

DIN EN ISO 19142

ICS 35.240.70

**Geoinformation –
Web Feature Service (ISO 19142:2010);
Deutsche Fassung EN ISO 19142:2010**

Geographic information –
Web Feature Service (ISO 19142:2010);
German version EN ISO 19142:2010

Information géographique –
Services d'accès aux entités géographiques par le web (ISO 19142:2010);
Version allemande EN ISO 19142:2010

Gesamtumfang 258 Seiten

Normenausschuss Bauwesen (NABau) im DIN

Nationales Vorwort

Dieses Dokument (EN ISO 19142:2010) wurde vom Technischen Komitee ISO/TC 211 „Geographic information/Geomatics“ in Zusammenarbeit mit dem OGC und dem Technischen Komitee CEN/TC 287 „Geoinformation“, dessen Sekretariat von BSI (Vereinigtes Königreich) gehalten wird, erarbeitet.

Der für die deutsche Mitarbeit zuständige Arbeitsausschuss im DIN Deutsches Institut für Normung e. V. ist der als Spiegelausschuss zum CEN/TC 287 und ISO/TC 211 eingesetzte Arbeitsausschuss NA 005-03-03 AA „Kartographie und Geoinformation“ des Normenausschusses Bauwesen (NABau).

Deutsche Fassung

Geoinformation —
Web Feature Service
(ISO 19142:2010)

Geographic information —
Web Feature Service
(ISO 19142:2010)

Information géographique —
Services d'accès aux entités géographiques par le web
(ISO 19142:2010)

Diese Europäische Norm wurde vom CEN am 14. Dezember 2010 angenommen.

Die CEN-Mitglieder sind gehalten, die CEN/CENELEC-Geschäftsordnung zu erfüllen, in der die Bedingungen festgelegt sind, unter denen dieser Europäischen Norm ohne jede Änderung der Status einer nationalen Norm zu geben ist. Auf dem letzten Stand befindliche Listen dieser nationalen Normen mit ihren bibliographischen Angaben sind beim Management-Zentrum des CEN-CENELEC oder bei jedem CEN-Mitglied auf Anfrage erhältlich.

Diese Europäische Norm besteht in drei offiziellen Fassungen (Deutsch, Englisch, Französisch). Eine Fassung in einer anderen Sprache, die von einem CEN-Mitglied in eigener Verantwortung durch Übersetzung in seine Landessprache gemacht und dem Management-Zentrum mitgeteilt worden ist, hat den gleichen Status wie die offiziellen Fassungen.

CEN-Mitglieder sind die nationalen Normungsinstitute von Belgien, Bulgarien, Dänemark, Deutschland, Estland, Finnland, Frankreich, Griechenland, Irland, Island, Italien, Kroatien, Lettland, Litauen, Luxemburg, Malta, den Niederlanden, Norwegen, Österreich, Polen, Portugal, Rumänien, Schweden, der Schweiz, der Slowakei, Slowenien, Spanien, der Tschechischen Republik, Ungarn, dem Vereinigten Königreich und Zypern.



EUROPÄISCHES KOMITEE FÜR NORMUNG
EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION

Management-Zentrum: Avenue Marnix 17, B-1000 Brüssel

Inhalt

	Seite
Vorwort	10
Einleitung.....	11
1 Anwendungsbereich	12
2 Konformität.....	12
3 Normative Verweisungen	14
4 Begriffe	15
5 Konventionen	19
5.1 Abkürzungen	19
5.2 Verwendung von Beispielen	20
5.3 XML-Schemata	20
5.4 UML-Notation	20
5.4.1 Klassendiagramme	20
5.4.2 Zustandsmaschinennotation	21
6 Grundlegende Dienstelemente.....	22
6.1 Einleitung.....	22
6.2 Versionsnummern und deren Verhandlung.....	22
6.2.1 Form und Wert der Versionsnummer	22
6.2.2 Vorkommen in Dienst-Metadaten und in Anforderungen.....	22
6.2.3 Verhandeln der Versionsnummer	23
6.2.4 Anforderungskodierung.....	23
6.2.5 Regeln für die KVP-Parameterkodierung	23
6.3 Namensräume	24
6.4 Dienstbindungen.....	24
7 Gemeinsame Elemente	25
7.1 Kodierung von Features	25
7.2 Ressourcen-Identifikatoren	25
7.2.1 Zuweisung von Ressourcen-Identifikatoren	25
7.2.2 Kodierung von Ressourcen-Identifikatoren.....	25
7.2.3 Versionsidentifizierung	25
7.3 Eigenschaftsverweisungen	25
7.3.1 XPath-Teilmenge.....	25
7.3.2 Zugriffsfunktion	26
7.4 Kodierung von Prädikatausdrücken	26
7.5 Ausnahmemeldungen	26
7.6 Gemeinsame Anforderungsparameter	28
7.6.1 Einleitung.....	28
7.6.2 Basisanforderungstyp.....	28
7.6.3 Standardpräsentationsparameter	30
7.6.4 Standardauflösungsparameter.....	33
7.6.5 Standardeingabeparameter	37
7.6.6 Zusätzliche gemeinsame Schlüsselwörter für KVP-kodierte Anforderungen.....	39
7.7 Standardantwortparameter.....	39
7.7.1 Parametersemantik.....	39
7.7.2 XML-Kodierung	40
7.7.3 KVP-Kodierung	40
7.7.4 Erläuterung der Parameter	40

