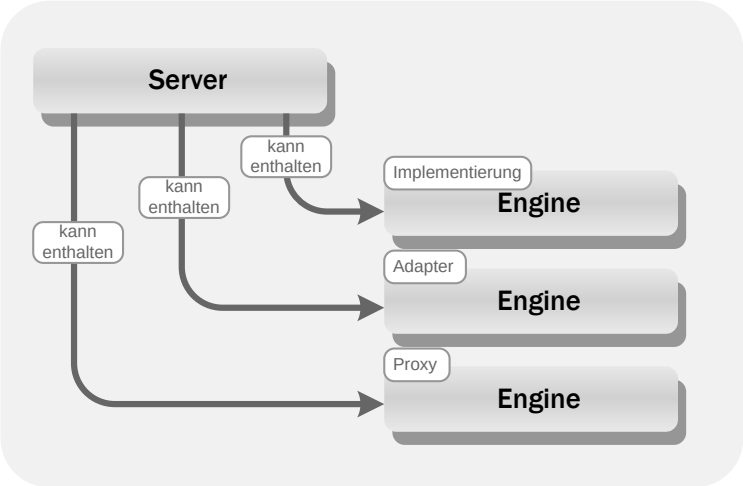

Abbildung 2-6: Server

2.3 Service

Ein Service übernimmt administrative Funktionen. Über einen Service können Dienste zur Verfügung gestellt werden, die je nach Bedarf gestartet oder gestoppt werden können.

Mögliche Dienste sind:

- Steuerung eines Such-Servers
  (siehe Kapitel 2.2 Seite 8 und Kapitel 4.5.1.1 Seite 20)
- Aktivieren der Protokollierung ("Logging")
  (siehe Kapitel 4.5.1.2, Seite 23)
- zeitgesteuerter Neuaufbau des Suchindexes (Re-Indizierung)
  (siehe Kapitel 4.5.1.3, Seite 21)

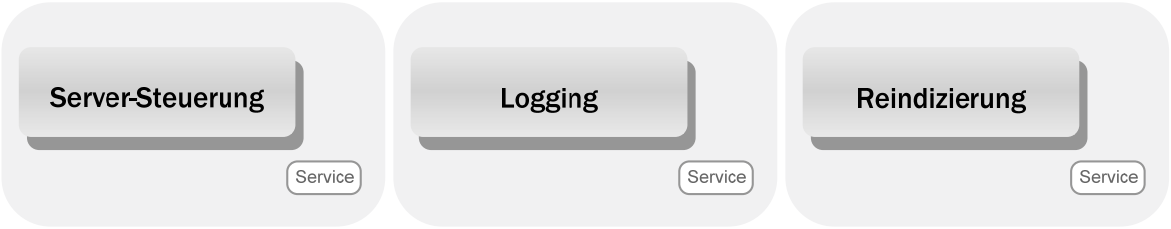

Abbildung 2-7: Services

Mehrere Services können über einen Proxy gebündelt werden:

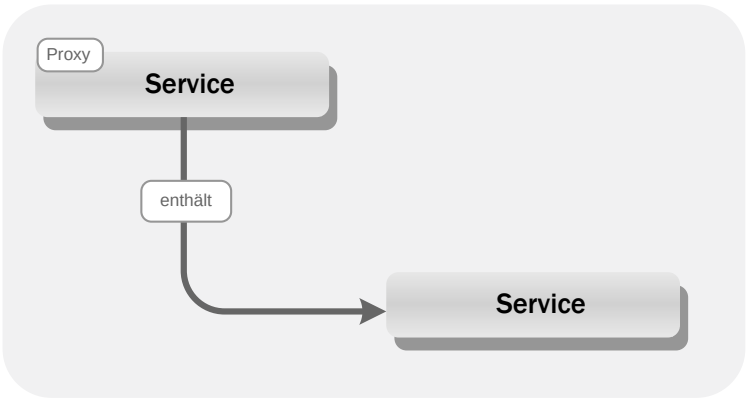

Abbildung 2-8: Service-Proxy

Oberhalb eines Service-Bausteines gibt es keinen anderen Baustein. Somit kann nur ein Service-Proxy oder ein Service (mehrere Services müssen über einen Proxy gebündelt werden) das oberste Element bei der Betrachtung sein.

3 Konfiguration

Die Konfiguration von FirstSpirit Search erfolgt in drei Schritten:

- Installation des Moduls (siehe Kapitel 3.1, Seite 10)
- Installation der Web-Komponente (siehe Kapitel 3.2, Seite 11)
- Konfiguration der Web-Komponente (siehe Kapitel 3.3, Seite 12)

In den nachfolgenden Kapiteln werden die drei benötigten Schritte beschrieben.

3.1 Installieren des Moduls auf dem Server

Das Modul FirstSpirit Search muss zunächst innerhalb der Anwendung zur Server- und Projektkonfiguration installiert werden. Im Bereich Servereigenschaften wird dazu der Menüeintrag "Module" selektiert. Mit einem Klick auf den Button "Installieren" öffnet sich ein Dateiauswahldialog. Hier kann die zu installierende fsm-Datei ausgewählt werden. Die erfolgreich installierte Datei wird anschließend im Dialog "Server-Eigenschaften" angezeigt:

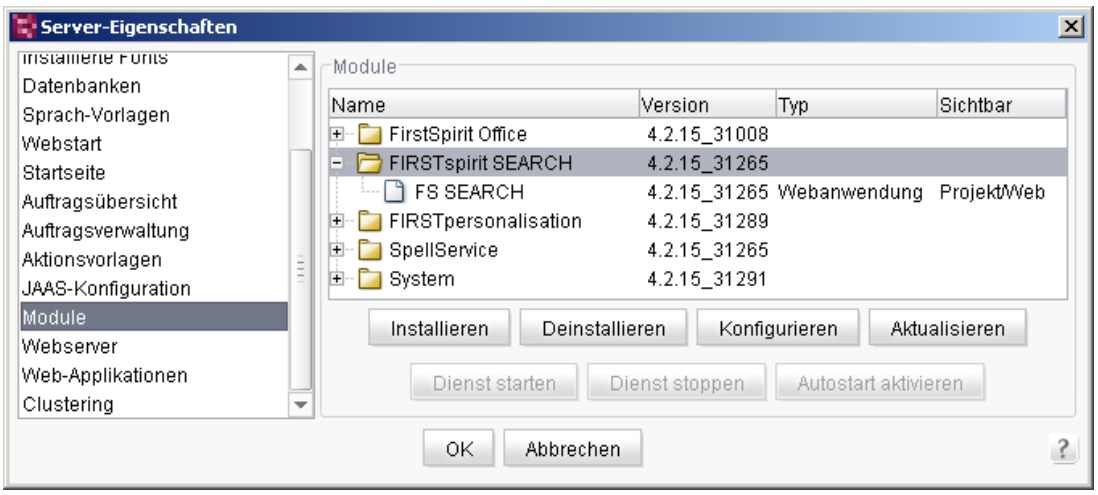

Abbildung 3-1: Installation des Moduls auf dem FirstSpirit-Server

Bestandteil des Moduls FirstSpirit Search ist die Webanwendung "FS SEARCH". Die Webanwendung stellt JSP-Tags und Servlets zur Verfügung, die innerhalb des Projekts verwendet und aufgerufen werden können.

Die Komponente ist "Sichtbar" für die Bereiche "Projekt/Web". Damit handelt es sich um eine "web-lokale" Komponente. Diese kann nach der Installation den unterschiedlichen Web-Bereichen (preview, staging, live, webedit) innerhalb der

gewünschten Projekte zugefügt werden (siehe Kapitel 3.2, Seite 11).

Weitere Informationen zu diesem Dialog siehe "FirstSpirit Handbuch für Administratoren".

3.2 Installieren der Webanwendung im Projekt

Die Webanwendung muss nun im gewünschten Projekt installiert werden. Dazu wird innerhalb der Projekteigenschaften der Menüeintrag "Web-Komponenten" aufgerufen. In diesem Bereich können die Web-Komponenten für ein Projekt aktiviert werden.

Abbildung 3-2: Installation der Webanwendung innerhalb der Web-Bereiche

Es existieren für jedes Projekt drei unterschiedliche Web-Bereiche. Über die jeweilige Registerkarte können die Web-Komponenten für jeden Bereich einzeln aktiviert und konfiguriert werden:

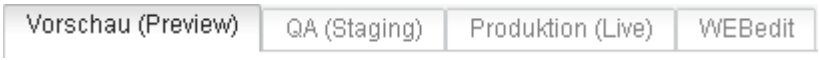

Abbildung 3-3: Web-Bereiche innerhalb eines Projekts

- Vorschau (Preview): Ort für die Projektinhalte für die eine Vorschau angefordert wurde.
- QA (Staging): Ort für die generierten Projektinhalte
- Produktion (Live): Ort für die veröffentlichten Projektinhalte
- WEBedit: Ort für die Projektinhalte von WebEdit

Hinzufügen: Mit einem Klick auf den Button öffnet sich der Dialog "Hinzufügen". In der Liste werden alle Web-Komponenten angezeigt, die auf dem Server installiert sind (siehe Kapitel 3.1, Seite 10).

Nach dem Hinzufügen zu einem Webbereich, besteht die Möglichkeit, die Komponenten zu konfigurieren, entweder mit einer von der Komponente erzeugten

oder einer generischen GUI (siehe 3.3 Seite 12). Nach der Konfiguration müssen die Komponenten noch aktiviert werden. Dabei kann eine Komponente innerhalb eines Projekts nur für bestimmte Bereiche aktiviert bzw. deaktiviert werden

Weitere Informationen zu diesem Dialog siehe "FirstSpirit Handbuch für Administratoren".

3.3 Konfigurieren der Webanwendung

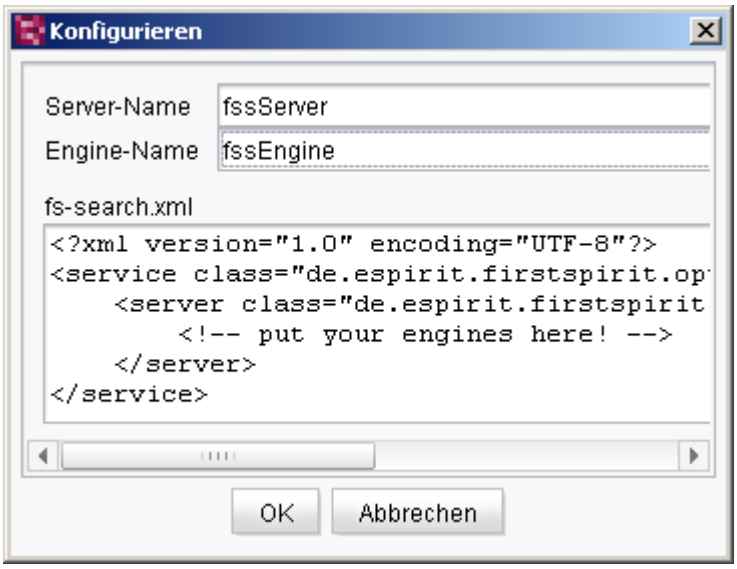

Abbildung 3-4: Konfigurieren der Webanwendung Integration

Zur Konfiguration von FirstSpirit Search stehen unterschiedliche Parameter zur Konfiguration zur Verfügung.

Server-Name: In diesem Feld wird der Name des Such-Servers eingetragen, der vom Modul als Einstiegspunkt für die Suche verwendet werden soll. Der hier eingetragene Name muss dem Namen des Servers innerhalb der Konfigurationsdatei "fs-search.xml" entsprechen (siehe unten).

Engine-Name: In diesem Feld wird der Name der Engine eingetragen, die vom Modul verwendet werden soll. Der hier eingetragene Name muss dem Namen einer Engine innerhalb der Konfigurationsdatei "fs-search.xml" entsprechen (siehe unten).

fs-search.xml: In diesem Feld wird die XML-basierte Konfiguration des Moduls vorgenommen. Die Konfigurationssyntax folgt dem Bausteinprinzip (siehe Kapitel 2, Seite 5) und wird in den nachfolgenden Kapiteln erläutern (ab Kapitel 4.1 ff.).

4 Syntax

4.1 Allgemeine Informationen

Die Konfiguration von FirstSpirit Search erfolgt mit XML (siehe Kapitel 3.3, Seite 12). Jeder Baustein (siehe Kapitel 2, Seite 5) wird mit einem eigenen XML-Tag beschrieben. Weiterhin verfügt die Syntax über allgemeine Attribute und allgemeine XML-Tags. Es werden sowohl die allgemeinen Attribute als auch die allgemeinen XML-Tags und die XML-Tags der einzelnen Bausteine erläutert.

Zusätzlich verfügt FirstSpirit Search über eine Tag-Library zur Steuerung der Webanwendung. Dabei werden sowohl allgemeine, formale Definitionen beschrieben (z. B. die Festlegung eines Präfixes für die JSP-Tags) als auch die Verwendung der verfügbaren JSP-Tags erläutert.

4.2 XML-Deklaration

XML-Dokumente müssen wohlgeformt und gültig sein, damit sie von einem Parser ausgelesen und interpretiert werden können. Eine Voraussetzung hierfür ist die Angabe einer XML-Deklaration.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
```

4.3 Allgemeine XML-Attribute

Allgemeine XML-Attribute können in allen Tags verwendet werden. Abhängig von ihrem Einsatzzweck werden allgemeine XML-Attribute in drei Kategorien unterteilt:

- Wiederverwendung:
  Attribute "id", "idref" und "fileref" (siehe Kapitel 4.3.1 Seite 14)
- Realisierung:
  Attribute "type" und "class" (siehe Kapitel 4.3.2, Seite 15)
- Bezeichner:
  Attribut "name" (siehe Kapitel 4.3.3, Seite 18)

4.3.1 Wiederverwendung (id, idref und fileref)

Über diese Attribute können zwei Aufgaben erfüllt werden:

- Wiederverwendung eines Objektes (siehe Kapitel 4.3.1.1, Seite 14)
- Wiederverwendung von XML-Dateien (siehe Kapitel 4.3.1.2, Seite 15)

4.3.1.1 Wiederverwendung eines Objektes (id und idref)

Mit dem Attribut "id" kann ein eindeutiger Bezeichner für ein Element festgelegt werden. Durch diesen eindeutigen Bezeichner ist es möglich, ein Element innerhalb der gesamten Konfiguration wieder zu verwenden.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  <engine id="ID" />
</service>
```

Im Beispiel wird für das Tag "engine" der eindeutige Bezeichner "ID" vergeben.

Um den eindeutigen Bezeichner in der Konfiguration wieder zu verwenden, ist zudem das Attribut "idref" nötig. Als Wert wird der mit dem Attribut "id" vergebene eindeutige Bezeichner erwartet. Die Verwendung des Attributes "idref" führt dazu, dass das Element des referenzierten Tags wiederverwendet wird. Somit wird von beiden Tags genau dasselbe Element bzw. Objekt verwendet.

Bei der Vergabe eines eindeutigen Bezeichners sollte darauf geachtet werden, dass das erste Zeichen ein Buchstabe ist. Grundsätzlich sollten nur folgende Zeichen verwendet werden: A-Z, a-z, 0-9 und _. Ein gültiger Bezeichner könnte somit "server01" lauten. Der Bezeichner muss innerhalb einer Konfiguration eindeutig vergeben werden.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  <engine id="ID">
    ...
  </engine>
  ...
  <engine idref="ID" />
</root>
```

Mit dem Attribut "id" wird für das erste Tag "engine" der eindeutige Bezeichner "ID" vergeben. Im zweiten Tag "engine" wird mit dem Attribut "idref", das Element des ersten Tags "engine" wieder verwendet (referenziert).

4.3.1.2 Wiederverwendung von XML-Dateien (fileref)

Das Attribut "fileref" kann in einem XML-Tag angegeben werden, um den Inhalt einer anderen XML-Datei einzufügen. Darüber können bestehende Konfigurationen wieder verwendet werden oder auch mehrere Konfigurationen zu einer Konfiguration zusammengefasst werden.

Als Wert erwartet das Attribut "fileref" einen URL. Dies kann ein Pfad in der Webanwendung (z. B. "/WEB-INF/datei.xml") oder im Dateisystem (z. B. "/home/user/datei.xml") sein.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  <service fileref="URL" />
</service>
```

4.3.2 Realisierung (type und class)

Die einzelnen Bausteine von FirstSpirit Search benötigen sowohl einfache als auch komplexe Objekte für die Verarbeitung.

Ein Beispiel für ein einfaches Objekt ist die Angabe einer Zeichenkette, als Parameter für eine Engine (String). Im Gegensatz dazu, handelt es sich bei der Engine selbst um ein komplexes Objekt.

Durch den modularen Aufbau von FirstSpirit Search, kann das Modul für die einzelnen Tags möglicherweise nicht entscheiden, welche Implementierung, welcher Adapter oder welcher Proxy zum Einsatz kommen soll. Weiterhin kann bei der Angabe eines Parameters die Interpretation nicht klar festgelegt werden. Das Modul kann beispielsweise nicht erkennen, ob die Angabe als Zeichenkette (String) oder aber als Zahl (Integer) behandelt werden soll. Aus diesem Grund muss bei der Verwendung von Tags und Attributen zusätzlich definiert werden, wie diese umzuwandeln sind bzw. interpretiert werden sollen.

Dazu können für jedes Tag die Attribute "type" und "class" angegeben werden. Mit diesen beiden Attributen kann der (abstrakte) Typ ("type") bzw. die konkrete

Realisierung ("class", Klassenname) eines Typs angegeben werden.

Wird das Attribut "type" nicht angegeben, so wird der Tagname als Typ interpretiert. Somit führen die Angaben von `<map />` oder `<tag type="map" />` beide dazu, dass als Typ "map" verwendet wird. Beim Tagnamen "attribute" ist der Standardtyp "string".

Beispiele:

```
<engine ... />
```

Hier wird als Typ "engine" verwendet.

```
<attribute type="list" ... />
```

Bei dieser Angabe wird als Typ "list" verwendet.

Da ein Typ auf unterschiedliche Art realisiert werden kann, muss für viele Typen angegeben werden, welche Realisierung verwendet werden soll. In Java kann eine Map beispielsweise durch "java.util.HashMap" und "java.util.TreeMap" realisiert werden.

Die Angabe der Realisierung erfolgt mit dem Attribut "class". Als Wert erwartet das Attribut den Klassenamen inkl. des Java-Packages (z. B. "java.util.HashMap").

Wird das Attribut "class" angegeben, so überprüft FirstSpirit Search, ob die angegebene Realisierung mit dem Typ übereinstimmt.

*Für viele Typen ist eine **Standardrealisierung** hinterlegt. Bei diesen Typen ist die Angabe des Attributes "class" **nicht zwingend**.*
*Ist für einen Typen **keine Standardrealisierung** hinterlegt, so ist die Angabe des Attributs "class" **zwingend**.*
Es wird empfohlen die Attribute "type" und "class" nur dann zu verwenden, wenn eine Angabe des Typs und der Realisierung zwingend nötig ist.

Die verwendbaren Typen können den nachfolgenden Tabellen entnommen werden. Für jeden Typ ist angegeben, ob eine Standardrealisierung hinterlegt ist und eine alternative Realisierung angegeben werden kann.

Die Typen werden unterteilt in:

- allgemeine Typen (siehe Kapitel 4.3.2.1 Seite 17)
- spezielle "FirstSpirit Search"-Typen (siehe Kapitel 4.3.2.2 Seite 17)

Beispiele zur Verwendung der beiden Typen finden sich in Kapitel 4.3.2.3 (ab Seite

18).

4.3.2.1 Allgemeine Typen

Die allgemeinen Typen werden ausschließlich als Wert für das Attribut "type" verwendet. Aus diesem Grund werden allgemeine und spezielle Typen (siehe Kapitel 4.3.2.2, Seite 17) miteinander kombiniert. Eine Gemeinsamkeit aller allgemeinen Typen ist, dass die Angabe des Attributs "class" nicht zwingend ist.

Typ	Beschreib.	Attribut "class"	Standard-Realisierung	Alternative Realisierung
string	Zeichenkette	Optional	java.lang.String	Keine
int	Zahl	Optional	java.lang.Integer	Keine
bool	Boolescher Wert	Optional	java.lang.Boolean	Keine
map	Menge von Schlüssel-Wert-Paaren	Optional	java.util.HashMap	java.util.TreeMap
list	Liste von Objekten	Optional	java.util.ArrayList	java.util.LinkedList
set	Menge von Objekten	Optional	java.util.HashSet	java.util.TreeSet

4.3.2.2 Spezielle Typen

Spezielle Typen können ausschließlich als Tag und nicht als Wert für das Attribut "type" verwendet werden. Aus diesem Grund werden allgemeine (siehe Kapitel 4.3.2.1 Seite 17) und spezielle Typen miteinander kombiniert.

Die Angabe des Attributs "class" ist für alle speziellen Typen, mit Ausnahme von "attribute", zwingend vorgeschrieben.

Typ	Beschreib.	Attribut "class"	Standard-Realisierung	Alternative Realisierung
engine	Such-Engine (Baustein "Engine")	Pflichtangabe	Keine	Keine

Typ	Beschreib.	Attribut "class"	Standard-Realisierung	Alternative Realisierung
server	Such-Server (Baustein "Engine")	Pflichtangabe	Keine	Keine
filter	Such-Filter	Pflichtangabe	Keine	Keine
attribute	Allgemeine Attribute (Parameter)	Optional	java.lang.String	Keine

4.3.2.3 Beispiele

Das folgende Beispiel erzeugt auf drei verschiedene Arten eine ansonsten identische Map:

```
<attribute class="java.util.HashMap" />
<attribute type="map" />
<map />
```

Die folgenden Einträge sind ungültig, weil entweder der Typ unbekannt oder die angegebene Klasse inkompatibel ist.

```
<unknowntag />
<attribute type="unknowntype" />
<map class="java.lang.String" />
<attribute type="map" class="java.util.ArrayList" />
```

4.3.3 Bezeichner (name)

Mit dem Attribut "name" kann ein Bezeichner für ein Tag angegeben werden. Beispielsweise kann bei der Konfiguration einer Engine mit dem Attribut "name" ein Bezeichner für die Engine angegeben werden. Beim speziellen Typ "attribute" (siehe Kapitel 4.3.2.2, Seite 17) entspricht der Wert des Attributs "name" dem Parameternamen für den umschließenden speziellen Typ.

Im ersten Beispiel wird für das Tag "engine" der Bezeichner "NAME" angegeben:

```
<?xml version="1.0" encoding="UTF-8"?>
<service>
  <engine name="NAME">
    ...
  </engine>
</service>
```

Im Gegensatz zum ersten Beispiel, wird im zweiten Beispiel der Parameter "PARAMETERNAME" für den speziellen Typ "engine" festgelegt:

```
...
<engine ... >
  <attribute name="PARAMETERNAME">...</attribute>
</engine>
...
```

Wird das Attribut "name" nicht angegeben, so wird der Tag-Name als Bezeichner interpretiert.

Daher sind folgende beide Angaben identisch:

```
<engine name="engine" ... >
```

und

```
<engine ... >
```

4.4 Allgemeine XML-Tags

Allgemeine XML-Tags können in allen Tags verwendet werden. Momentan verfügt FirstSpirit Search über das allgemeine XML-Tag "attribute".

4.4.1 Parameterangabe (attribute)

Mit dem Tag "attribute" können Parameter für ein Element angegeben werden. Die angegebenen Parameter gelten für das jeweils übergeordnete Element.

Für das Tag muss der Bezeichner (das Attribut "name") angegeben werden (siehe Kapitel 4.3.3 Seite 18). Das Attribut "name" erwartet in diesem Fall den Parameternamen als Wert:

```
<attribute name="PARAMETERNAME">
```

Der Parameterwert ist zwischen dem öffnenden und schließenden Tag anzugeben:

```
<attribute name="PARAMETERNAME">PARAMETERWERT</attribute>
```

4.5 Spezielle XML-Tags

4.5.1 Services

Ein Service stellt einen Dienst zur Verfügung. Mit einem Dienst werden administrative Aufgaben realisiert.

Momentan sind die folgenden Dienste für FirstSpirit Search verfügbar:

- Such-Server:
  Steuerung und Bereitstellung eines Such-Servers (siehe Kapitel 4.5.1.1 Seite 20)
- Logging:
  Konfiguration der Log-Ausgaben (siehe Kapitel 4.5.1.2, Seite 23)
- Re-Indizierung:
  Zeitabhängig Re- bzw. Neu-Indizierung einer Such-Engine
  (siehe Kapitel 4.5.1.3 Seite 21)

4.5.1.1 Such-Server

Über diesen Service können alle (innerhalb des <service>-Tags) enthaltenen Such-Engines und der Such-Server gestartet werden. Weiterhin stellt der Service den Clients (z. B. einer JSP-Seite) eine Schnittstelle zum Such-Server zur Verfügung, über die beispielsweise Anfragen gestellt werden können. Der Such-Server kann auf unterschiedliche Arten bereitgestellt werden:

- in der aktuellen Java Virtual Machine (JVM)
- durch Remote Method Invocation (RMI)
- durch das Java Naming and Directory Interface (JNDI)

Zur Konfiguration des Such-Servers ist ein <service>-Tag mit dem entsprechenden Klassennamen (Attribut "class") anzugeben. Der Service verfügt über mehrere Konfigurationsparameter, die mit einem <attribute>-Tag (innerhalb des <service>-Tags) angegeben werden können.

Die entsprechenden Attribute und Parameter können den nachfolgenden Tabellen entnommen werden. Bei den Werten handelt es sich um Zeichenketten. Aus diesem Grund braucht das Attribut "type" nicht angegeben zu werden (siehe Kapitel 4.3.2.2, Seite 17).

<service>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.service.adapter.ServerService

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
bindTo	Mit "bindTo" wird der Ort der Bereitstellung des Such-Servers festgelegt. Mögliche Werte sind: ▪ *local*: lokale JVM ▪ *rmi*: RMI-Registrierung ▪ *jndi*: JNDI-Kontext Es können mehrere kommaseparierte Werte angegeben werden. Wird ein fehlerhafter Wert für "bindTo" angegeben, so wird "local" verwendet.	*local,rmi*
createRMI[2]	Durch "createRMI" kann ein RMI-Server gestartet werden. Der Parameter erwartet einen Booleschen Wert. Mit *true* wird ein RMI-Server gestartet, mit *false* wird das Starten eines RMI-Servers verhindert. Stattdessen wird versucht, eine vorhandene RMI-Registrierung zu ermitteln.	*true*
localName	Lokaler Bezeichner für die JVM.	*FIRSTseek.SearchServer*
rmiHost[2]	Rechnername des RMI-Servers.	*localhost*
rmiName[2]	Name des Servers innerhalb der RMI-Registrierung.	*FIRSTseek.SearchServer*
rmiPort[2]	RMI-Port	*1099*

[1] "name"-Attribut
[2] Parameter wird nur ausgewertet, wenn für den Parameter "bindTo" der Wert "rmi" angegeben wurde

Innerhalb des <service>-Tags muss der <server>-Tag angegeben werden. Innerhalb des <server>-Tags können dann die gewünschten Engines (siehe Kapitel 4.5.4, Seite 35 ff.) definiert werden. Für das <server>-Tag muss das Pflichtattribut "class" angegeben werden.

Der Klassename lautet:
`de.espirit.firstspirit.opt.search.server.SimpleServer`.

Alternativ kann die Definition des Servers auch mit einem <attribute>-Tag (siehe Kapitel 4.4.1, Seite 19) in Verbindung mit der Angabe des speziellen Typs "server" (siehe Kapitel 4.3.2.2, Seite 17) erfolgen (<attribute type="server"...>). Es wird jedoch die Angabe des <server>-Tags empfohlen, um die Konfiguration möglichst übersichtlich zu gestalten.

<server>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.server.SimpleServer

[1] Pflichtparameter

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service
class="de.espirit.firstspirit.opt.search.service.adapter.ServerSer
vice">
  <server
class="de.espirit.firstspirit.opt.search.server.SimpleServer">
    ...
  </server>
  <attribute name="createRMI">false</attribute>
  <attribute name="bindTo">local</attribute>
  <attribute name="localName">fssServer</attribute>
</service>
```

Im Beispiel wird die Erzeugung eines RMI-Servers verhindert ("createRMI"). Als Bezeichner wird "fssServer" ("localName") vergeben. Zusätzlich wird festgelegt, dass die Bereitstellung des Such-Servers in der lokalen JVM erfolgen soll ("bindTo").

4.5.1.2 Logging

Dieser Service konfiguriert das in FirstSpirit Search verwendete Logging-System Log4j[1]. Der Einsatz des Services ist nur dann sinnvoll, wenn Log4j in der eingesetzten Umgebung nicht bereits konfiguriert ist.

Der Logging-Service sollte als erster Service gestartet werden, damit ein frühzeitiges Logging möglich ist.

Zur Konfiguration des Loggings ist ein <service>-Tag mit dem entsprechenden Klassennamen (Attribut "class") anzugeben. Die Konfigurationsparameter, die mit einem <attribute>-Tag (innerhalb des <service>-Tags) angegeben werden können, werden unverändert an Log4j weitergeleitet.

Die entsprechenden Attribute und Parameter können den nachfolgenden Tabellen entnommen werden. Bei den Werten handelt es sich um Zeichenketten. Aus diesem Grund braucht das Attribut "type" nicht angegeben zu werden (siehe Kapitel 4.3.2.2, Seite 17).

<service>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.service.adapter.Log4jService

[1] Pflichtparameter

<attribute>-Tag
Alle Parameter ("name"-Attribut) werden unverändert an Log4j weitergeleitet.

[1] Siehe http://logging.apache.org/log4j/

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service
class="de.espirit.firstspirit.opt.search.service.adapter.Log4jService">
  <attribute name="log4j.rootCategory">
    DEBUG, file
  </attribute>
  <attribute name="log4j.appender.file">
    org.apache.log4j.RollingFileAppender
  </attribute>
  <attribute name="log4j.appender.file.File">
    E:/FirstSpirit/log/fs-search.log
  </attribute>
  <attribute name="log4j.appender.file.MaxFileSize">
    5MB
  </attribute>
  <attribute name="log4j.appender.file.MaxBackupIndex">
    5
  </attribute>
  <attribute name="log4j.appender.file.layout">
    org.apache.log4j.PatternLayout
  </attribute>
  <attribute
name="log4j.appender.file.layout.ConversionPattern">%-5p %d (%c)
%m%n</attribute>
</service>
```

4.5.1.3 Re-Indizierung

Die einzelnen Such-Engines verfügen über eine Schnittstelle, um den Neuaufbau ihres Indexes zu initiieren. Mit dem Re-Indizierungs-Service kann der Neuaufbau des Indexes zu einem festgelegten Zeitpunkt und in festen zeitlichen Abständen gestartet werden.

Zur Konfiguration der Re-Indizierung ist ein <service>-Tag mit dem entsprechenden Klassennamen (Attribut "class") anzugeben. Der Service verfügt über mehrere Konfigurationsparameter, die mit einem <attribute>-Tag (innerhalb des <service>-Tags) angegeben werden können.

Die entsprechenden Attribute und Parameter können den nachfolgenden Tabellen entnommen werden. Bei den Werten handelt es sich um Zeichenketten. Aus diesem Grund braucht das Attribut "type" nicht angegeben zu werden (siehe Kapitel 4.3.2.2, Seite 17).

<service>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.service.RebuildIndexTimerService

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
engineURL[2]	Angabe des vollständigen Such-Engine-URLs. Der URL besteht aus mehreren Teilen: ▪ Server-URL ▪ Engine-Bezeichner Beispiel: *SERVER_URL[ENGINE]* Als Server-URL kann eine RMI-URL, eine JNDI-URL oder aber ein lokaler Bezeichner angegeben werden. Der lokale Bezeichner entspricht dem Wert des Parameters "localName" eines Such-Servers (siehe Kapitel 4.5.1.1, Seite 20). Ein RMI- und JNDI-URL ist in dem Format *//rmiHost:rmiPort/rmiName* anzugeben. Die Parameter "rmiHost", "rmiPort" und "rmiName" können dem Kapitel 4.5.1.1, Seite 20 entnommen werden. Beim Start des Service wird zunächst versucht einen lokalen Server zu ermitteln. Anschließend wird versucht mit RMI und danach mit JNDI eine Verbindung mit einem Such-Server herzustellen.	Keiner

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
startTime	Startzeitpunkt der Neuindizierung. Die Angabe erfolgt im Format *hh:mm*. Es ist zunächst die Stunde im 24-Stundenformat (*00-23*) anzugeben und getrennt durch einen Doppelpunkt die Minute (*00-59*). Beispiel: *21:00*	Startzeitpunkt des Service + 5 Sekunden
period[2]	Abstand zwischen zwei Neuindizierungen. Als Wert wird eine Zahl, gefolgt von einem Buchstaben erwartet. Mit diesem Buchstaben wird festgelegt, ob es sich bei der Angabe um Sekunden (*s*), Minuten (*m*), Stunden (*h*), Tage (*d*) oder Wochen (*w*) handelt. Beispiel: *1d*	Keiner
startNow	Mit diesem Parameter wird festgelegt, ob eine Neuindizierung – unabhängig vom Startzeitpunkt und Indizierungsintervall – beim Start des Services durchgeführt werden soll. Wird als Wert *true* angegeben, so wird eine Neuindizierung erzwungen, bei *false* jedoch nicht.	*false*

[1] "name"-Attribut

[2] Pflichtparameter

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service
class="de.espirit.firstspirit.opt.search.service.RebuildIndexTimer
Service">
  <attribute name="engineURL">fssServer[fssEngine]</attribute>
  <attribute name="startTime">6:00</attribute>
  <attribute name="period">24h</attribute>
  <attribute name="startNow">true</attribute>
</service>
```

Im Beispiel wird eine Neuindizierung konfiguriert. Beim Start der Webanwendung soll

immer eine Neuindizierung erfolgen ("startNow"). Diese Neuindizierung wird jeden Tag ("period") um 6:00 Uhr ("startTime") ausgeführt. Die Neuindizierung gilt für die Such-Engine "fssEngine" des lokalen Such-Servers "fssServer" ("engineURL").

4.5.2 Service-Proxy

Über einen Service-Proxy können mehrere Services zusammengefasst werden. Dies ist notwendig, da bei der Konfiguration der Webanwendung, über die Server- und Projektkonfiguration, nur eine XML-Datei konfiguriert werden kann (siehe Kapitel 3.3, Seite 12).

Der Service-Proxy wird innerhalb eines <service>-Tags definiert. Innerhalb dieses <service>-Tags wird mithilfe des <attribute>-Tags eine Liste von Services angegeben.

Der Parametername für die Liste von Services lautet: "services". Als Typ muss "list" angeben werden. Für das <attribute>-Tag dürfen somit nur weitere <service>-Tags angegeben werden.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service class="...">
  <attribute name="services" type="list">
    <service class="...">
      ...
    </service>
    <service class="...">
      ...
    </service>
    ...
  </attribute>
</service>
```

Die entsprechenden Attribute und Parameter können den nachfolgenden Tabellen entnommen werden.

<service>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.service.proxy.MultiServiceProxy

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
services[2][3]	Eine Liste von Services (<service>-Tags)	Keine

[1] "name"-Attribut
[2] Pflichtparameter
[3] Als "type"-Attribut ist "list" anzugeben

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service
class="de.espirit.firstspirit.opt.search.service.proxy.MultiServiceProxy">
  <attribute name="services" type="list">
    <service
class="de.espirit.firstspirit.opt.search.service.adapter.ServerService">
      <server
class="de.espirit.firstspirit.opt.search.server.SimpleServer">
        ...
      </server>
      <attribute name="createRMI">false</attribute>
      <attribute name="bindTo">local</attribute>
      <attribute name="localName">fssServer</attribute>
    </service>
    <service
class="de.espirit.firstspirit.opt.search.service.RebuildIndexTimerService">
      <attribute name="engineURL">fssServer[fssEngine]</attribute>
      <attribute name="startTime">6:00</attribute>
      <attribute name="period">24h</attribute>
      <attribute name="startNow">true</attribute>
    </service>
  </attribute>
</service>
```

Im Beispiel werden zwei Services zusammengefasst: ein Such-Server und eine Re-Indizierung.

4.5.3 Service-Runner

Bei einem so genannten Runner handelt es sich um die Steuerungsmöglichkeit eines Such-Services aus verschiedenen Umgebungen heraus.

Momentan werden folgende Runner von FirstSpirit Search zur Verfügung gestellt:

- Standalone: Startet einen Service in einer separaten JVM (siehe Kapitel 4.5.3.1 Seite 29)
- Web: Integration eines Services in einer Webanwendung (siehe Kapitel 4.5.3.2 Seite 32)
- JMX: Integriert einen Service in einem Java-Management-Extensions-Container (JMX) (siehe Kapitel 4.5.3.3 Seite 34)

4.5.3.1 Standalone

Mit dem Standalone-Service-Runner können Services in einer separaten JVM als eigenständiges Programm gestartet werden. Die Services werden in der angegebenen Reihenfolge gestartet und beim Beenden des Programms in umgekehrter Reihenfolge wieder gestoppt.

Sinnvoll ist der Einsatz des Service-Runners, wenn ein separater RMI-Server in einer eigenen JVM verwendet werden soll.

Der Klassenname des Standalone-Service-Runners lautet:

`de.espirit.firstspirit.opt.search.admin.standalone.ServiceRunner`.

Die Klasse befindet sich in der Runtime-Jar-Datei "fs-search-rt.jar", das in der Modul-Datei von FirstSpirit Search enthalten ist. Die Modul-Datei enthält viele der benötigten Klassen (z. B. Lucene-Core, Log4j) im "lib"-Verzeichnis. Weiterhin benötigen einige Services und Engines weitere Klassen, die im "fs-server.jar" enthalten sind.

Als Parameter erwartet der Service-Runner eine oder mehrere XML-Konfigurationsdateien.

Syntax:

```
java –cp CLASSPATH
de.espirit.firstspirit.opt.search.admin.standalone.ServiceRunner
XMLFILE1 XMLFILE2 ...
```

Über den Service-Proxy können mehrere Services gebündelt werden. Es besteht jedoch auch die Möglichkeit, die einzelnen Services auf mehrere XML-Dateien aufzuteilen.

Beispiel einer Service-Bündelung ("services.xml"):

```
<?xml version="1.0" encoding="UTF-8"?>
<service
class="de.espirit.firstspirit.opt.search.service.proxy.MultiServic
eProxy">
  <attribute name="services" type="list">
    <service
class="de.espirit.firstspirit.opt.search.service.adapter.Log4jServ
ice">
      <attribute name="log4j.rootCategory">
        DEBUG, file
      </attribute>
      <attribute name="log4j.appender.file">
        org.apache.log4j.RollingFileAppender
      </attribute>
      <attribute name="log4j.appender.file.File">
        E:/Standalone/fs-search.log
      </attribute>
      <attribute name="log4j.appender.file.MaxFileSize">
        5MB
      </attribute>
      <attribute name="log4j.appender.file.MaxBackupIndex">
        5
      </attribute>
      <attribute name="log4j.appender.file.layout">
        org.apache.log4j.PatternLayout
      </attribute>
      <attribute
name="log4j.appender.file.layout.ConversionPattern">%-5p %d (%c)
%m%n</attribute>
    </service>
    <service
class="de.espirit.firstspirit.opt.search.service.adapter.ServerSer
vice">
      <server
class="de.espirit.firstspirit.opt.search.server.SimpleServer">
        <engine name="fssEngine"
class="de.espirit.firstspirit.opt.search.engine.proxy.MonitorEngin
eProxy">
          <engine
class="de.espirit.firstspirit.opt.search.engine.spider.SpiderEngin
e">
            <attribute name="urls" type="list">