7.8	Verwendung des Attributs „schemaLocation“	43
7.9	Abfrageausdrücke	43
7.9.1	Einleitung	43
7.9.2	Ad-hoc-Abfrageausdruck	44
7.9.3	Gespeicherter Abfrageausdruck	55
8	„GetCapabilities“-Operation	57
8.1	Einleitung	57
8.2	Anforderung	57
8.2.1	Anforderungssemantik	57
8.2.2	XML-Kodierung	58
8.2.3	KVP-Kodierung	58
8.3	Antwort	58
8.3.1	Antwortsemantik	58
8.3.2	XML-Kodierung	59
8.3.3	Fähigkeitendokument	59
8.3.4	FeatureTypeList-Abschnitt	60
8.3.5	Parameter-Wertebereiche und Einschränkungen	62
8.4	Erweiterungspunkte	66
8.5	Ausnahmen	67
9	„DescribeFeatureType“-Operation	68
9.1	Einleitung	68
9.2	Anforderung	68
9.2.1	Anforderungssemantik	68
9.2.2	XML-Kodierung	68
9.2.3	KVP-Kodierung	68
9.2.4	Erläuterung der Parameter	69
9.3	Antwort	70
9.3.1	Einleitung	70
9.3.2	Unterstützung mehrerer Namensräume	70
9.4	Ausnahmen	71
10	„GetPropertyValue“-Operation	71
10.1	Einleitung	71
10.2	Anforderung	71
10.2.1	Anforderungssemantik	71
10.2.2	XML-Kodierung	72
10.2.3	KVP-Kodierung	72
10.2.4	Erläuterung der Parameter	72
10.3	Antwort	74
10.3.1	Antwortsemantik	74
10.3.2	XML-Kodierung	74
10.3.3	„state“-Parameter	75
10.3.4	Standardantwortparameter	76
10.4	Ausnahmen	76
11	„GetFeature“-Operation	76
11.1	Einleitung	76
11.2	Anforderung	77
11.2.1	Anforderungssemantik	77
11.2.2	XML-Kodierung	77
11.2.3	KVP-Kodierung	78
11.2.4	Erläuterung der Parameter	78
11.3	Antwort	79
11.3.1	Antwortsemantik	79
11.3.2	XML-Kodierung	79
11.3.3	Erläuterung der Parameter	80
11.3.4	Zusätzliche Objekte	83
11.3.5	Antwort auf „GetFeatureById“	84