<attribute>E:/FIRSTsprit4/web/fs4staging_41038/41118/de/index.html
```

```
</attribute>
            </attribute>
            <attribute name="index">E:/Standalone/fs-
search.index</attribute>
          </engine>
        </engine>
      </server>
      <attribute name="createRMI">true</attribute>
      <attribute name="bindTo">rmi</attribute>
      <attribute name="rmiHost">localhost</attribute>
      <attribute name="rmiPort">7788</attribute>
      <attribute name="rmiName">fssServer</attribute>
    </service>
  </attribute>
</service>
```

Im Beispiel wird das Logging mit einem separaten RMI-Server (Such-Service) verbunden. Ein exemplarischer Aufruf in der Kommandozeile sieht folgendermaßen aus:

```
java -cp log4j-1.2.14.jar;lucene-core-2.1.0.jar;commons-
httpclient.jar;fs-search-rt.jar
de.espirit.firstspirit.opt.search.admin.standalone.ServiceRunner
services.xml
```

Diese Konfiguration kann in zwei Konfigurationen aufgeteilt werden.

"logging.xml":

```
<?xml version="1.0" encoding="UTF-8"?>
<service
class="de.espirit.firstspirit.opt.search.service.adapter.Log4jServ
ice">
  <attribute name="log4j.rootCategory">
    DEBUG, file
  </attribute>
  <attribute name="log4j.appender.file">
    org.apache.log4j.RollingFileAppender
  </attribute>
  <attribute name="log4j.appender.file.File">
    E:/Standalone/fs-search.log
  </attribute>
  <attribute name="log4j.appender.file.MaxFileSize">
    5MB
  </attribute>
  <attribute name="log4j.appender.file.MaxBackupIndex">
    5
  </attribute>
  <attribute name="log4j.appender.file.layout">
    org.apache.log4j.PatternLayout
  </attribute>
  <attribute
name="log4j.appender.file.layout.ConversionPattern">%-5p %d (%c)
%m%n</attribute>
</service>
```

"rmi_server.xml":

```
<?xml version="1.0" encoding="UTF-8"?>
<service
class="de.espirit.firstspirit.opt.search.service.adapter.ServerSer
vice">
  <server
class="de.espirit.firstspirit.opt.search.server.SimpleServer">
    <engine name="fssEngine"
class="de.espirit.firstspirit.opt.search.engine.proxy.MonitorEngin
eProxy">
      <engine
class="de.espirit.firstspirit.opt.search.engine.spider.SpiderEngin
e">
        <attribute name="urls" type="list">

<attribute>E:/FIRSTsprit4/web/fs4staging_41038/41118/de/index.html
</attribute>
        </attribute>
        <attribute name="index">E:/Standalone/fs-
search.index</attribute>
      </engine>
    </engine>
  </server>
  <attribute name="createRMI">true</attribute>
  <attribute name="bindTo">rmi</attribute>
  <attribute name="rmiHost">localhost</attribute>
  <attribute name="rmiPort">7788</attribute>
  <attribute name="rmiName">fssServer</attribute>
</service>
```

Der Aufruf in der Konsole lautet dann entsprechend:

```
java -cp log4j-1.2.14.jar;lucene-core-2.1.0.jar;commons-
httpclient.jar;fs-search-rt.jar
de.espirit.firstspirit.opt.search.admin.standalone.ServiceRunner
logging.xml rmi_server.xml
```

4.5.3.2 Web

Der Web-Service-Runner ist ein Servlet, das die Integration eines Services in eine neue oder bestehende Webanwendung ermöglicht. Neben dem automatischen Starten und Stoppen des Services, erlaubt das Servlet die Administration über die Laufzeitparameter, ohne das ein vorheriger Neustart der Webanwendung notwendig ist.

Der Service-Runner ist standardmäßig in der Datei "web.xml" konfiguriert, die bei einer Konfiguration über die Server- und Projektkonfiguration angelegt wird (siehe Kapitel 3.2, Seite 11):

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.4">
  ...
  <display-name>41133STAGING</display-name>
  <servlet>
    <servlet-name>fss-Init</servlet-name>
    <servlet-
class>de.espirit.firstspirit.opt.search.admin.web.ServiceServlet</
servlet-class>
    <init-param>
      <param-name>service</param-name>
      <param-value>/WEB-INF/fs-search.xml</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
  ...
</web-app>
```

Aus Sicherheitsgründen ist in der Datei "web.xml" standardmäßig kein Servlet-Mapping angegeben. Ein solches Mapping muss daher manuell ergänzt werden.

Nach dem Eintragen eines Servlet-Mappings kann der Service innerhalb der Webanwendung gestartet und gestoppt werden, wenn keine weiteren Absicherungsmaßnahmen getroffen werden.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.4">
  ...
  <display-name>41133STAGING</display-name>
  <servlet>
    <servlet-name>fss-Init</servlet-name>
    <servlet-
class>de.espirit.firstspirit.opt.search.admin.web.ServiceServlet</
servlet-class>
    <init-param>
      <param-name>service</param-name>
      <param-value>/WEB-INF/fs-search.xml</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
  ...
  <servlet-mapping>
    <servlet-name>fss-Init</servlet-name>
    <url-pattern>*.service</url-pattern>
  </servlet-mapping>
  ...
</web-app>
```

Nach dem Hinzufügen des Servlet-Mappings kann der Service nun gestartet und gestoppt werden.

Statusinformationen können mit folgendem Aufruf angezeigt werden:

```
PROTOKOLL://HOST:PORT/APPLIKATION/do.service
```

z. B. http://localhost/search/do.service

Der Aufruf zum Starten des Services lautet:

```
PROTOKOLL://HOST:PORT/APPLIKATION/do.service?action=start
```

z. B. http://localhost/search/do.service?action=start

Der Aufruf zum Stoppen des Services lautet:

```
PROTOKOLL://HOST:PORT/APPLIKATION/do.service?action=stop
```

z. B. http://localhost/search/do.service?action=stop

Der Aufruf zum Neuladen des Services lautet:

```
PROTOKOLL://HOST:PORT/APPLIKATION/do.service?action=reload
```

z. B. http://localhost/search/do.service?action=reload

4.5.3.3 JMX

Der Service-Runner für JMX ist ein Managed Bean (MBean)[2] und kann in beliebige JMX Agents (bspw. JBoss) integriert werden.

Der Klassename für den Service-Runner lautet:
`de.espirit.firstspirit.opt.search.admin.jmx.ServiceMX`

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
Service[2][3]	Pfad zur Konfigurationsdatei	Keine

[1] "name"-Attribut
[2] Pflichtparameter
[3] Großschreibung des Parameters ist erforderlich

[2] Siehe http://java.sun.com/products/JavaManagement/index.html.

Beispiel:

```
<mbean
    code="de.espirit.firstspirit.opt.search.admin.jmx.ServiceMX"
    name="fs:service=SearchMX">
  <attribute name="Service">META-INF/fs-search.xml</attribute>
</mbean>
```

4.5.4 Engines

4.5.4.1 Übersicht

Wie in Kapitel 2.1 (Seite 5 ff.) beschrieben, ist eine Such-Engine ein Baustein von FirstSpirit Search. Das Modul unterstützt eine Vielzahl von Engines (siehe Abbildung 2-2 auf Seite 6, Abbildung 2-3 auf Seite 6 und Abbildung 2-5 auf Seite 7).

Der nachfolgenden Tabelle können die einzelnen Such-Engines – sortiert nach Art (Implementierung / Adapter / Proxy) – entnommen werden.

Name	Art	Beschreibung	Kapitel
Spider-Engine	Engine-Implementierung	Eigene Suchmaschine mit integriertem Spider und Indexer für verschiedene Dokumentformate.	Kapitel 4.5.4.2 Seite 37
Lucene	Engine-Adapter	Integration eines bestehenden Lucene-Indexes.	Kapitel 4.5.4.2 Seite 37
JDBC	Engine-Adapter	Anbindung einer Datenbank.	Kapitel 4.5.4.4 Seite 48

Name	Art	Beschreibung	Kapitel
Einfacher Filter	Engine-Proxy	Statisches Filtern bestimmter Ergebnisse.	Kapitel 4.5.4.5 Seite 51
Mehrsprachigkeit (Filter)	Engine-Proxy	Unterstützung von Mehrsprachigkeit durch das Filtern bestimmter Ergebnisse.	Kapitel 4.5.4.6 Seite 53
Kategorisierung (Filter)	Engine-Proxy	Unterstützung von Kategorien durch das Filtern bestimmter Ergebnisse.	Kapitel 4.5.4.7 Seite 55
Einfacher Engine-Verbund	Engine-Proxy	Zusammenfassen mehrerer Engines.	Kapitel 4.5.4.8 Seite 57
Mehrsprachigkeit (Engine-Verbund)	Engine-Proxy	Simulation von Mehrsprachigkeit durch mehrere Engines.	Kapitel 4.5.4.9 Seite 59
Kategorisierung (Engine-Verbund)	Engine-Proxy	Simulation von Kategorien durch mehrere Engines.	Kapitel 4.5.4.10 Seite 61
Personalisierung	Engine-Proxy	Personalisierte Suchergebnisse, abhängig von den Gruppen- bzw. Benutzerrechten.	Kapitel 4.5.4.11 Seite 63
Session-Management	Engine-Proxy	Automatische Session-Timeouts	Kapitel 4.5.4.12 Seite 66

Bei den Definitionen und den Beispielen zu den Engines wird zur besseren Übersichtlichkeit das <engine>-Tag verwendet. Alternativ kann die Definition einer Engine auch mit einem <attribute>-Tag (siehe Kapitel 4.4.1, Seite 19) in Verbindung mit der Angabe des speziellen Typs "engine" (siehe Kapitel 4.3.2.2, Seite 17) erfolgen (<attribute type="engine"...>).

4.5.4.2 Engine-Implementierung "Spider Engine"

Die Spider-Engine ist eine eigene Implementierung der Suchmaschinenfunktionalität. Sie basiert auf einer Reihe offener Bibliotheken (Lucene[3], Apache POI[4], HTML Parser[5] und Jakarta Commons HttpClient[6]), die durch einen eigenen Spider zum Durchsuchen von Webseiten kombiniert wurden.

Die Engine unterstützt Mehrsprachigkeit und Kategorien, wenn diese Informationen als Meta-Tags im HTML vorhanden sind (siehe Parameter "categoryField", "categories", "localeField" und "locales"). Enthält die Suchanfrage eine Einschränkung, bzgl. Sprache oder Kategorie, so wird die an Lucene übergebene Query, um die entsprechende Feldsuche erweitert.

Ein Dokument wird indiziert, wenn seine URL erlaubt und nicht verboten ist (siehe Parameter "allowed" und "forbidden"). Der Indizierungsprozess ist beendet, wenn eine der folgenden drei Bedingungen eintritt:

- Alle Seiten wurden durchsucht.
- Das Zeitlimit wurde überschritten (siehe Parameter "maxTime").
- Die maximale Anzahl an Dokumenten wurde überschritten (siehe Parameter "maxDocuments").

Der erzeugte Lucene-Index wird in einem eigenen Verzeichnis unterhalb des Basisverzeichnisses (siehe Parameter "index") angelegt. Der Name dieses Verzeichnisses stellt den Startzeitpunkt der Indizierung in codierter Form dar. Schlägt eine Indizierung fehl, wird das dafür angelegte Verzeichnis aus Gründen der Fehlersuche nicht gelöscht. Im Erfolgsfall werden jedoch alte Verzeichnisse entfernt, sofern sie nicht noch von laufenden Such-Sessions verwendet werden. Der Verzeichnisname des aktuell gültigen Indexes befindet sich in der Datei "index.cfg".

Die indizierten Felder und Dokumente im Lucene-Index können mit dem Tool Luke - Lucene Index Toolbox[7] betrachtet werden (siehe Kapitel 8.1 Seite 96).

Die Indizierung von Teilen eines Dokumentes kann mit dem Kommentar
`<!-- noindex -->`

[3] http://lucene.apache.org/

[4] http://poi.apache.org/

[5] http://htmlparser.sourceforge.net/

[6] http://hc.apache.org/httpclient-3.x/index.html

[7] http://www.getopt.org/luke/

unterdrückt werden. Ab diesem Kommentar werden die Inhalte des Dokumentes nicht indiziert. Die Unterdrückung der Indizierung wird mit dem Kommentar
`<!-- index -->`
beendet.

Bei einer Verschachtelung von <!-- noindex --> Kommentaren wird die Anzahl der Kommentare ermittelt. Um hier die Unterdrückung der Indizierung zu beenden, müssen bei einer Verschachtelung die gleiche Anzahl von <!-- index --> Kommentaren angegeben werden.

Beispiel:

```
JA, dieser Abschnitt wird indiziert
<!-- noindex -->
NEIN, dieser Abschnitt wird nicht mehr indiziert
<!-- noindex -->
NEIN, dieser Abschnitt wird nicht mehr indiziert
<!-- index -->
NEIN, dieser Abschnitt wird nicht mehr indiziert
<!-- index -->
JA, ab hier wird wieder indiziert
```

<engine>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.engine.spider.SpiderEngine

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
analyzer	Angabe des zu verwendenden Lucene-Analyzers. Eine Liste der möglichen Analyzer kann der API-Dokumentation von Lucene entnommen werden: http://lucene.apache.org/java/2_1_0/api/org/apache/lucene/analysis/Analyzer.html Achtung: Die Konfiguration erfolgt nicht wie üblich über das Attribut-Element unter Angabe des Klassennamens, sondern über den folgenden Ausdruck: `<analyzer class="org.apache.lucene.analysis.SimpleAnalyzer"/>` (NICHT: `<attribute name="analyzer" class="org.apache.lucene.analysis.SimpleAnalyzer"/>`)	StandardAnalyzer
index[2]	Angabe des Basisverzeichnisses für den Lucene-Index	Keiner
defaultField	Mit diesem Parameter kann der Standard-Feldname für die Suche festgelegt werden. Standardmäßig werden bei der Indizierung durch Lucene die Inhalte im Feld "content" abgelegt.	content

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
categoryField[4]	In einer Webseite können die Seiten mit Metainformationen versehen werden (<meta>-Tags). Diese Informationen werden bei der Indizierung im Index abgelegt. Mit dem Parameter "categoryField" wird festgelegt, welche <meta>-Tag-Information für die Kategorien genutzt werden soll. Der Wert von "categoryField" entspricht somit dem Wert des Attributs "name" eines <meta>-Tags, z. B. "keywords". Zur Angabe der einzelnen Kategoriewerte kann der Parameter "categories" verwendet werden.	Keiner
categories[4]	Bei "categories" handelt es sich um eine Map, mit der ein einzelner interner Kategoriename (z. B. "Cat_Common") auf einen Wert in der <meta>-Tag-Angabe (z. B. "common") abgebildet werden kann. Der Vorteil bei der Vergabe von internen Kategorienamen ist, dass bei verschiedenen Engine-Konfigurationen mit dem gleichen Kategorienamen – trotz potentiell verschiedener Werte im <meta>-Tag – zugegriffen werden kann (s. u.).	Keiner
credentials	Authentifizierungsinformationen (s. u.)	Keiner
localeField[4]	Entspricht dem Parameter "categoryField", jedoch für die Angabe den Sprachfeldes. Wichtig ist hier, dass <meta>-Tags mit dem Attribut "http-equiv" nicht berücksichtigt werden.	Keiner
locales[4]	Entspricht dem Parameter "categories", jedoch für Sprachwerte und Sprachkürzel.	Keiner
allowed	Liste von erlaubten URLs (s. u.)	Alle URLs sind erlaubt

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
forbidden	Liste von nicht erlaubten URLs (s. u.)	Keine URLs sind verboten
maxContentLength	Größenbeschränkung für den zu speichernden Inhalt eines Dokumentes. Ist der Inhalte des Dokumentes größer als die Beschränkung, so werden die ersten Zeichen bis zum Erreichen der Größenbeschränkung abgelegt.	16384 (=16 kB)
maxDocuments	Anzahl der maximal erlaubten Dokumente für die Indizierung (Abbruchkriterium)	Anzahl ist unbeschränkt
maxFieldLength	Anzahl der Wörter, die maximal in einem Feld eines Dokuments gespeichert werden können. Enthält ein Dokument mehr Wörter, so wird es während der Indizierung abgeschnitten. Wird ein großer Wert gewählt, so muss genügend Hauptspeicher vorhanden sein.	10000
maxTime	Maximale Dauer des Indizierungsvorganges. Als Wert wird eine Zahl, gefolgt von einem Buchstaben erwartet. Mit diesem Buchstaben wird festgelegt, ob es sich bei der Angabe um Sekunden (*s*), Minuten (*m*), Stunden (*h*), Tage (*d*) oder Wochen (*w*) handelt. Beispiel: *1d*	Dauer ist unbeschränkt
maxThreads	Gleichzeitig laufende Threads zur Indizierung. Je mehr Threads hier angegeben werden, desto höher fällt der Speicherbedarf während der Indizierung aus.	1
threadPriority	Priorität der Indizierungsthreads (1=Minimum, 10=Maximum).	5

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
url oder urls[3]	Ein URL bzw. eine Liste von URLs (s. u.), die als Startpunkt von der Spider-Engine genutzt werden.	Keiner

[1] "name"-Attribut
[2] Pflichtparameter
[3] Die Angabe eines der beiden Parameter ist Pflicht
[4] Die Parameter "locales" und "localeField" bzw. "categories" und "categoryField" müssen jeweils zusammen angegeben werden

Parameter "allowed" / "forbidden":

Bei den Parametern "allowed" und "forbidden" wird ein <attribute>-Tag definiert. Innerhalb dieses Tags wird mithilfe weiterer <attribute>-Tags eine Liste von Filtern (Teile eines URLs) angegeben. Für diese <attribute>-Tags ist der folgende RegexWebLinkFilter zu verwenden, für den ein regulärer Ausdruck (in Java-Syntax) anzugeben ist. Klasse:
`de.espirit.firstspirit.opt.search.engine.spider.link.RegexWebLinkFilter`

Als Typ für die Parameter "allowed" und "forbidden" muss "list" angeben werden. Für die inneren <attribute>-Tags, ist der RegexWebLinkFilter zu verwenden.

Beispiel:

```
<attribute name="allowed" type="list">
  <attribute
class="de.espirit.firstspirit.opt.search.engine.spider.link.RegexW
ebLinkFilter">intranet.meinServer.de/de/</attribute>
 <attribute
class="de.espirit.firstspirit.opt.search.engine.spider.link.RegexW
ebLinkFilter">intranet.meinServer.de/en/</attribute>
</attribute>
```

Parameter "urls":

Die Angabe der URLs ist der Angabe der Parameter "allowed" und "forbidden" ähnlich, jedoch wird in den inneren <attribute>-Tags kein "class"-Attribut angegeben.

Beispiel:

```
<attribute name="urls" type="list">
<attribute>http://localhost:80/site1/index.html</attribute>
```

```
<attribute>http://localhost:80/site2/index.html</attribute>
</attribute>
```

Parameter "categories" / "locales":

Für diese beiden Parameter wird eine Map mit Schlüssel-/Wertpaaren angegeben. Als "type"-Attribut muss der Wert "map" angegeben werden. Innerhalb des Parameters wird für jedes Schlüssel-/Wertpaar ein <attribute>-Tag angegeben. Der Schlüssel wird mit dem "name"-Attribut angegeben, der Wert innerhalb des öffnenden und schließenden Tags.

Bei dem Schlüssel ("name"-Attribut) handelt es sich um den intern zu verwendenden Kategorienamen bzw. das zu verwendende Sprachkürzel (z. B. "int"). Mit dem Wert wird angegeben, wie der im Such-Index gespeicherte Wert für eine Kategorie oder ein Sprachkürzel lautet. Durch die Angabe von "categoryField" bzw. "localeField" wird festgelegt, in welchem Feld der Wert für eine Kategorie bzw. ein Sprachkürzel im Such-Index gespeichert ist.

Beispiel:

```
<attribute name="locales" type="map">
  <attribute name="de">DE</attribute>
  <attribute name="en">GB</attribute>
</attribute>
<attribute name="localeField">language</attribute>
```

Das Beispiel enthält zwei Schlüssel-/Wertpaare für den Parameter "locales". Das erste Paar besteht aus dem Schlüssel "de" und dem Wert "DE". Beim zweiten Paar lautet der Schlüssel "en" und der Wert "GB". Durch den Parameter "localeField" wird festgelegt, dass die Werte für die Sprachkürzel im Such-Index im Feld "language" zu finden sind. Die Werte für die Sprachkürzel in diesem Feld lauten "DE" (Deutsch) und "GB" (Englisch). In FirstSpirit Search werden durch die Festlegung mit dem Parameter "locales" für diese Sprachkürzel "de" und "en" verwendet.

```
<attribute name="categories" type="map">
  <attribute name="int">intro</attribute>
  <attribute name="doc">documentation</attribute>
</attribute>
<attribute name="categoryField">categories</attribute>
```

Parameter "credentials":

Falls die zu indizierende Website eine Authentifizierung erfordert, können die dafür nötigen Informationen in der Spider-Engine konfiguriert werden. Der Parameter "credentials" wird als Liste von Map-Einträgen konfiguriert, wobei jeder einzelne Eintrag für einen Bereich und eine Authentifizierungsmethode steht. Damit ist es möglich, sich gegenüber verschiedenen Servern, in einem Indexdurchlauf, unterschiedlich zu authentifizieren.

Die Parameter der Map-Einträge:

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
authType[2]	Typ der Authentifizierung. Mögliche Werte sind: "ntlm" oder "basic".	Keiner
authUsername[2]	Angabe des Benutzernamens	Keiner
authPassword[2]	Angabe der Passwortes für den angegebenen Benutzers	Keiner
authHost[2][3]	Angabe des Servers gegen den die Authentifizierung erfolgen soll	Keiner
authDomain[2][3]	Authentifizierungsdomain	Keiner
scopeHost	Die Authentifizierung gilt für den angegebenen Host.	ANY_HOST (also: alle Hosts)
scopePort	Die Authentifizierung gilt für den angegebenen Port	ANY_PORT (also: alle Ports)

[1] "name"-Attribut
[2] Pflichtparameter
[3] Angabe nur bei "authType" "ntlm"

Beispiel:

```
<attribute name="credentials" type="list">
  <attribute type="map">
    <attribute name="authType">ntlm</attribute>
    <attribute name="authUsername">smith</attribute>
    <attribute name="authPassword">verysecret</attribute>
    <attribute name="authHost">myserver</attribute>
    <attribute name="authDomain">MYCOMPANY</attribute>
  </attribute>
</attribute>
```

Vollständiges Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  ...
  <engine
class="de.espirit.firstspirit.opt.search.engine.spider.SpiderEngin
e">
    <attribute name="urls" type="list">
        <attribute>http://intranet.mydomain.com</attribute>
        <attribute>http://extranet.mydomain.com</attribute>
    </attribute>
    <attribute name="index">/var/www/search/index</attribute>
    <attribute name="maxThreads">2</attribute>
    <attribute name="threadPriority">1</attribute>
    <attribute name="allowed" type="list">
      <attribute class="
de.espirit.firstspirit.opt.search.engine.spider.link.RegexWebLinkF
ilter ">intranet</attribute>
      <attribute class="
de.espirit.firstspirit.opt.search.engine.spider.link.RegexWebLinkF
ilter ">extranet</attribute>
    </attribute>
    <attribute name="maxDocuments">1000</attribute>
    <attribute name="maxTime">60m</attribute>
    <attribute name="credentials" type="list">
      <attribute type="map">
        <attribute name="authType">ntlm</attribute>
        <attribute name="authUsername">smith</attribute>
        <attribute name="authPassword">verysecret</attribute>
        <attribute name="authHost">myserver</attribute>
        <attribute name="authDomain">MYCOMPANY</attribute>
      </attribute>
    </attribute>
  </engine>
</service>
```

4.5.4.3 Engine-Adapter "Lucene"

Der Lucene-Engine-Adapter integriert einen bestehenden Index, der mit der Lucene-Bibliothek erstellt wurde.

Die Engine unterstützt Mehrsprachigkeit und Kategorien, wenn diese Informationen als Felder in den Dokumenten vorhanden sind (siehe Parameter "categoryField", "categories", "localeField" und "locales"). Enthält die Suchanfrage eine Einschränkung, bzgl. der Sprache oder der Kategorie, so wird die an Lucene übergebene Anfrage (Query), um die entsprechende Feldsuche erweitert.

<engine>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.engine.adapter.LuceneEngineAdapter

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
analyzer	Angabe des zu verwendenden Lucene-Analyzers. Eine Liste der möglichen Analyzer kann der API-Dokumentation von Lucene entnommen werden: http://lucene.apache.org/java/2_1_0/api/org/apache/lucene/analysis/Analyzer.html Achtung: Die Konfiguration erfolgt nicht wie üblich über das Attribut-Element mit Angabe des Klassennamens, sondern über den folgenden Ausdruck: `<analyzer class="org.apache.lucene.analysis.SimpleAnalyzer"/>` (NICHT: `<attribute name="analyzer" class="org.apache.lucene.analysis.SimpleAnalyzer"/>`)	StandardAnalyzer

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
index[2]	Angabe des Basisverzeichnisse für den Lucene-Index	Keiner
defaultField	Mit diesem Parameter kann der Standard-Feldname für die Suche festgelegt werden. Standardmäßig werden bei der Indizierung durch Lucene die Inhalte im Feld "content" abgelegt.	content
categoryField[3]	In einer Webseite können die Seiten mit Metainformationen versehen werden (<meta>-Tags). Diese Informationen werden bei der Indizierung im Index abgelegt. Mit dem Parameter "categoryField" wird festgelegt, welche <meta>-Tag-Information für die Kategorien genutzt werden soll. Der Wert von "categoryField" entspricht somit dem Wert des Attributs "name" eines <meta>-Tags, z. B. "keywords". Zur Angabe der einzelnen Kategoriewerte kann der Parameter "categories" verwendet werden.	Keiner
categories[3]	Bei "categories" handelt es sich um eine Map, mit der ein einzelner interner Kategoriename (z. B. "Cat_Common") auf einen Wert in der <meta>-Tag-Angabe (z. B. "common") abgebildet werden kann. Der Vorteil bei der Vergabe von internen Kategorienamen ist, dass bei verschiedenen Engine-Konfigurationen mit dem gleichen Kategorienamen – trotz potentiell verschiedener Werte im <meta>-Tag – zugegriffen werden kann (s. u.).	Keiner
localeField[3]	Entspricht dem Parameter "categoryField", jedoch für die Angabe des Sprachfeldes. Wichtig ist hier, dass <meta>-Tags mit dem Attribut "http-equiv" nicht berücksichtigt werden.	Keiner
locales[3]	Entspricht dem Parameter "categories", jedoch für Sprachwerte und Sprachkürzel	Keiner

[1] "name"-Attribut

[2] Pflichtparameter

[3] Die Parameter "locales" und "localeField" bzw. "categories" und "categoryField" müssen jeweils

zusammen angegeben werden

Parameter "categories" / "locales":
Für diese beiden Parameter wird eine Map mit Schlüssel-/Wertpaaren angegeben. Als "type"-Attribut muss der Wert "map" angegeben werden. Innerhalb des Parameters wird für jedes Schlüssel-/Wertpaar ein <attribute>-Tag angegeben. Der Schlüssel wird mit dem "name"-Attribut angegeben, der Wert innerhalb des öffnenden und schließenden Tags.

Bei dem Schlüssel ("name"-Attribut) handelt es sich um den intern zu verwendenden Kategorienamen bzw. das zu verwendende Sprachkürzel (z. B. "int"). Mit dem Wert wird angegeben, wie der im Such-Index gespeicherte Wert für eine Kategorie oder eine Sprachkürzel lautet. Durch die Angabe von "categoryField" bzw. "localeField" wird festgelegt, in welchem Feld der Wert für eine Kategorie bzw. ein Sprachkürzel im Such-Index gespeichert ist.

Beispiel:

```
<attribute name="locales" type="map">
  <attribute name="de">DE</attribute>
  <attribute name="en">GB</attribute>
</attribute>
<attribute name="localeField">language</attribute>
```

Das Beispiel enthält zwei Schlüssel-/Wertpaare für den Parameter "locales". Das erste Paar besteht aus dem Schlüssel "de" und dem Wert "DE". Beim zweiten Paar lautet der Schlüssel "en" und der Wert "GB". Durch den Parameter "localeField" wird festgelegt, dass die Werte für die Sprachkürzel im Such-Index im Feld "language" zu finden sind. Die Werte für die Sprachkürzel in diesem Feld lauten "DE" (Deutsch) und "GB" (Englisch). In FirstSpirit Search werden durch die Festlegung mit dem Parameter "locales" für diese Sprachkürzel "de" und "en" verwendet.

```
<attribute name="categories" type="map">
  <attribute name="int">intro</attribute>
  <attribute name="doc">documentation</attribute>
</attribute>
<attribute name="categoryField">categories</attribute>
```

4.5.4.4 Engine-Adapter "JDBC"

Der JDBC-Engine-Adapter bindet beliebige Datenbanken an FirstSpirit Search an, für die ein JDBC-Treiber vorhanden ist. Er wandelt die Suchanfrage in einen passenden SQL-Befehl um und extrahiert aus den Datenbankergebnissen die gewünschten Spalten bzw. Spaltenwerte. Auf die Ergebnisse kann dann in der Webanwendung zugegriffen werden.

<engine>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.engine.adapter.JdbcEngineAdapter

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
driver[2]	Klassenname des JDBC-Treibers, z. B. `de.espirit.or.impl.mysql.MySQLLayer`	Keiner
url[2]	JDBC-URL	Keiner
username[2]	Benutzername für die Authentifizierung über den JDBC-Treiber an der Datenbank	Keiner
password[2]	Passwort für die Authentifizierung des Benutzers über den JDBC-Treiber an der Datenbank	Keiner
columns[2]	Angabe der Datenbankspalten (Spaltenname in der Datenbank), die für die Anfrage berücksichtigt werden sollen	Alle Spalten ("*")
rawQuery[2]	Mit diesem Parameter kann die Abfrage eingeschränkt werden (WHERE-Klausel). Innerhalb des Parameters kann der Platzhalter `$query$` verwendet werden. Dieser Platzhalter wird gegen den Suchtext ersetzt	Keiner

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
resultMapping[2]	Durch den Parameter werden die Werte der einzelnen Datenbankspalten auf Variablen, die in der Webanwendung verwendet werden können, abgebildet. Als Wert erwartet der Parameter eine kommaseparierte Liste von Schlüssel-/Wertpaaren. Der Schlüssel (Variablenbezeichner) wird vom Wert (Datenbankspalte) durch ein Gleichheitszeichen ("=") getrennt. Beim Wert bzw. der Datenbankspalte gibt es eine Besonderheit: der Name der Datenbankspalte muss in Dollar-Zeichen (z. B: $SPALTENNAME$) eingefasst werden. Beispiel: `content=$DATA$,title=$TITLE$` Wichtig ist, dass für die Verwendung in der Webanwendung die folgenden Variablen im resultMapping auf Datenbankspalten abgebildet werden sollten, falls sie in der Anwendung verwendet werden: ▪ "author" (Autor) ▪ "categories" (Kategorien) ▪ "content" (Inhalt) ▪ "date" (Datum) ▪ "locale" (Sprache) ▪ "score" (Trefferquote) ▪ "size" (Größenangabe) ▪ "title" (Titel) ▪ "url" (URL zum Treffer)	Keiner
fixedCategory	Suchergebnisse werden mit der angegebenen Kategorie zurückgeliefert.	Keiner

[1] "name"-Attribut

[2] Pflichtparameter

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  ...
  <engine
class="de.espirit.firstspirit.opt.search.engine.adapter.JdbcEngine
Adapter">
    <attribute
name="driver">de.espirit.or.impl.mysql.MySQLLayer</attribute>
    <attribute
name="url">jdbc:mysql://localhost:3306/intranet</attribute>
    <attribute name="user">user</attribute>
    <attribute name="password">password</attribute>
    <attribute name="columns">*</attribute>
    <attribute name="tables">CALLS</attribute>
    <attribute name="rawQuery">(BESCHREIBUNG like "%$query$%") or
(SUBJECT like "%$query$%") or (LOESUNG like "%$query$%") or (KUNDE
like "%$query$%") or (KONTAKT like "%$query$%") or (CALLSTATUS
like "%$query$%")</attribute>
    <attribute
name="resultMapping">title=$KUNDE$,content=$BESCHREIBUNG$,score=10
00,url=http://intranet/db/view.jsp?id=$ID$</attribute>
    <attribute name="fixedCategories">db</attribute>
  </engine>
</service>
```

4.5.4.5 Engine-Proxy "Einfacher Filter"

Dieser Engine-Proxy fügt jeder Suche eine Menge fest vorgegebener Filter hinzu. Damit lassen sich beispielsweise sicherheitsrelevante Informationen herausfiltern oder ungültige Datensätze ausblenden. Die angegebenen Filter werden automatisch "oder"-verknüpft. Es werden nur die Ergebnisse zurückgeliefert, auf die die Filterbedingung zutrifft (Positiv-Filter).

 *Innerhalb des <engine>-Tags des Proxys muss eine weitere Engine definiert oder referenziert werden!*

<engine>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.engine.proxy.FilterEngineProxy

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
filters	Liste von Ergebnisfiltern. Diese Filter werden jeder Suche hinzugefügt. Die einzelnen Filter werden hierbei "oder"-verknüpft (s. u.).	Keiner

[1] "name"-Attribut

Parameter "filters":
Beim Parameter "filters" wird ein <attribute>-Tag definiert. Innerhalb dieses Tags wird mithilfe des <filter>-Tags eine Liste von Filtern angegeben. Für diese <filter>-Tags, ist der RegexFilter (Klasse: `de.espirit.firstspirit.opt.search.filter.RegexFilter`) zu verwenden. Im <filter>-Tag sind zwei weitere <attribute>-Tags anzugeben.

Für das erste Attribut mit dem Wert "property" für das "name"-Attribut, ist die Eigenschaft (z. B. die Variable "url") anzugeben, auf die der Filterausdruck angewendet werden soll.

Das zweite Attribut (mit dem "name"-Attributwert "pattern") enthält den Filterausdruck, der als regulärer Ausdruck (in Java-Syntax) anzugeben ist.

Als Typ für den Parameter "filters" muss "list" angeben werden. Für das <filter>-Tag muss die Klasse des RegexFilters angegeben werden. Innerhalb des <filter>-Tags sind zwei <attribute>-Tags ("property" und "pattern") anzugeben.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  ...
  <engine
class="de.espirit.firstspirit.opt.search.engine.proxy.FilterEngine
Proxy">
    <engine idref="_engine" />
    <attribute name="filters" type="list">
      <filter class="de.espirit.firstspirit.opt.search.filter.
RegexFilter">
        <attribute name="property">url</attribute>
        <attribute name="pattern">\.html?$</attribute>
      </filter>
      <filter class="de.espirit.firstspirit.opt.search.filter.
RegexFilter">
        <attribute name="property">url</attribute>
        <attribute name="pattern">\.php$</attribute>
      </filter>
      <filter class="de.espirit.firstspirit.opt.search.filter.
RegexFilter">
        <attribute name="property">url</attribute>
        <attribute name="pattern">\.jsp$</attribute>
      </filter>
    </attribute>
  </engine>
  ...
</service>
```

Im Beispiel werden drei Filter für die Variable "url" definiert. Der erste Filter lässt nur Elemente zu, deren URL auf ".htm" oder ".html" enden. Durch den zweiten Filter werden alle Elemente zugelassen, deren URL auf ".php" enden. Beim dritten Filter werden alle Elemente zugelassen, deren URL auf ".jsp" enden.

4.5.4.6 Engine-Proxy "Mehrsprachigkeit (Filter)"

Dieser Engine-Proxy simuliert Mehrsprachigkeit mithilfe von Filtern. Hierbei entspricht eine Sprache genau einem Filter. Wird die Suche auf eine bestimmte Sprache eingegrenzt, wird der dazugehörige Filter aktiviert und verwendet. Bei der Rückgabe der Ergebnisse werden diese mithilfe der Filter analysiert und einer Sprache zugeordnet.

Innerhalb des <engine>-Tags des Proxys muss eine weitere Engine definiert oder referenziert werden!

<engine>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.engine.proxy.FilterLocalizeEngineProxy

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
locales	Mithilfe einer Map werden Sprachkürzel (z. B. "de") auf die Filter-Objekte des Suchergebnisses abgebildet (s. u.).	Keiner

[1] "name"-Attribut

Parameter "locales":
Beim Parameter "locales" wird ein <attribute>-Tag definiert. Innerhalb dieses Tags wird mithilfe des <filter>-Tags eine Map von Filtern angegeben. Für diese <filter>-Tags ist der RegexFilter zu verwenden:
Klasse: `de.espirit.firstspirit.opt.search.filter.RegexFilter`
Zusätzlich muss für jedes <filter>-Tag im "name"-Attribut ein Sprachkürzel (z. B. "de") angegeben werden. Hiermit wird festgelegt, für welche Sprache, welcher Filter angewendet wird.

Im <filter>-Tag sind zwei weitere <attribute>-Tags anzugeben:

Für das erste Attribut mit dem Wert "property" für das "name"-Attribut, ist die Eigenschaft anzugeben, auf die der Filterausdruck angewendet werden soll.

Das zweite Attribut (mit dem "name"-Attributwert "pattern") enthält den Filterausdruck, der als regulärer Ausdruck (in Java-Syntax) anzugeben ist.

Als Typ für den Parameter "locales" muss "map" angeben werden. Für das <filter>-Tag muss der Sprachkürzel ("name") und die Klasse des RegexFilters angegeben werden. Innerhalb des <filter>-Tags sind zwei <attribute>-Tags ("property" und "pattern") anzugeben.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  ...
  <engine
class="de.espirit.firstspirit.opt.search.engine.proxy.FilterLocali
zeEngineProxy">
    <engine idref="_engine" />
    <attribute name="locales" type="map">
      <filter name="de"
class="de.espirit.firstspirit.opt.search.filter.RegexFilter">
        <attribute name="property">url</attribute>
        <attribute name="pattern">/de/</attribute>
      </filter>
      <filter name="en"
class="de.espirit.firstspirit.opt.search.filter.RegexFilter">
        <attribute name="property">url</attribute>
        <attribute name="pattern">/en/</attribute>
      </filter>
    </attribute>
  </engine>
</service>
```

Im Beispiel werden zwei Filter für die Variable "url" definiert. Der erste Filter ist für Deutsch (Sprachkürzel "de") definiert und wird angewendet, wenn der URL die Zeichenkette "/de/" enthält. Im Gegensatz dazu, ist der zweite Filter für Englisch (Sprachkürzel "en") definiert und wird angewendet, wenn der URL die Zeichenkette "/en/" enthält.

4.5.4.7 Engine-Proxy "Kategorisierung (Filter)"

Dieser Engine-Proxy simuliert Kategorien mithilfe von Filtern. Hierbei entspricht eine Kategorie genau einem Filter. Wird die Suche auf bestimmte Kategorien eingegrenzt, werden die dazugehörigen Filter aktiviert und verwendet. Bei der Rückgabe der Ergebnisse werden diese mithilfe der Filter analysiert und kategorisiert.

Innerhalb des <engine>-Tags des Proxys muss eine weitere Engine definiert oder referenziert werden!

<engine>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.engine.proxy.FilterCategorizeEngineProxy

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
categories	Mithilfe einer Map werden Kategorien (z. B. "pdf") auf die Filter-Objekte des Suchergebnisses abgebildet (s. u.).	Keiner
inheritCategories	Durch den Parameter kann festgelegt werden, ob die Kategorien der zu filternden Such-Engine geerbt (Wert "true") oder aber ignoriert (Wert "false") werden sollen.	false

[1] "name"-Attribut

Parameter "categories":

Beim Parameter "categories" wird ein <attribute>-Tag definiert. Innerhalb dieses Tags wird mithilfe des <filter>-Tags eine Map von Filtern angegeben. Für diese <filter>-Tags ist der RegexFilter zu verwenden:

Klasse: `de.espirit.firstspirit.opt.search.filter.RegexFilter`.

Zusätzlich muss für jedes <filter>-Tag im "name"-Attribut eine Kategorie (z. B. "pdf") angegeben werden. Hiermit wird festgelegt, für welche Kategorie, welcher Filter angewendet wird.

Im <filter>-Tag sind zwei weitere <attribute>-Tags anzugeben. Für das erste Attribut mit dem Wert "property" für das "name"-Attribut, ist die Eigenschaft anzugeben, auf die der Filterausdruck angewendet werden soll. Das zweite Attribut (mit dem "name"-Attributwert "pattern") enthält den Filterausdruck, der als regulärer Ausdruck (in Java-Syntax) anzugeben ist.

Als Typ für den Parameter "categories" muss "map" angeben werden. Für das <filter>-Tag muss der Sprachkürzel ("name") und die Klasse des RegexFilters angegeben werden. Innerhalb des <filter>-Tags sind zwei <attribute>-Tags ("property" und "pattern") anzugeben.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  ...
  <engine
class="de.espirit.firstspirit.opt.search.engine.proxy.FilterCatego
rizeEngineProxy">
    <engine idref="_engine" />
    <attribute name="categories" type="map">
      <filter name="pdf"
class="de.espirit.firstspirit.opt.search.filter.RegexFilter">
        <attribute name="property">url</attribute>
        <attribute name="pattern">\.pdf$</attribute>
      </filter>
      <filter name="html"
class="de.espirit.firstspirit.opt.search.filter.RegexFilter">
        <attribute name="property">url</attribute>
        <attribute name="pattern">\.html?$</attribute>
      </filter>
      <filter name="office"
class="de.espirit.firstspirit.opt.search.filter.RegexFilter">
        <attribute name="property">url</attribute>
        <attribute name="pattern">(\.doc$)|(\.xsl$)</attribute>
      </filter>
    </attribute>
  </engine>
</service>
```

Im Beispiel werden drei Filter für die Variable "url" definiert. Mit dem ersten Filter wird allen Dokumenten, deren URL auf ".pdf" enden, die Kategorie "pdf" zugewiesen. Die Kategorie "html" erhalten alle Dokumente, deren URL auf ".htm" oder ".html" enden (zweiter Filter). Die Kategorie "office" wird Dokumenten zugewiesen, deren URL auf ".doc" oder ".xsl" endet (dritter Filter).

4.5.4.8 Engine-Proxy "Einfacher Engine-Verbund"

Dieser Engine-Proxy fasst, ähnlich einer Meta-Suchmaschine, mehrere Engines zu einer zusammen. Ein solcher Zusammenschluss wird auch als Engine-Verbund bezeichnet. Dabei werden die jeweiligen Kategorien und Sprachen berücksichtigt und eine Suche wird nur an diejenigen Engines weitergeleitet, die diese auch unterstützen. Die Ergebnisse der einzelnen Engines werden, abhängig vom

gewünschten Sortierkriterium, zusammengeführt und zurückgeliefert. Das Vorgehen kann der nachfolgenden Abbildung entnommen werden (siehe Abbildung 4-1).

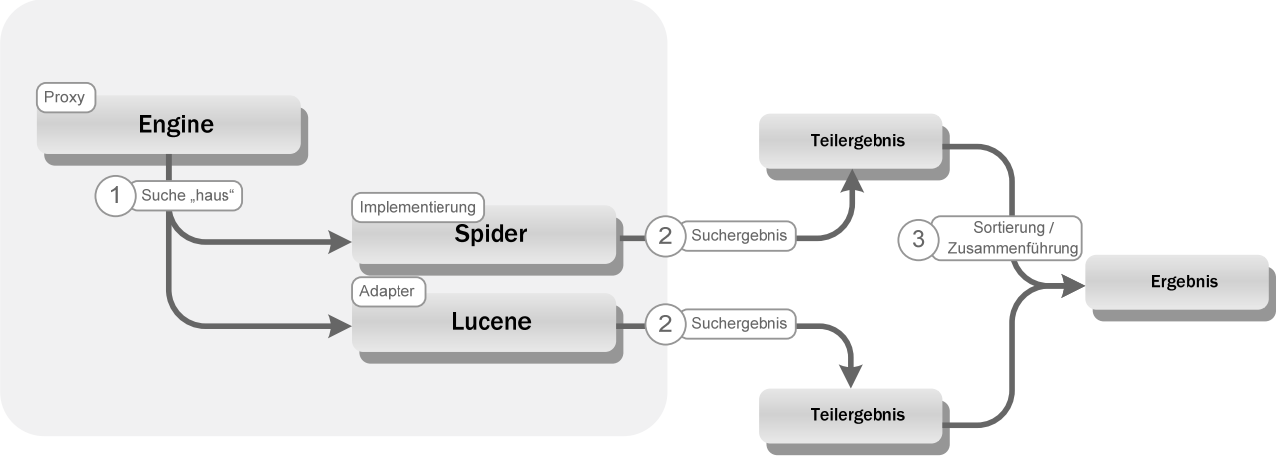

Abbildung 4-1: Suchanfrage/Suchergebnis beim Multi-Engine-Proxy

<engine>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.engine.proxy.MultiEngineProxy

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
engines[2]	Eine Liste von Such-Engines (s. u.).	Keiner

[1] "name"-Attribut
[2] Pflichtparameter

Parameter "engines":
Beim Parameter "engines" wird ein <attribute>-Tag definiert. Innerhalb dieses Tags wird mithilfe von <engine>-Tags eine Liste von Such-Engines angegeben.

Als Typ für den Parameter "engines" muss "list" angeben werden. Weiterhin müssen eine oder mehrere Such-Engines innerhalb des <attribute>-Tags angegeben werden.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  ...
  <engine
class="de.espirit.firstspirit.opt.search.engine.proxy.MultiEngineP
roxy">
    <attribute name="engines" type="list">
      <engine idref="_engine_de" />
      <engine idref="_engine_en" />
    </attribute>
  </engine>
</service>
```

4.5.4.9 Engine-Proxy "Mehrsprachigkeit (Engine-Verbund)"

Dieser Engine-Proxy simuliert die Mehrsprachigkeit durch den Zusammenschluss mehrerer Engines (Engine-Verbund). Dabei wird eine Sprache auf genau eine Engine abgebildet. Wird die Suche auf eine bestimmte Sprache begrenzt, so wird die Anfrage nur an die entsprechende Engine weitergeleitet. Umgekehrt wird, abhängig von der Engine (die das Ergebnis liefert), das Ergebnis um die entsprechenden Sprachinformationen erweitert. Die Ergebnisse der einzelnen Engines werden, abhängig vom gewünschten Sortierkriterium, zusammengeführt und zurückgeliefert.

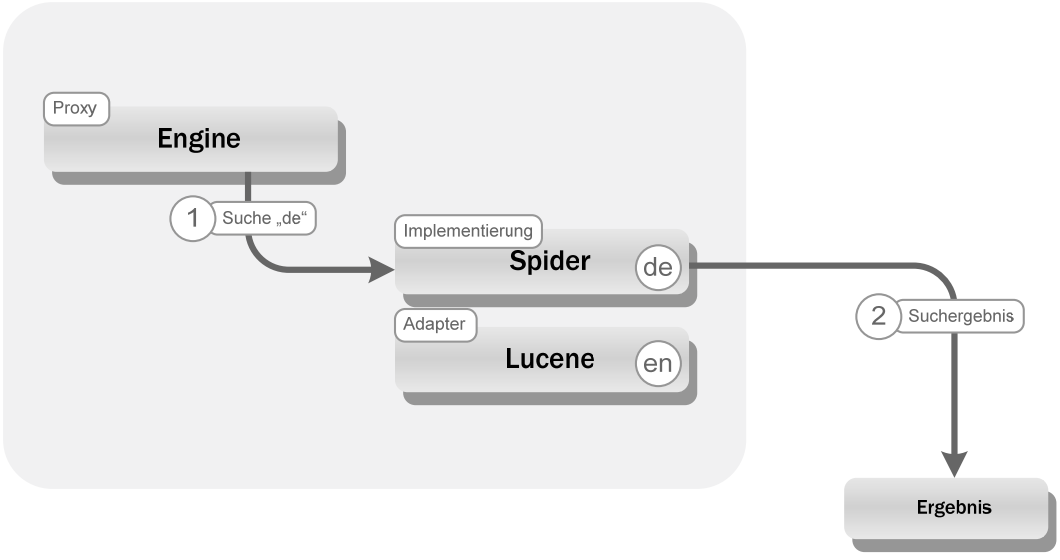

Abbildung 4-2: Suchanfrage/Suchergebnis beim MultiLocalizeEngineProxy

<engine>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.engine.proxy.MultiLocalizeEngineProxy

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
locales[2]	Abbildung von Sprachkürzeln (z. B. "de") auf einzelne Such-Engines (s. u.).	Keiner

[1] "name"-Attribut
[2] Pflichtparameter

Parameter "locales":
Für diesen Parameter wird eine Map mit Schlüssel-/Wertpaaren angegeben. Als "type"-Attribut muss der Wert "map" angegeben werden. Innerhalb des Parameters wird für jedes Schlüssel-/Wertpaar ein <engine>-Tag angegeben. Der Schlüssel wird mit dem "name"-Attribut angegeben, der Wert innerhalb des öffnenden und schließenden Tags. Beim Wert handelt es sich um das Engine-Objekt.

Beispiel:

```
<attribute name="locales" type="map">
  <engine name="de" idref="_engine_de" />
  <engine name="en" idref="_engine_en" />
</attribute>
```

Das Beispiel enthält zwei Schlüssel-/Wertpaare für den Parameter "locales". Das erste Paar besteht aus dem Schlüssel (Sprachkürzel) "de" und dem Engine-Objekt, welches durch das Attribut "idref" referenziert wird ("_engine_de"). Beim zweiten Paar lautet der Schlüssel (Sprachkürzel) "en" und hier wird die Engine mit der ID "_engine_en" referenziert.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  ...
  <engine
class="de.espirit.firstspirit.opt.search.engine.proxy.MultiLocaliz
eEngineProxy">
    <attribute name="locales" type="map">
      <engine name="de" idref="_engine_de" />
      <engine name="en" idref="_engine_en" />
    </attribute>
  </engine>
</service>
```

4.5.4.10 Engine-Proxy "Kategorisierung (Engine-Verbund)"

Dieser Engine-Proxy simuliert Kategorien durch den Zusammenschluss mehrerer Engines (Engine-Verbund). Dabei wird eine Kategorie auf genau eine Engine abgebildet. Wird die Suche dann auf bestimmte Kategorien begrenzt, so wird die Anfrage nur an die entsprechenden Engines weitergeschickt. Umgekehrt wird, abhängig von der Engine (die das Ergebnis liefert), das Ergebnis um die entsprechenden Kategorien erweitert. Eine Sucheanfrage wird an jede definierte Engine gestellt. Die Ergebnisse der einzelnen Engines werden, abhängig vom gewünschten Sortierkriterium, zusammengeführt und zurückgeliefert.

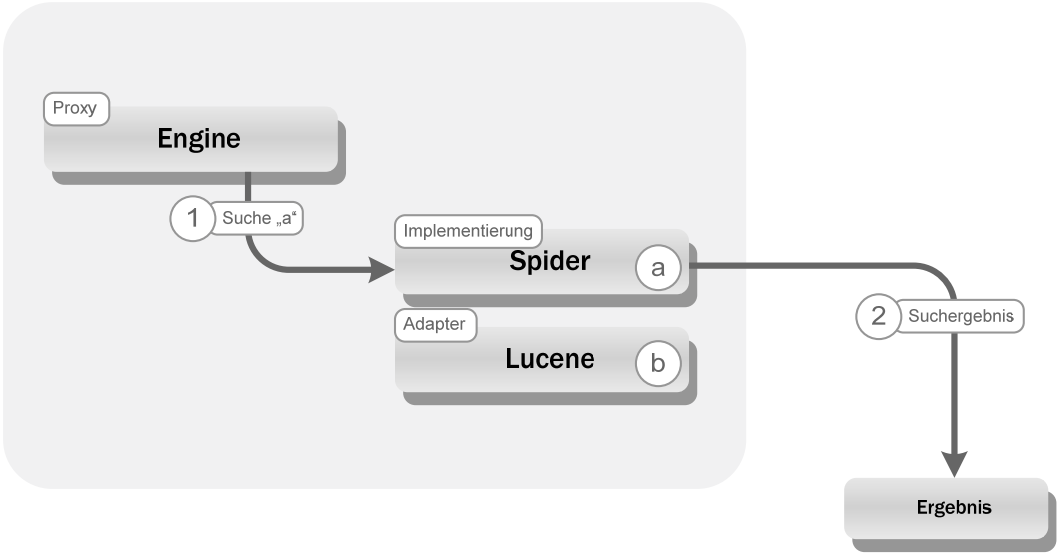

Abbildung 4-3: Suchanfrage/Suchergebnis beim MultiCategorizeEngineProxy

<engine>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.engine.proxy.MultiCategorizeEngineProxy

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
categories[2]	Abbildung von Kategorien (z. B. "pdf", "intranet") auf einzelne Such-Engines (s. u.).	Keiner

[1] "name"-Attribut
[2] Pflichtparameter

Parameter "categories":
Für diesen Parameter wird eine Map mit Schlüssel-/Wertpaaren angegeben. Als "type"-Attribut muss der Wert "map" angegeben werden. Innerhalb des Parameters wird für jedes Schlüssel-/Wertpaar ein <engine>-Tag angegeben. Der Schlüssel wird mit dem "name"-Attribut angegeben, der Wert innerhalb des öffnenden und schließenden Tags. Beim Wert handelt es sich um das Engine-Objekt.

Beispiel:

```
<attribute name="categories" type="map">
  <engine name="intranet" idref="_engine_intranet" />
  <engine name="internet" idref="_engine_internet" />
</attribute>
```

Das Beispiel enthält zwei Schlüssel-/Wertpaare für den Parameter "categories". Das erste Paar besteht aus dem Schlüssel (Kategorie) "intranet" und dem Engine-Objekt, welches durch das Attribut "idref" referenziert wird ("_engine_intranet"). Beim zweiten Paar lautet der Schlüssel (Kategorie) "internet" und hier wird die Engine mit der ID "_engine_internet" referenziert.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  ...
  <engine
class="de.espirit.firstspirit.opt.search.engine.proxy.MultiCategor
izeEngineProxy">
    <attribute name="categories" type="map">
      <engine name="intranet" idref="_engine_intranet" />
      <engine name="internet" idref="_engine_internet" />
    </attribute>
  </engine>
</service>
```

4.5.4.11 Engine-Proxy "Personalisierung"

Mit diesem Engine-Proxy können Suchergebnisse abhängig von den Rechten der Gruppen- bzw. der Benutzer ausgegeben werden. Durch die Verbindung von FirstSpirit Search und FirstSpirit Personalisation können bestimmten Benutzergruppen Suchergebnisse vorenthalten werden bzw. bestimmte Suchergebnisse ausgewählten Benutzergruppen zur Verfügung gestellt werden.

Damit eine Personalisierung von Dokumenten vorgenommen werden kann, müssen für die einzelnen Dokumente im Suchindex die erlaubten Gruppen hinterlegt werden. Um für die Dokumente Gruppeninformationen zu hinterlegen, können in den HTML-Dateien weitere Metainformationen (<meta>-Tag) angegeben werden, die durch eine Spider-Engine in den Suchindex aufgenommen werden (siehe Kapitel 4.5.4.2, Seite 37).

Beispiel:

```
<meta name="groups" content="ceo,qeo" />
```

Mithilfe des <meta>-Tags, soll das aktuelle Dokument im Beispiel nur für die Gruppen "ceo" und "qeo" einsehbar sein. Bei einer Indizierung durch die Spider-Engine würde eine entsprechende Information im Index zu diesem Dokument abgelegt (unter der Eigenschaft "groups" mit dem Inhalt "ceo,geo").

Bei einer Suchanfrage (Client-Seite bzw. Webanwendung) wird die Suche um die Gruppen (bzw. Rollen) des aktuellen Benutzers ergänzt.

Durch die Gruppeninformationen des aktuell angemeldeten Benutzers und der einzelnen Dokumente ist es dem Proxy möglich, nur Dokumente im Suchergebnis zurückzuliefern, für die der Benutzer über ausreichende Berechtigungen verfügt.

Innerhalb des <engine>-Tags des Proxys muss eine weitere Engine definiert oder referenziert werden!

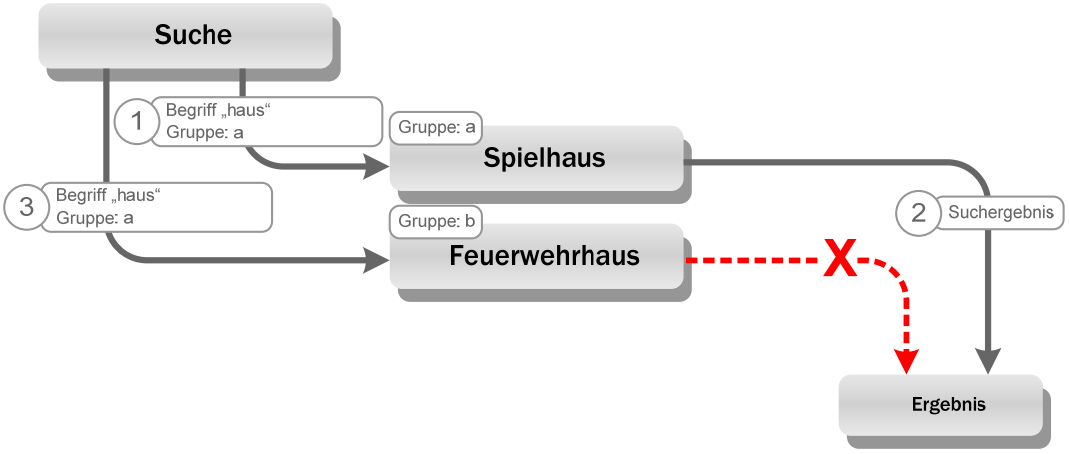

Abbildung 4-4: Suchanfrage/Suchergebnis beim SecurityEngineProxy

<engine>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.engine.proxy.SecurityEngineProxy

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
acceptUndefinedResults	Mit diesem Parameter kann angegeben werden, ob Suchergebnisse zugelassen werden, für die keine Gruppen- bzw. Rolleninformationen hinterlegt wurden. Bei der Angabe von "true" werden Suchergebnisse ohne Gruppeninformationen zugelassen, bei "false" nicht.	false

resultKey[2]	Über "resultKey" wird der Bezeichner bzw. der Feldname für die Gruppeninformation (Eigenschaft im obigen Beispiel), der zu einem Dokument im Suchindex hinterlegt ist, angegeben.	Keiner
propertyKey[2]	Angabe des Attributnamens, unter dem die Gruppeninformationen des aktuell angemeldeten Benutzers in der HTTP-Session (Webanwendung) hinterlegt sind. Bei FirstSpirit Personalisation werden die Gruppen unter dem Session-Attribut "FIRSTpersonalisation.usergroups" abgelegt.	Keiner
urlGroupMappings	Angabe eines URL zu einer Datei mit zusätzlichen Gruppeninformationen. Dadurch können weitere Dateien, beispielsweise PDF-Dateien, um Gruppeninformationen erweitert werden. In der angegebenen Datei muss folgendes Format verwendet werden: `URL=Gruppen` Bei `URL` handelt es sich um die absolute URL ausgehenden von der Webanwendung. Für `Gruppen` ist eine kommaseparierte Liste von Gruppen anzugeben.	Keiner

[1] "name"-Attribut
[2] Pflichtparameter

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  ...
  <engine
class="de.espirit.firstspirit.opt.search.engine.proxy.SecurityEngi
neProxy">
    <engine idref="_engine" />
    <attribute name="acceptUndefinedResults">false</attribute>
    <attribute name="resultKey">groups</attribute>
    <attribute
name="propertyKey">FIRSTpersonalisation.usergroups</attribute>
    <attribute name="urlGroupMappings">file:/META-
INF/secinfo.txt</attribute>
  </engine>
</service>
```

`/META-INF/secinfo.txt`:

```
/media/doc/howto-shutdown.pdf=admin
/media/office/gehaltsabrechnungen.xls=office,ceo
```

4.5.4.12 Engine-Proxy "Session-Management"

Dieser Engine-Proxy überwacht eine Engine und fügt den zurückgegebenen Sessions ein Timeout-Handling hinzu. Sind diese Sessions eine gewisse Zeit ohne Zugriff, werden sie automatisch geschlossen und die damit verbundenen Ressourcen freigegeben.

Innerhalb des <engine>-Tags des Proxys muss eine weitere Engine definiert oder referenziert werden!

<engine>-Tag	
Attribut	**Wert**
class[1]	de.espirit.firstspirit.opt.search.engine.proxy.MonitorEngineProxy

[1] Pflichtparameter

<attribute>-Tag		
Parameter[1]	**Wert**	**Standardwert**
sessionTimeout	Angabe des Session-Timeouts in Millisekunden.	1800000 (30 Min.)

[1] "name"-Attribut

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<service ...>
  ...
  <engine
class="de.espirit.firstspirit.opt.search.engine.proxy.MonitorEngin
eProxy">
    <engine idref="_engine" />
    <attribute name="sessionTimeout">30000</attribute>
  </engine>
</service>
```

5 JSP-Tags

5.1 Allgemeine Informationen

Das folgende Kapitel erläutert die Funktionen der Tag-Library zur Steuerung der Webanwendung. Dabei werden sowohl allgemeine, formale Definitionen beschrieben (z. B. die Festlegung eines Präfixes für die JSP-Tags) als auch die Verwendung der verfügbaren JSP-Tags erläutert.

5.2 Präfix

Für die Verwendung von FirstSpirit Search muss in den JSP-Seiten zunächst die entsprechende Taglib angegeben werden. Diese Dokumentation verwendet das Präfix "fss" für FirstSpirit-Search-Tags.

Beispiel für die Einbindung in JSP-Seiten:

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ taglib uri="fs-search" prefix="fss" %>
```

Zu beachten ist hierbei, dass der URI für FirstSpirit immer "fs-search" lautet.

*Wird das Präfix "fss" auf einen anderen Wert geändert, so ist dieses Präfix für die einzelnen Tags zu verwenden, d.h. "<**myPrefix**:isTrue>" anstelle von "<**fss**:isTrue>".*

5.3 Durch Tags definierte Variablen

In den Tags `<fss:getSearchDetails>`, `<fss:iterateResults>`, `<fss:navFrame>`, `<fss:navTop>`, `<fss:navBottom>`, `<fss:url>` werden Variablen definiert.

Dabei kann es zu Überschneidung mit manuell definierten Variablen in der JSP-Seite kommen.

Um Überschneidungen zu vermeiden, sollten folgende Variablenbezeichner nicht (bzw. mit Bedacht) verwendet werden:

- absNo
- ascending
- author
- categories
- content
- cursor
- cursorId
- date
- encodedUrl
- locales
- orderBy
- pageNo
- pageSize
- query
- relNo
- score
- se
- size
- title
- totalPages
- totalResults
- url
- urlPageNo
- urlPageSize

Welches Tag, welche Variable definiert, kann den einzelnen Tag-Beschreibungen entnommen werden (siehe Kapitel 5.5 ff.).

5.4 Übersicht

Die FirstSpirit-Search-JSP-Tags ermöglichen die Präsentation der Suchergebnisse in einer Webanwendung, sowie die Anzeige und Ermittelung von Detailinformationen (Gesamtanzahl der Ergebnisse, usw.) und die Darstellung einer dazu passenden Navigation.

Übersicht der JSP-Tags:

Name	Beschreibung
`isTrue`, `isFalse`	Bedingte Ausgabe von Inhalten.
`getSearchDetails`[1]	Informationen zur Suche abfragen.
`iterateResults`[2]	Ausgabe der Suchergebnisse.
`isCategory`[1], `isLocale`[1]	Entscheidet die Frage, ob ein Suchergebnis einer Kategorie bzw. einer Sprache angehört.
`highlighter`	Hervorgehobene Darstellung der Suchwörter.
`navigation`	Anzeige der Navigationsleiste.
`navTop`[1], `navBottom`[1]	Kopf bzw. Fuß der aktuellen Navigation anzeigen.
`navFrame`[1][2]	Fenster der aktuellen Navigation anzeigen.
`url`[1]	URL erzeugen.

[1] informelle Tags

[2] Ausgabe-Tags

Informelle Tags geben Auskunft über einen Zustand oder eine Gegebenheit, beispielsweise die Zugehörigkeit eines Ergebnisses zu einer Kategorie. Die Ausgabe

kann beispielweise bei der Prüfung auf Zugehörigkeit zu einer Kategorie (Bedingung) unterschiedlich gestaltet werden (bedingte Ausgabe), je nachdem ob ein Ergebnis zu dieser Kategorie gehört oder nicht. Hierfür stehen die beiden JSP-Tags `<fss:isTrue>` und `<fss:isFalse>` zur Verfügung. Ist die geprüfte Bedingung (Ergebnis gehört zu einer bestimmten Kategorie) erfüllt, so wird der Inhalt in `<fss:isTrue>` angezeigt. Wenn die Bedingung nicht erfüllt ist (Ergebnis gehört nicht zu einer bestimmten Kategorie), so wird der Inhalt in `<fss:isFalse>` angezeigt. Informelle Tags werden genau einmal ausgeführt.

Ausgabe-Tags dagegen werden mehrfach ausgeführt, beispielsweise für jedes Suchergebnis der aktuellen Ergebnisseite.

5.5 Suchergebnisse anzeigen

Suchergebnisse werden in einer Webanwendung seitenweise dargestellt. Die Seitengröße, also der Anzahl der Ergebnisse je (virtueller) Seite, wird beim Start der Suche über den Parameter `pageSize` bestimmt. Dieser Wert kann jedoch während der Anzeige jederzeit geändert werden.

5.5.1 Informationen zur Abfrage (<fss:getSearchDetails>)

Dieses JSP-Tag stellt einige Werte der aktuellen Suche als Variablen zur Verfügung. Wurde mindestens ein Ergebnis gefunden, so wird der Inhalt ausgegeben, der innerhalb des Tags `<fss:isTrue>` angegeben ist. Konnte kein Ergebnis gefunden werden, so wird der innerhalb des Tags `<fss:isFalse>` angegebene Inhalt ausgegeben. Wird weder ein `<fss:isTrue>`, noch ein `<fss:isFalse>` angegeben, so wird der Inhalt immer ausgegeben.

Innerhalb des Tags `<fss:getSearchDetails>` stehen die Variablen "ascending", "cursor", "cursorId", "orderBy", "pageNo", "pageSize", "query", "se", "totalPages" und "totalResults" zur Verfügung.

Variablen:

Variable	Bedeutung	Rückgabedatentyp
ascending	Sortierung der Ergebnisse. Der Wert ist "true", wenn die Ergebnisse alphabetisch aufsteigend sortiert sind und "false", wenn sie absteigend sortiert sind.	Boolean
cursor	Cursor-Objekt. Liefert das Objekt zur aktuell angezeigten (virtuellen) Seite zurück.	WebCursor[8]
cursorId	Liefert die ID des Cursor-Objektes zurück	String
orderBy	Liefert zurück, welches Feld für die Sortierung der Ergebnisse verwendet wurde ("author", "date", "score", "size" oder "title").	String
pageNo	Nummer der aktuell angezeigten (virtuellen) Seite (beginnend bei "1").	Integer
pageSize	Anzahl der auf der Seite darzustellenden Elemente.	Integer
query	Liefert den Suchtext zurück.	String
se	Liefert den Name der verwendeten Such-Engine zurück.	String
totalPages	Anzahl aller (virtuellen) Seiten (Berechnungs-Grundlage: Anzahl der Datensätze und "pageSize")	Integer
totalResults	Anzahl aller Elemente (über alle Seiten). Ist die Anzahl der Elemente (noch) nicht bekannt, wird -1 zurückgeliefert.	Integer

Beispiel:

```
<fss:getSearchDetails>
  <fss:isTrue>
    <p><%= totalResults %> Ergebnisse gefunden.</p>
  </fss:isTrue>
  <fss:isFalse>
    Keine Ergebnisse gefunden!
  </fss:isFalse>
</fss:getSearchDetails>
```

[8] Vollständiger Klassennamen: de.espirit.firstspirit.opt.search.web.WebCursor

5.5.2 Ausgabe aller Elemente einer Seite (<fss:iterateResults>)

Bei einer Suche werden die Ergebnisse auf unterschiedliche (virtuelle) Seiten verteilt. Mit dem JSP-Tag `<fss:iterateResults>` werden die Ergebnisse der aktuellen (virtuellen) Seite ausgegeben. Neben den bekannten Feldern wie "Autor", "Titel" und "URL", können mithilfe des Attributs "properties" beliebige Werte aus den Ergebnissen zur Verfügung gestellt werden. Auf diesem Wege wird sichergestellt, dass auch auf frei definierbare Metainformationen aus den Dokumenten zugegriffen werden kann.

Die Verwendung des JSP-Tags `<fss:iterateResults>` ist nur dann möglich, wenn Suchergebnisse gefunden werden konnten. Eine Prüfung, ob Sucheregebnisse gefunden wurden, kann mit dem JSP-Tag `<fss:getSearchDetails>` realisiert werden. Das JSP-Tag `<fss:iterateResults>` sollte aus diesem Grund innherhalb von `<fss:isTrue>` in einem `<fss:getSearchDetails>` JSP-Tag verwendet werden.

Attribute:

Attribut	Erwarteter Wert	Pflichtparameter
dateFormat	Angabe des zu verwendenden Datumsformats, z. B. "dd.MM.yyyy". Die Syntax für das Datumsformat kann der Sun-API-Dokumentation[9] entnommen werden.	Nein

properties	Mithilfe dieses Attributs können neben den Standardvariablen, zusätzliche Informationen der Ergebnisse zur Verfügung gestellt werden (z. B. Rechte).
	
	Das Attribut erwartet eine kommaseparierte Liste von Definition. Eine Definition erfolgt im Format:
	
	`variableName`
	`variableName:variableType`
	`variableName:variableType=resultKey`
		Nein

[9] http://java.sun.com/j2se/1.5.0/docs/api/index.html?java/text/SimpleDateFormat.html

Attribut	Erwarteter Wert	Pflichtparameter

	`variableName=resultKey`
	
	Durch "variableName" wird festgelegt, wie die Variable innerhalb der JSP-Seite heißt. Mit der optionalen Angabe ":variableType" kann der Typ der Variable festgelegt werden (z. B. java.lang.String). Wird der Typ nicht angegeben, so wird "java.lang.Object" verwendet.
	
	Wird optional "=resultKey" angegeben, so wird das angegebene Feld für ein Ergebnis verwendet. Wird das Feld nicht angegeben, wird der Wert von "variableName" als Feldbezeichner verwendet.
	
	Beispiel:
	`groups:java.lang.String`
	`groups`
	`g:java.lang.String:groups`	

Variablen:

Variable	Bedeutung	Rückgabedatentyp
absNo	Nummer des Ergebnisses in Bezug auf die Gesamtzahl der Ergebnisse (über alle Seiten) (beginnend bei "1").	Integer
author	Liefert den Autor zurück.	String
categories	Liefert die Kategorien für ein Dokument zurück.	String[]
content	Liefert einen Auszug aus dem Dokument zurück, in dem der Suchbegriff gefunden wurde.	String
date	Liefert das Datum des Dokumentes zurück. Die Formatierung des Datums erfolgt mit dem Format, welches mit dem Attribut "dateFormat" angegeben wurde. Wurde kein Datumsformat angegeben, so wird das Standard-Datumsformat verwendet.	String
locale	Liefert die Sprache zurück.	Locale

relNo	Nummer des Ergebnisses innerhalb einer (virtuellen) Seite.	Integer
score	Relevanz bzw. Güte eines Ergebnisses (0 bis 100).	Integer
size	Liefert die Größe des Dokumentes zurück. Die Maßeinheit ist von der verwendeten Such-Engine abhängig und kann von Such-Engine zu Such-Engine variieren. Üblicherweise wird als Maßeinheit Bytes verwendet.	Long
title	Liefert den Titel des Dokumentes zurück.	String
url	Liefert den URL des Dokumentes zurück.	String

Beispiel:

```
<fss:getSearchDetails>
  <fss:isTrue>

    <fss:iterateResults
properties="tel:java.lang.String=CUST_PHONE,fax:java.lang.String=C
UST_FAX">
      <%= relNo %> von <%= pageSize %>
      Telefon: <%= tel %>
      Telefax: <%= fax %>
    </fss:iterateResults>

  </fss:isTrue>
  <fss:isFalse>
    Keine Ergebnisse gefunden!
  </fss:isFalse>
</fss:getSearchDetails>
```

Im Beispiel werden zwei zusätzliche Werte aus den Ergebnissen als Variablen des Typs "String" zur Verfügung gestellt. Dabei bekommen "CUST_PHONE" und "CUST_FAX" in der JSP-Seite die Bezeichnungen "tel" und "fax".

5.6 Überprüfungen

5.6.1 Kategorieüberprüfung (<fss:isCategory>)

Das Tag `<fss:isCategory>` kann innerhalb der Tags `<fss:getSearchDetails>` (siehe Kapitel 5.5.1, Seite 70) und `<fss:iterateResults>` (siehe Kapitel 5.5.2, Seite 72) verwendet werden.

Bei der Verwendung innerhalb von <fss:getSearchDetails> gibt <fss:isCategory> Auskunft über die verwendeten Suchparameter und innerhalb von <fss:iterateResults> über das Suchergebnis.

Innerhalb von <fss:isCategory> sind <fss:isTrue> und <fss:isFalse> zu verwenden. Der Inhalt von <fss:isTrue> wird genau dann ausgegeben, wenn eine der angegebenen Kategorien (Attribut "categories") einer Kategorie des Dokumentes (Schnittmenge) entspricht. Im Gegensatz dazu, wird der Inhalt von <fss:isFalse> genau dann angezeigt, wenn keine Kategorie mit einer Kategorie des Dokumentes übereinstimmt.

Attribute:

Attribut	Erwarteter Wert	Pflichtparameter
categories	Kommaseparierte Liste von Kategorien für den Vergleich mit den Kategorien des Suchergebnisses.	Ja

Beispiel:

Das erste Beispiel demonstriert die Anwendung des JSP-Tags, um das Suchformular mit den Parametern der aktuellen Suche vor zu belegen:

```
<fss:getSearchDetails>
  <input type="text" name="query" value="<%= query %>" />
  | PDF <input type="checkbox" name="pdf"
    <fss:isCategory categories="pdf">
      <fss:isTrue>checked</fss:isTrue>
    </fss:isCategory>
  />
</fss:getSearchDetails>
```

Im zweiten Beispiel dient das JSP-Tag zur Darstellung eines Symbols vor den Ergebnissen einer bestimmten Kategorie:

```
<fss:iterateResults>
  <fss:isCategory cagegories="pdf">
    <fss:isTrue><img src="/icons/pdf.gif"></fss:isTrue>
  </fss:isCategory>
</fss:iterateResults>
```

5.6.2 Sprachüberprüfung (<fss:isLocale>)

Das Tag <fss:isCategory> kann innerhalb der Tags <fss:getSearchDetails> (siehe Kapitel 5.5.1, Seite 70) und <fss:iterateResults> (siehe Kapitel 5.5.2, Seite 72) verwendet werden.

Bei der Verwendung innerhalb von `<fss:getSearchDetails>` gibt `<fss:isLocale>` Auskunft über die verwendeten Suchparameter und innerhalb von `<fss:iterateResults>` über das Suchergebnis.

Innerhalb von `<fss:isLocale>` sind `<fss:isTrue>` und `<fss:isFalse>` zu verwenden. Der Inhalt von `<fss:isTrue>` wird genau dann ausgegeben, wenn eine der angegebenen Sprachkürzel (Attribut "locales") einem Sprachkürzel des Dokumentes (Schnittmenge) entspricht. Im Gegensatz dazu, wird der Inhalt von `<fss:isFalse>` genau dann angezeigt, wenn kein Sprachkürzel mit einem Sprachkürzel des Dokumentes übereinstimmt.

Attribute:

Attribut	Erwarteter Wert	Pflichtparameter
locales	Kommaseparierte Liste von Sprachkürzeln für den Vergleich mit den Sprachkürzeln des Suchergebnisses.	Ja

Beispiel:

Das erste Beispiel demonstriert die Anwendung des JSP-Tags, um das Suchformular mit den Parametern der aktuellen Suche vor zu belegen:

```
<fss:getSearchDetails>
  <input type="text" name="query" value="<%= query %>" />
  | PDF <input type="checkbox" name="german"
    <fss:isLocale locales="de">
      <fss:isTrue>checked</fss:isTrue>
    </fss:isLocale>
  />
</fss:getSearchDetails>
```

Im zweiten Beispiel dient das JSP-Tag zur Darstellung eines Symbols vor den Ergebnissen einer bestimmten Sprache:

```
<fss:iterateResults>
  <fss:isLocale locales="en">
    <fss:isTrue><img src="/icons/en.gif"></fss:isTrue>
  </fss:isLocale>
</fss:iterateResults>
```

5.7 Hervorhebungen

Für eine Suche werden üblicherweise Suchbegriffe angegeben. Wurde der Suchbegriff innerhalb eines Dokuments gefunden, so wird das Dokument und die Fundstelle als Ergebnis zurückgeliefert. Standardmäßig wird die Fundstelle nicht hervorgehoben. Eine Hervorhebung der Fundstelle kann mit dem Tag `<fss:highlighter>` erzielt werden.

5.7.1 Fundstellen hervorheben (<fss:highlighter>)

Mit dem Tag `<fss:hightlighter>` werden die Fundstellen innerhalb eines Textes hervorgehoben. Darüber hinaus, kann die Ausgabe auf eine maximale Anzahl von Zeichen beschränkt werden. Anstelle des Suchbegriffes kann auch eine Liste von Wörtern angegeben werden, die in den Dokumenten hervorgehoben werden sollen.

Attribute:

Attribut	Erwarteter Wert	Pflichtparameter
on	Angabe eines (HTML-)Quelltexts, der vor der Fundstelle eingefügt werden soll. Standardmäßig wird "" eingefügt.	Nein
off	Angabe eines (HTML-)Quelltexts, der hinter der Fundstelle eingefügt werden soll. Standardmäßig wird "" eingefügt.	Nein
maxSize	Mit diesem Attribut kann festgelegt werden, wie viele Zeichen (inkl. der Fundstelle) maximal ausgegeben werden sollen. Als Wert wird eine Ganzzahl erwartet. Für längere Textpassagen ist eine Beschränkung (z. B. auf 250 Zeichen) sinnvoll.	Nein
words	Kommaseparierte Liste von Wörtern, die anstelle des Suchtextes hervorgehoben werden sollen, z. B. `free,the`	Nein

Beispiel: Im Beispiel wird die Fundstelle im Titel fett hervorgehoben und mit der Hintergrundfarbe #DDDDDD (Grauton) hinterlegt:

```
<fss:highlighter on="<b><span style='background-color:#DDDDDD'>"
off="</span></b>"><%= title %></fss:highlighter>
```

In diesem Beispiel wird die Fundstelle ebenfalls im Textausschnitt fett hervorgehoben und mit der Hintergrundfarbe #DDDDDD (Grauton) hinterlegt. Zusätzlich wird die Anzahl der ausgegebenen Zeichen auf 250 beschränkt:

```
<fss:highlighter maxSize="250" on="<b><span style='background-
color:#DDDDDD'>" off="</span></b>"><%= content
%></fss:highlighter>
```

Hier werden die Wörter "the" und "free" im Titel fett hervorgehoben und mit der Hintergrundfarbe #DDDDDD (Grauton) hinterlegt:

```
<fss:highlighter words="the,free" on="<b><span style='background-
color:#DDDDDD'>" off="</span></b>"><%= title %></fss:highlighter>.
```

5.8 Navigation

Bei einer Suche werden die Ergebnisse auf virtuelle Seiten verteilt. Zur Navigation zwischen den einzelnen, virtuellen Seiten bietet FirstSpirit Search Tags zur Erzeugung einer Navigationsleiste an.

Eine Navigationsleiste besteht aus drei Bereichen:

- Erste virtuelle Seite bzw. obere Navigationsgrenze (fss:navTop)
- Teilbereich bzw. Navigationsausschnitt virtueller Seiten, ausgehend von der aktuellen Seite (fss:navFrame)
- Letzte virtuelle Seite bzw. untere Navigationsgrenze (fss:navBottom)

Beispiele von Navigationsleisten (die aktuelle Seite ist jeweils fett hervorgehoben):

- Anzeige der ersten und letzten virtuellen Seite und des Teilbereichs:
  [1] … [6] [7] **[8]** [9] [10] … [21]
- Anzeige der letzten, virtuellen Seite und des Teilbereichs
  (die erste Seite wird im Teilbereich angezeigt):
  [1] **[2]** [3] [4] [5] … [21]
- Anzeige der ersten, virtuellen Seite und des Teilbereichs
  (die letzte Seite wird im Teilbereich angezeigt):
  [1] … [17] [18] [19] **[20]** [21]

- Anzeige des Teilbereichs
  (die erste und die letzte Seite werden im Teilbereich angezeigt):
  [1] [2] [3] [4] [5]

5.8.1 Navigationsleiste (<fss:navigation>)

Das Tag `<fss:navigation>` erzeugt eine Navigationsleiste. Innerhalb des Tags `<fss:navigation>` können die Tags `<fss:navTop>`, `<fss:navFrame>` und `<fss:navBottom>` verwendet werden.

Über den Parameter "frameSize" kann die maximale Anzahl der virtuellen Seiten im Teilbereich (`<fss:navFrame>`) definiert werden.

Attribute:

Attribut	Erwarteter Wert	Pflichtparameter
frameSize	Anzahl der in <fss:navFrame> darzustellenden virtuelle Seiten (Ganzzahl)	Ja

Beispiel:

```
...
<fss:navigation frameSize="3">

  <fss:navTop>
    ...
  </fss:navTop>

  <fss:navFrame>
    ...
  </fss:navFrame>

  <fss:navBottom>
    ...
  </fss:navBottom>

</fss:navigation>
...
```

5.8.2 Erste / Letzte virtuelle Seite (<fss:navTop>, <fss:navBottom>)

Die beiden JSP-Tags `<fss:navTop>` und `<fss:navBottom>` stehen innerhalb des Tags `<fss:navigation>` zur Verfügung.

Mit den Tags `<fss:isTrue>` und `<fss:isFalse>` kann überprüft werden, ob die erste (`<fss:navTop>`) oder letzte (`<fss:navBottom>`) virtuelle Seite nicht innerhalb des Teilbereiches (`<fss:navFrame>`) dargestellt wird. Der Inhalt von `<fss:isTrue>` wird angezeigt, wenn die erste bzw. letzte virtuelle Seite nicht im Teilbereich dargestellt wird. So kann beispielsweise eine doppelte Ausgabe verhindert werden.

```
...
<fss:navTop>
  <fss:isTrue>
    ...
  </fss:isTrue>
  <fss:isFalse>
    ...
  </fss:isFalse>
</fss:navTop>
...
```

Innerhalb der beiden Tags stehen die Variablen "urlPageNo" und "url" zur Verfügung. Mit "urlPageNo" kann die Nummer und mit "url" der URL der virtuellen Zielseite ausgegeben werden. Zusätzlich kann die Anzahl der virtuellen Seiten mit "urlPageSize" ausgegeben werden.

Variablen:

Variable	Bedeutung	Rückgabedatentyp
urlPageNo	Nummer der virtuellen Zielseite.	Integer
url	URL der virtuellen Zielseite (inkl. Parameter).	String
urlPageSize	Anzahl der virtuellen Seiten.	Integer

Beispiel:

```
...
<fss:navigation frameSize="3">

  <fss:navTop>
    <fss:isTrue>
      <a href="<%= url %>">[<%= urlPageNo %>]</a> ...
    </fss:isTrue>
  </fss:navTop>

  <fss:navFrame>
    <fss:isTrue>
      <a href="<%= url %>">[<%= urlPageNo %>]</a>
    </fss:isTrue>
    <fss:isFalse>
      <b>[<%= urlPageNo %>]</b>
    </fss:isFalse>
  </fss:navFrame>

  <fss:navBottom>
    <fss:isTrue>
      ... <a href="<%= url %>">[<%= urlPageNo %>]</a>
    </fss:isTrue>
  </fss:navBottom>

</fss:navigation>

...
```

5.8.3 Teilbereich aller virtuellen Seiten (<fss:navFrame>)

Mit dem Tag `<fss:navFrame>`, das innerhalb des Tags `<fss:navigation>` zur Verfügung steht, werden alle virtuellen Seiten im Teilbereich dargestellt.

Die Anzahl der anzuzeigenden Seiten wird durch das Attribut "frameSize" festgelegt.

Mit dem Tag `<fss:isTrue>` kann überprüft werden, ob die im Teilbereich darzustellende Seite, nicht der aktuellen Seite entspricht. Der Inhalt von `<fss:isTrue>` wird angezeigt, wenn die vom Teilbereich anzuzeigende Seite, nicht die aktuelle Seite ist.

```
...
<fss:navFrame>
  <fss:isTrue>
    ...
  </fss:isTrue>
  <fss:isFalse>
    ...
  </fss:isFalse>
</fss:navFrame>
...
```

Innerhalb des Tags sind die beiden Variablen "navPageNo" und "navUrl" verfügbar. Mit "navPageNo" kann die Nummer und mit "navUrl" der URL, der vom Teilbereich anzuzeigenden, virtuellen Seite ausgegeben werden. Zusätzlich kann die Anzahl der virtuellen Seiten mit "navPageSize" ausgegeben werden.

Variablen:

Variable	Bedeutung	Rückgabedatentyp
urlPageNo	Nummer der virtuelle Zielseite	Integer
url	URL der virtuellen Zielseite (inkl. Parameter)	String
urlPageSize	Anzahl der virtuellen Seiten	Integer

Beispiel:

```
...
<fss:navigation frameSize="3">

  <fss:navTop>
    <fss:isTrue>
      <a href="<%= url %>">[<%= urlPageNo %>]</a> ...
    </fss:isTrue>
  </fss:navTop>

  <fss:navFrame>
    <fss:isTrue>
      <a href="<%= url %>">[<%= urlPageNo %>]</a>
    </fss:isTrue>
    <fss:isFalse>
      <b>[<%= urlPageNo %>]</b>
    </fss:isFalse>
  </fss:navFrame>

  <fss:navBottom>
    <fss:isTrue>
      ... <a href="${navUrl}">[${navPageNo}]</a>
    </fss:isTrue>
  </fss:navBottom>

</fss:navigation>
...
```

5.8.4 Verweise auf bestimmte Seiten (<fss:url>)

Mit `<fss:url>` kann ein Verweis auf eine bestimmte virtuelle Seite erzeugt werden. Die Auswahl kann dabei relativ zur aktuellen Seite oder absolut zur ersten bzw. letzten Seite geschehen. Darüber hinaus kann mit dem Verweis, die Anzahl der auf einer Seite dargestellten Ergebnisse ("pageSize") verändert werden. Mit dem Tag `<fss:isTrue>` kann geprüft werden, ob es sich bei der angegebenen Seite, um eine gültige Seite handelt. Ist dies der Fall, wird der Inhalt von `<fss:isTrue>` angezeigt. Wenn nicht, wird der Inhalt von `<fss:isFalse>` angezeigt, sofern das Tag angegeben wurde.

Parameter:

Attribut	Erwarteter Wert	Pflichtparameter
jump	Relativer Sprung zu einer virtuellen Seite. Mögliche Werte: ▪ `fp`: erste Seite ▪ `pp`: vorherige Seite ▪ `np`: nächste Seite ▪ `lp`: letzte Seite	Nein
pageNo	Absoluter Wert der virtuellen Zielseite. Als Wert wird eine Ganzzahl erwartet	Nein
pageSize	Ändert den Wert, der pro virtuelle Seite dargestellten Ergebnisse.	Nein

Variablen:

Variable	Bedeutung	Rückgabedatentyp
urlPageNo	Nummer der virtuellen Zielseite.	Integer
url	URL der virtuellen Zielseite (inkl. Parameter).	String
urlPageSize	Anzahl der virtuellen Seiten.	Integer

Beispiel:

Das folgende Beispiel zeigt eine einfache Navigationsleiste mit vier Verweisen zur ersten, vorherigen, nächsten und letzten, virtuellen Seite:

```
<fss:url jump="fp"><a href="<%= url %>">[|<]</a></fss:url>
<fss:url jump="pp"><a href="<%= url %>">[<-]</a></fss:url>
<fss:url jump="np"><a href="<%= url %>">[->]</a></fss:url>
<fss:url jump="lp"><a href="<%= url %>">[>|]</a></fss:url>
```

Das nächste Beispiel demonstriert die Änderung der anzuzeigenden Ergebnisse pro virtuelle Seite:

```
<fss:url>
  <form action="<%= url %>" method="GET">
    <select name="pageSize">
      <option value="10">10 Einträge/Seite</option>
      <option value="25">25 Einträge/Seite</option>
      <option value="50">50 Einträge/Seite</option>
    </select>

    ...
  </form>
</fss:url>
```

```
<fss:url jump="2000">
  <fss:isTrue>
    <a href="<%= url %>">2000</a>
  </fss:isTrue>
  <fss:isFalse>
    Verweis ungültig
  </fss:isFalse>
</fss:url>
```

6 Suchanfragen (Search-Servlet)

Mit dem Search-Servlet kann eine Suchanfrage durchgeführt werden. Das Ergebnis einer solchen Suchanfrage wird in der Benutzersession gespeichert und kann mithilfe des Tags `<fss:iterateResults>` ausgegeben werden (siehe Kapitel 5.5.2, Seite 72).

Der Aufruf des Search-Servlets erfolgt mit:

```
"<%= request.getContextPath() %>/do.search"
```

In einem HTML-Formular wird das Tag <form> verwendet:

```
<form method="post"
      action="<%= request.getContextPath() %>/do.search">
  ...
</form>
```

Für die HTTP-Übertragungsmethode ist "post" zu verwenden, da die Länge der Parameternamen und -werte sehr lang sein kann!

Das Search-Servlet verfügt über den **Request-Parameter** "pageSize", um Ergebnisse auf verschiedene, virtuelle Seiten zu verteilen. Der Parameter erwartet als Wert die Anzahl der Ergebnisse, die maximal auf einer virtuellen Seite dargestellt werden sollen. Wird der Parameter nicht angegeben, wird der Standardwert 10 (Ergebnisse pro virtuelle Seite) verwendet.

```
<input type="hidden" name="pageSize" value="1" />
```

Für die Darstellung einer Navigationsleiste (zur Navigation durch die einzelnen virtuellen Seiten) stehen die folgenden Tags zur Verfügung:

- <fss:navigation> (siehe Kapitel 5.8.1, Seite 79)
- <fss:navTop> (siehe Kapitel 5.8.2, Seite 80)
- <fss:navFrame> (siehe Kapitel 5.8.3, Seite 81)
- <fss:navBottom> (siehe Kapitel 5.8.2, Seite 80)
- <fss:url> (siehe Kapitel 5.8.4, Seite 83)

Mit dem Pflicht-Request-Parameter "query" können ein oder mehrere Suchbegriffe übergeben werden. Standardmäßig werden zwei Suchbegriffe "oder"-verknüpft. Die Form der Verknüpfung der Suchbegriffe kann durch den Request-Parameter "op"

geändert werden. Für diesen Request-Parameter sind die Werte "and" ("und"-Verknüpfung), "or" ("oder"-Verknüpfung) und "not" (Negation des Suchbegriffes) verfügbar.

Beispiel:

```
<input type="hidden" name="query" value="city" />
<input type="hidden" name="op" value="and" />
<input type="hidden" name="query" value="tree" />
<input type="hidden" name="op" value="not" />
<input type="hidden" name="query" value="leaf" />
<input type="hidden" name="query" value="sun" />
```

Hier wird nach "city" und "tree" und nicht "leaf" oder "sun" gesucht.

Ab FirstSpirit Version 4.2 Release 2 kann der Request-Parameter "wildcardSearch" verwendet werden, um nur nach Wortanfängen (rechtsseitige Trunkierung) oder nach Zeichenfolgen (rechts- und linksseitige Trunkierung) suchen zu können, wobei als Trunkierungszeichen `*` verwendet wird. Für diesen Parameter sind die folgenden Werte verfügbar:

- "contains": Es wird eine Platzhalter-Suche durchgeführt, wobei der vom Benutzer eingegebene Suchtext automatisch um einen Stern-Operator (`*`) ergänzt wird, und zwar am Anfang **und** Ende (rechts- und linksseitige Trunkierung). D.h., eine Suche nach *list* findet sowohl *List*, *Liste* als auch *Contentlist*.
  Beispiel: `wildcardSearch="contains"`
- "startsWith": Es wird eine Suche nach dem Wortanfang durchgeführt, wobei der vom Benutzer eingegebene Suchtext automatisch um einen Stern-Operator (`*`) ergänzt wird, und zwar nur am Ende (rechtsseitige Trunkierung). D.h., eine Suche nach *list* findet sowohl *List* als auch *Liste*, nicht aber *Contentlist*.
  Beispiel: `wildcardSearch="startsWith"`
- "none": Es wird nach genau dem eingegebenen Suchbegriff gesucht. D.h., eine Suche nach *list* findet nur List, nicht aber *Liste* oder *Contentlist*. Dies ist das Standard-Verhalten, wenn der Parameter nicht gesetzt wird.
  Beispiel: `wildcardSearch="none"`

*Verwendet der Benutzer Anführungszeichen zur Phrasensuche (z. B. "List") oder gibt er selbst den Platzhalter * an (z. B. *List), wird keine automatische Trunkierung durchgeführt.*

Bei der Verwendung des Parameters sollte berücksichtigt werden, dass eine Suche mit `wildcardSearch="contains"` aufwändiger und damit langsamer ist als eine Suche mit `wildcardSearch="startsWith"`.

Der Parameter kann in der Datei "web.xml" als Konfigurationsparameter als auch als Request-Parameter in einem HTML-Formular verwendet werden. Wird der Parameter sowohl in der Datei "web.xml" als auch als Request-Parameter definiert, wird die Definition in der Datei "web.xml" vorrangig verwendet.

Neben der Angabe der Suchbegriffe und ihrer logischen Verknüpfung kann die anfängliche Anzahl der zu suchenden Ergebnisse über den Request-Parameter "initialSize" beschränkt werden. Wird dieser Parameter nicht angegeben, so werden initial 10 Ergebnisse angefordert.

Die Maximalanzahl der Suchergebnisse kann über den Request-Parameter "maxResults" beschränkt werden. Werden mehr Suchergebnisse ermittelt als die angegebene Maximalanzahl, so werden nur die ersten Suchergebnisse zurückgeliefert, bis die Maximalanzahl erreicht wird.

Zusätzlich kann über den Request-Parameter "orderBy" festgelegt werden, welches Feld für die Sortierung verwendet werden soll. Der Request-Parameter "ascending" definiert, ob die Sortierung auf- oder absteigend erfolgen soll.

Soll nach einer erfolgreichen Suche eine abweichende Ergebnisseite angezeigt werden, so kann der Request-Parameter "resultsURL" genutzt werden. Wurden keine Ergebnisse gefunden, so kann über den Request-Parameter "noResultsURL" ebenfalls eine separate Seite angesteuert werden.

Die Suche kann zudem auf ein bestimmtes Sprachkürzel (über den Request-Parameter "locale") oder bestimmte Kategorien (über den Request-Parameter "categories") eingegrenzt werden. Wird der Request-Parameter "categories" mehrfach angegeben, so werden die angegebenen Kategorien "oder"-verknüpft. Ist der Request-Parameter "categories" gesetzt, aber wurde kein Wert angegeben (`categories=`), wird **bis FirstSpirit Version 4.2 einschließlich** eine Suche nach einer leeren Kategorie (`""`) durchgeführt, **ab Version 4.2R2** wird ein solcher Parameter bei der Suche ignoriert.

In der Datei **web.xml** wird über die Eingabefelder "Server-Name" und "Engine-

Name" in der Server- und Projektkonfiguration der standardmäßig zu verwendende Such-Server und die Such-Engine vorgegeben (siehe Kapitel 3.3, Seite 12):

```
<servlet>
  <servlet-name>fss-Search</servlet-name>
  <servlet-
class>de.espirit.firstspirit.opt.search.web.SearchServlet</servlet
-class>
  <init-param>
    <param-name>serverURL</param-name>
    <param-value>fssServer</param-value>
  </init-param>
  <init-param>
    <param-name>searchEngine</param-name>
    <param-value>fssEngine</param-value>
  </init-param>
</servlet>
```

Hier wird der als Such-Server ("serverURL") "fssServer" und als Such-Engine ("searchEngine") "fssEngine" konfiguriert.

Die Such-Engine kann auch mit dem Request-Parameter "se" für eine Suchanfrage angegeben werden. In diesem Fall muss der Parameter "searchEngine" aus der Datei "web.xml" entfernt werden.

Aus Sicherheitsgründen können die Werte aus der Datei "web.xml" durch Request-Parameter nicht überschrieben werden!

Zusätzlich kann die Anzahl der möglichen Suchanfragen (pro HTTP-Session) über den Request-Parameter "singleton" auf eine beschränkt werden.

Beispiel:

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ taglib uri="fs-search" prefix="fss" %>
...
<fss:getSearchDetails>
  <form method="post"
        action="<%= request.getContextPath() %>/do.search">
    Suchtext:
    <input type="text"
           name="query"
           value="<%= (query != null ? query : "word") %>"/>
    <input type="hidden" name="initialSize" value="10" />
    <input type="hidden" name="pageSize" value="5" />
    <input type="hidden" name="maxResults" value="500" />
    <input type="hidden" name="singleton" value="true" />
    <input type="hidden"
           name="resultsURL"
```

```
            value="$CMS_REF(pageref:"homepage", abs:1)$" />
    <input type="submit" value="Suchen" />
  </form>
</fss:getSearchDetails>
...
```

In der Datei "web.xml" kann **ab FirstSpirit Version 4.2 Release 2** auch der Parameter "wildcardSearch" verwendet werden. Wird der Parameter sowohl in der Datei "web.xml" als auch als Request-Parameter definiert, wird die Definition in der Datei "web.xml" vorrangig verwendet.

Eine Liste der verfügbaren Request-Parameter kann der nachfolgenden Tabelle entnommen werden:

Konfigurationsparameter ("web.xml"):

Parameter	Erwarteter Wert
formEncoding	Mit diesem Parameter kann das Encoding festgelegt werden, mit dem die Formulardaten verarbeitet werden sollen (z. B. "UTF-8").
searchEngine	Angabe der Such-Engine. Wenn der Parameter "searchEngine" aus der Datei "web.xml" entfernt wird, ist der Request-Parameter "se" zu verwenden.

sessionSearchProperties	Kommaseparierte Liste von Attributen, die automatisch aus der HTTP-Session an die Suche übergeben wird. Hierdurch können sicherheitsrelevante Informationen, z. B. die Rollen bzw. Gruppen des aktuellen Benutzers, übergeben werden, ohne dass Laufzeitparameter die Werte beeinflussen können.
	
	Beispiel:
	`<init-param>`
	`<param-name>sessionSearchProperties</param-name>`
	`<param-value>FIRSTpersonalisation.usergroups</param-value>`
	`</init-param>`
	
	Hier wird das Session-Attribut: "FIRSTpersonalisation.usergroups", welches von FirstSpirit Personalisation zur Verfügung gestellt wird, an die Suche übergeben.

serverURL	Angabe der URL des Such-Servers, der für die Suche verwendet werden soll (siehe Kapitel 4.5.1.3, Seite 24, Parameter "engineURL"). Bei diesem Parameter handelt es sich um einen Pflichtparameter.
wildcardSearch (ab FirstSpirit Version 4.2 Release 2)	Dieser Parameter kann verwendet werden, um nur nach Wortanfängen (rechtsseitige Trunkierung) oder nach Zeichenfolgen (rechts- und linksseitige Trunkierung) suchen zu können. Mit "contains" wird eine Platzhalter-Suche durchgeführt, wobei der vom Benutzer eingegebene Suchtext automatisch um einen Stern-Operator (*) ergänzt wird, und zwar am Anfang und Ende (rechts- und linksseitige Trunkierung). Mit "startsWith" wird eine Suche nach dem Wortanfang durchgeführt, wobei der vom Benutzer eingegebene Suchtext automatisch um einen Stern-Operator (*) ergänzt wird, und zwar nur am Ende (rechtsseitige Trunkierung). Mit "none" wird nach genau dem eingegebenen Suchbegriff gesucht. Dies ist das Standard-Verhalten, wenn der Parameter nicht gesetzt wird.

Request-Parameter:

Parameter	Erwarteter Wert
ascending	Ergebnisse alphabetisch aufsteigend (Wert: "true") oder absteigend (Wert: "false") sortieren; Standardwert: "false".
categories	Angabe einer oder mehrerer Kategorien (= mehrfache Angabe des Request-Parameters), auf die die Suche eingegrenzt werden soll. Bei der Angabe mehrerer Request-Parameter werden die Werte "oder"-verknüpft.
errorURL	Angabe eines URLs, der angezeigt werden soll, wenn bei der Suche ein Fehler aufgetreten ist. Wird kein Wert angegeben, so wird der Wert des Parameters "noResultsURL" verwendet.
field	Wird dieser Parameter angegeben, so bezieht sich die Suche auf das angegebene Feld in den Ergebnissen (z. B. "title").
initialSize	Anzahl der Ergebnisse, die zu Beginn angefordert werden. Die Suchmaschine wartet solange, bis entweder eine entsprechende Anzahl Ergebnisse gefunden wird oder die Suche beendet ist.
locale	Angabe eines Sprachkürzels (Sprache), auf das die Suche eingegrenzt werden soll.

Parameter	Erwarteter Wert
maxResults	Anzahl der Ergebnisse, die maximal gefunden werden sollen. Werden mehr Ergebnisse gefunden, als angegeben, so bricht die Suche ab und zeigt nur die ersten Ergebnisse, bis zum Erreichen der Maximalanzahl, an.
noResultsURL	Dieser URL wird aufgerufen, wenn die Suche keine Ergebnisse ermitteln konnte. Wird für den Parameter "noResultsURL" kein Wert angegeben, so wird der Wert des Parameters "resultsURL" verwendet.
op	Mit diesem Parameter wird angegeben, wie die Suchbegriffe (siehe Parameter "query") miteinander verknüpft werden. Der Parameter "op" kann mehrfach angegeben werden. Es werden die Parameter in der angegebenen Reihenfolge für die Verknüpfung berücksichtigt. Als Werte stehen zur Verfügung: "and" ("und"-Verknüpfung), "or" ("oder"-Verknüpfung) und "not" (Negation); Standardwert: "or".
orderBy	Angabe der Feldes, welches für die Sortierung herangezogen werden soll: "author", "date", "score", "size" oder "title"; Standardwert: "score".
pageSize	Anzahl der Elemente, die pro Seite dargestellt werden sollen; Standardwert: "10".
query	Suchwort bzw. -text. Bei mehrfacher Angabe des Parameters werden die Suchbegriffe mit dem Wert aus dem Parameter "op" verknüpft.
regexPattern	Mit diesem Request-Parameter können eine oder mehrere reguläre Ausdrücke (=mehrfache Verwendung des Request-Parameters) in Java-Syntax an die Suche übergeben werden. Die angegebenen Ausdrücke, werden der Suche als "oder"-verknüpfte Filter beigefügt.
resultsURL	Dieser URL wird aufgerufen, wenn die Suche fehlerfrei war und Ergebnisse ermittelt werden konnten. Wird der Parameter nicht angegeben, wird die aufrufende Seite erneut aufgerufen.
se	Synonym für den Parameter "searchEngine". Der Request-Parameter kann nur verwendet werden, wenn der Parameter "searchEngine" aus der Datei "web.xml" entfernt wird.

Parameter	Erwarteter Wert
searchProperties	Kommaseparierte Liste von Request-Parameternamen, die automatisch an die Suche übergeben wird.
singleton	Beschränkung auf eine gleichzeitige Suchanfrage in einer HTTP-Session (Wert: "true"). Eine unbeschränkte Anzahl von Suchanfragen kann mit dem Wert "false" erzielt werden; Standardwert: "false".
wildcardSearch (ab FirstSpirit Version 4.2 Release 2)	siehe Dokumentation zu "wildcardSearch" als Konfigurationsparameter ("web.xml"); wird der Parameter sowohl in der Datei web.xml als auch als Request-Parameter definiert, wird die Definition in der Datei web.xml vorrangig verwendet

Um Open-Redirect-Angriffe zu unterbinden, sind in Projekten, die das Modul FirstSpirit Search verwenden, ab ***FirstSpirit Version 4.2R4*** *externe Redirects verboten. D. h. es sind nur relative und absolute URLs möglich (z. B.*
`start.jsp`
`../bereich/index.html`
`../bereich/search.jsp`*)*
Ausnahme: Redirects auf denselben Host oder einen Host in der gleichen Domain sind zulässig. Soll zu einer externen URL weitergeleitet werden, muss dafür eine Weiterleitungsseite erstellt werden.

7 Beispiele

7.1 Kategorien- und Sprachensimulation (URL)

Im Beispiel werden drei Services definiert:

- ein Logging-Service
- ein Server-Service und
- ein Re-Indizierungs-Service.

Im Re-Indizierungs-Service wird, täglich um 06:00 Uhr, eine Aktualisierung des Indexes vorgenommen.

Im Server-Service wird eine Spider-Engine definiert, die auf Basis des Generierungsverzeichnisses eines Projektes mit zwei Sprachen und zwei Präsentationskanäle, einen Index erzeugt.

Die Sprachen ("de", und "en") werden über einen Mehrsprachigkeitsfilter und die Kategorien ("web" und "print") über einen Kategorienfilter simuliert.

```
<?xml version="1.0" encoding="UTF-8"?>
<service
class="de.espirit.firstspirit.opt.search.service.proxy.MultiServic
eProxy">
  <attribute name="services" type="list">

    <service
class="de.espirit.firstspirit.opt.search.service.adapter.Log4jServ
ice">
      <attribute name="log4j.rootCategory">DEBUG, file</attribute>
      <attribute
name="log4j.appender.file">org.apache.log4j.RollingFileAppender</a
ttribute>
      <attribute
name="log4j.appender.file.File">E:/FirstSpirit/log/fs-
search.log</attribute>
      <attribute
name="log4j.appender.file.MaxFileSize">5MB</attribute>
      <attribute
name="log4j.appender.file.MaxBackupIndex">5</attribute>
      <attribute
name="log4j.appender.file.layout">org.apache.log4j.PatternLayout</
attribute>
      <attribute
name="log4j.appender.file.layout.ConversionPattern">%-5p %d (%c)
%m%n</attribute>
    </service>

    <service
```

```
class="de.espirit.firstspirit.opt.search.service.adapter.ServerSer
vice">
      <server
class="de.espirit.firstspirit.opt.search.server.SimpleServer">
        <engine name="fssEngine"
class="de.espirit.firstspirit.opt.search.engine.proxy.MonitorEngin
eProxy">

          <engine
class="de.espirit.firstspirit.opt.search.engine.proxy.FilterCatego
rizeEngineProxy">

            <engine
class="de.espirit.firstspirit.opt.search.engine.proxy.FilterLocali
zeEngineProxy">

              <engine
class="de.espirit.firstspirit.opt.search.engine.spider.SpiderEngin
e">

                <attribute name="urls" type="list">

<attribute>http://localhost/fs4staging_12345/23456/de/dokumentatio
n/documentation.html</attribute>

<attribute>http://localhost/fs4staging_12345/23456/de_1/dokumentat
ion/documentation.html</attribute>

<attribute>http://localhost/fs4staging_12345/23456/en/dokumentatio
n/documentation.html</attribute>

<attribute>http://localhost/fs4staging_12345/23456/en_1/dokumentat
ion/documentation.html</attribute>
                </attribute>
                <attribute
name="index">E:/FirstSpirit/web/fs4staging_12345/WEB-
INF/lucene.index</attribute>
                <attribute name="maxThreads">1</attribute>
                <attribute name="threadPriority">1</attribute>
                <attribute
name="maxFieldLength">200000</attribute>
                <attribute name="maxContentLength">100</attribute>
                <attribute name="allowed" type="list">
                  <attribute
class="de.espirit.firstspirit.opt.search.engine.spider.link.RegexW
ebLinkFilter">/de/</attribute>
                  <attribute
class="de.espirit.firstspirit.opt.search.engine.spider.link.RegexW
ebLinkFilter">/de_1/</attribute>
                  <attribute
class="de.espirit.firstspirit.opt.search.engine.spider.link.RegexW
ebLinkFilter">/en/</attribute>
                  <attribute
class="de.espirit.firstspirit.opt.search.engine.spider.link.RegexW
ebLinkFilter">/en_1/</attribute>
                </attribute>
                <attribute name="maxTime">120m</attribute>
```

```
              </engine>

              <attribute name="locales" type="map">
                <filter name="de"
class="de.espirit.firstspirit.opt.search.filter.RegexFilter">
                  <attribute name="property">url</attribute>
                  <attribute
name="pattern">(/de/)|(/de_1/)</attribute>
                </filter>
                <filter name="en"
class="de.espirit.firstspirit.opt.search.filter.RegexFilter">
                  <attribute name="property">url</attribute>
                  <attribute
name="pattern">(/en/)|(/en_1/)</attribute>
                </filter>
              </attribute>

            </engine>

            <attribute name="categories" type="map">
              <filter name="web"
class="de.espirit.firstspirit.opt.search.filter.RegexFilter">
                <attribute name="property">url</attribute>
                <attribute
name="pattern">(/de/)|(/en/)</attribute>
              </filter>
              <filter name="print"
class="de.espirit.firstspirit.opt.search.filter.RegexFilter">
                <attribute name="property">url</attribute>
                <attribute
name="pattern">(/de_1/)|(/en_1/)</attribute>
              </filter>
            </attribute>

          </engine>

        </engine>
      </server>
      <attribute name="createRMI">false</attribute>
      <attribute name="bindTo">local</attribute>
      <attribute name="localName">fssServer</attribute>
    </service>

    <service
class="de.espirit.firstspirit.opt.search.service.RebuildIndexTimer
Service">
      <attribute name="engineURL">fssServer[fssEngine]</attribute>
      <attribute name="startTime">6:00</attribute>
      <attribute name="period">24h</attribute>
      <attribute name="startNow">false</attribute>
    </service>

  </attribute>
</service>
```

8 Anhang

8.1 Fehlerbehebung

Die indizierten Felder und Dokumente im Lucene-Index können mit dem Tool Luke - Lucene Index Toolbox[10] betrachtet werden. Das Tool kann unter der angegebenen Adresse heruntergeladen werden (siehe Fußnote) und entweder über die Kommandozeile mit dem Befehl:

```
java -jar lukeall-0.8.1.jar
```

oder über den JNLP-Verweis auf der Website gestartet werden.

Nach dem Öffnen des Indexes, z. B. E:\FIRSTsprit\web\fs4staging_12345\WEB-INF\lucene.index\fig4je34 (siehe Abbildung 8-1), können die indizierten Felder und Dokumente betrachtet und durchsucht werden (siehe Abbildung 8-2 und Abbildung 8-3) (vgl. Kapitel 4.5.4.3 Seite 46).

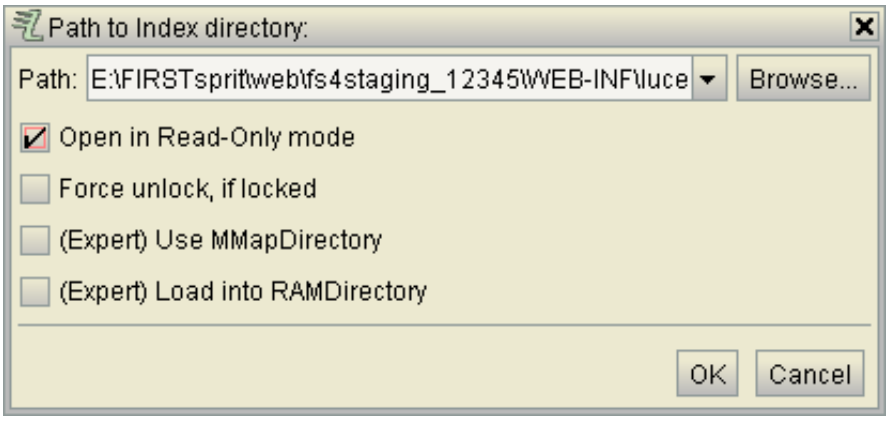

Abbildung 8-1:Öffnen des Lucene-Indexes (Luke)

[10] http://www.getopt.org/luke/

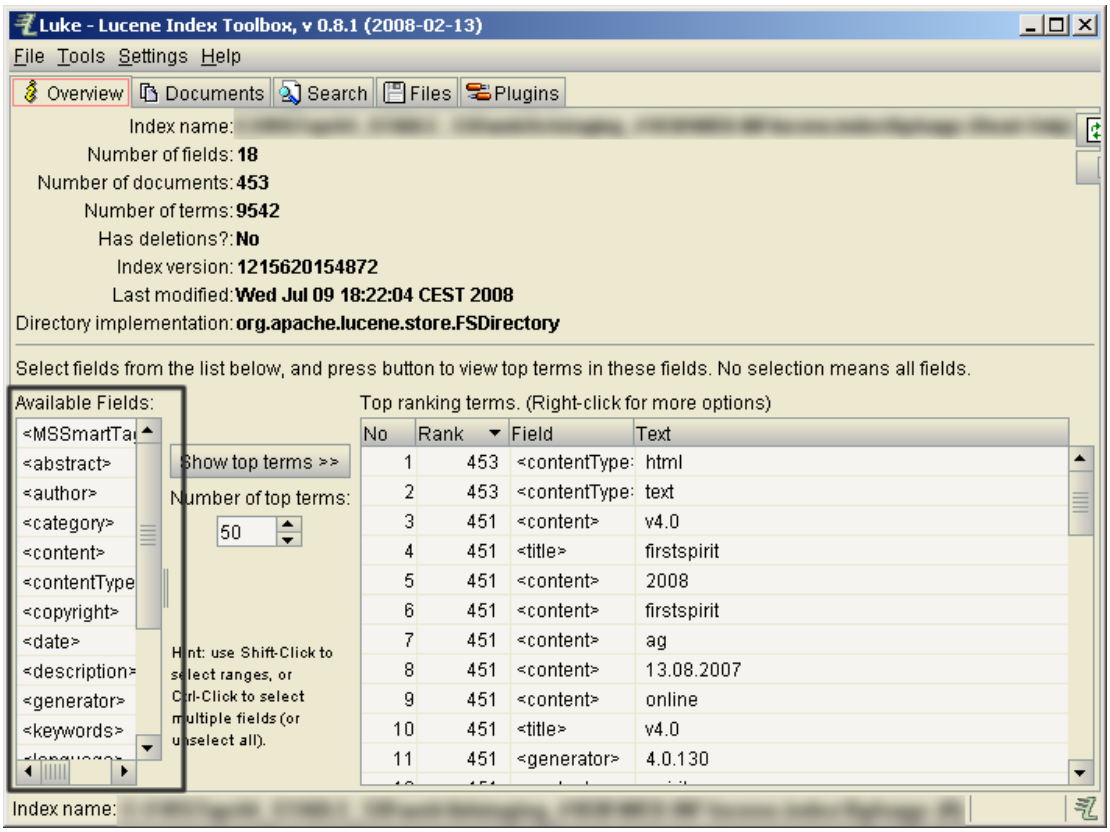

Abbildung 8-2: Ansicht der indizierten Felder (Luke)

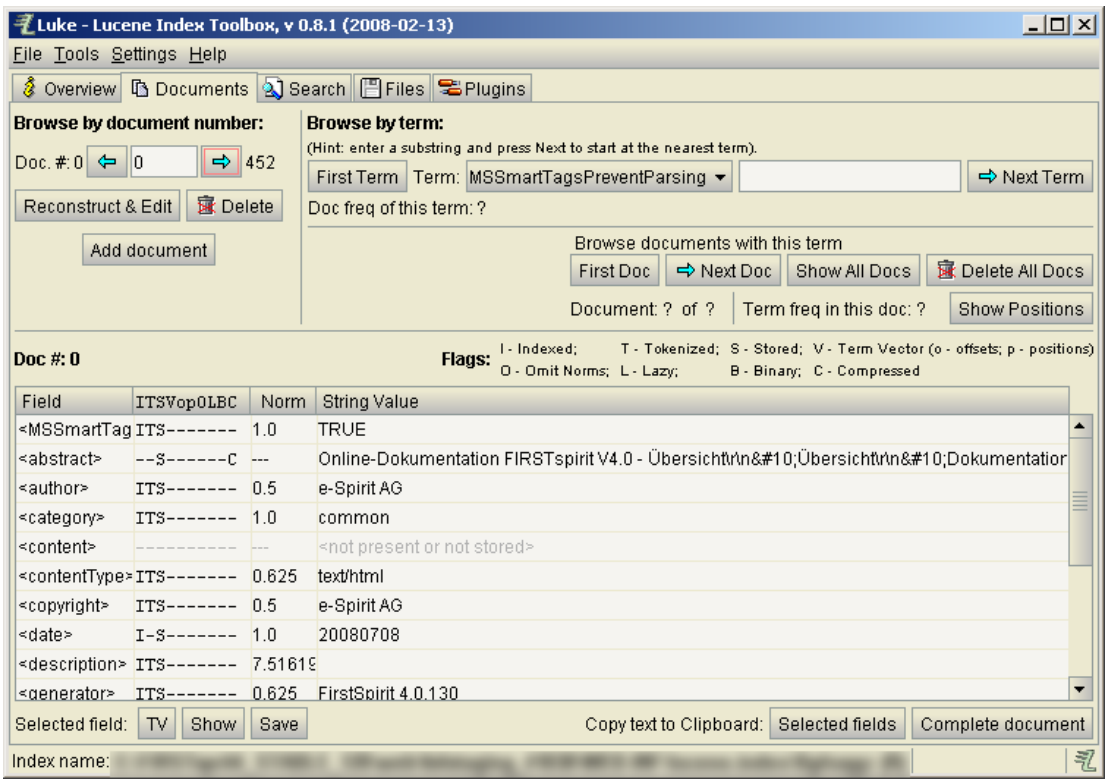

Abbildung 8-3: Ansicht der indizierten Dokumente (Luke)

9 Rechtliche Hinweise

Das Modul "FirstSpirit™ Search" ist ein Produkt der e-Spirit AG, Dortmund, Germany.

Für die Verwendung des Moduls gilt gegenüber dem Anwender nur die mit der e-Spirit AG vereinbarte Lizenz.

Details zu möglicherweise fremden, nicht von der e-Spirit AG hergestellten, eingesetzten Software-Produkten, deren eigenen Lizenzen und gegebenenfalls Aktualisierungs-Informationen, finden Sie auf der Startseite jedes FirstSpirit-Servers im Bereich "Rechtliche Hinweise".