11.4	Ausnahmen	84
12	„LockFeature“-Operation.....	84
12.1	Einleitung.....	84
12.2	Anforderung	85
12.2.1	Anforderungssemantik	85
12.2.2	XML-Kodierung	85
12.2.3	KVP-Kodierung	86
12.2.4	Erläuterung der Parameter	86
12.2.5	Zustandsmaschine für WFS-Sperren.....	88
12.3	Antwort	89
12.3.1	Antwortsemantik.....	89
12.3.2	XML-Kodierung	89
12.4	Ausnahmen	90
13	„GetFeatureWithLock“-Operation.....	90
13.1	Einleitung.....	90
13.2	Anforderung	90
13.2.1	Anforderungssemantik	90
13.2.2	XML-Kodierung	91
13.2.3	KVP-Kodierung	91
13.2.4	Erläuterung der Parameter	91
13.3	Antwort	92
13.3.1	Einleitung.....	92
13.3.2	„lockId“-Paramter	92
13.4	Ausnahmen	92
14	Verwaltung gespeicherte Abfrageausdrücke	93
14.1	Einleitung.....	93
14.2	Definition gespeicherter Abfragen.....	93
14.2.1	XML-Kodierung	93
14.2.2	Erläuterung der Parameter	94
14.3	„ListStoredQueries“-Operation.....	97
14.3.1	Anforderungssemantik	97
14.3.2	XML-Kodierung	97
14.3.3	KVP-Kodierung	97
14.3.4	Antwort	98
14.3.5	Ausnahmen	98
14.4	„DescribeStoredQueries“-Operationen.....	98
14.4.1	Anforderungssemantik	98
14.4.2	XML-Kodierung	99
14.4.3	KVP-Kodierung	99
14.4.4	Antwort	100
14.5	„CreateStoredQuery“-Operation	100
14.5.1	Anforderungssemantik	100
14.5.2	XML-Kodierung	101
14.5.3	KVP-Kodierung	101
14.5.4	Erläuterung der Parameter	101
14.5.5	Antwort	101
14.6	„DropStoredQuery“-Operationen.....	102
14.6.1	Anforderungssemantik	102
14.6.2	XML-Kodierung	102
14.6.3	KVP-Kodierung	103
14.6.4	Antwort	103
14.7	Ausnahmen	103
15	„Transaction“-Operation.....	103
15.1	Einleitung.....	103
15.2	Anforderung	104
15.2.1	Anforderungssemantik	104

15.2.2	XML-Kodierung.....	105
15.2.3	Erläuterung der Parameter	106
15.2.4	„Insert“-Aktion	107
15.2.5	„Update“-Aktion	108
15.2.6	„Replace“-Aktion	110
15.2.7	„Delete“-Aktion	110
15.2.8	„Native“-Aktion	111
15.3	Antwort	112
15.3.1	Antwortsemantik	112
15.3.2	„TransactionResponse“-Element	112
15.3.3	„TransactionSummary“-Element	113
15.3.4	„InsertResults“-Element	113
15.3.5	„UpdateResults“-Element	114
15.3.6	„ReplaceResults“-Element	114
15.4	Ausnahmen	114
Anhang A	(normativ) Konformitätsprüfung	115
A.1	Konformitätsklassen	115
A.1.1	Simple WFS	115
A.1.2	Basic WFS	115
A.1.3	Transactional WFS	115
A.1.4	Locking WFS	116
A.1.5	HTTP GET	116
A.1.6	HTTP POST	116
A.1.7	SOAP	116
A.1.8	Inheritance	117
A.1.9	Remote Resolve	117
A.1.10	Response Paging	117
A.1.11	Standard joins	117
A.1.12	Spatial joins	117
A.1.13	Temporal joins	118
A.1.14	Feature versions	118
A.1.15	Manage stored queries	118
A.2	Grundlegende Prüfungen	118
A.2.1	Versionsverhandlung	118
A.2.2	Auflisten von Versionsnummer 2.0.0 als Versionsnummer der unterstützten Anforderung	119
A.2.3	Ungültige Versionsnummer	119
A.2.4	Versionsverhandlung für die „GetCapabilities“-Anforderung	119
A.2.5	Antwort auf XML- und KVP-kodierte Anforderungen	119
A.2.6	Parameter-Reihenfolge und -Fall	120
A.2.7	Nicht erkannte Parameter	120
A.2.8	Server operiert auf GML-Features	120
A.2.9	Feature-Identifikatoren	121
A.2.10	Invarianter Identifikator	121
A.2.11	Versionierung	121
A.2.12	XPath-Teilmenge	122
A.2.13	Prädikatkodierung	122
A.2.14	Ausnahmemeldung	122
A.2.15	Gemeinsame Anforderungsparameter	123
A.2.16	Standardpräsentationsparameter	124
A.2.17	Standardauflösungsparameter	125
A.2.18	Standardeingabeparameter	128
A.2.19	Standardantwortparameter	129
A.2.20	Antwortpaging	130
A.2.21	„schemaLocation“-Parameter	131
A.2.22	Abfrageausdrücke	131
A.2.23	Deklarieren der Konformität	136

Anhang B (informativ) Beispiele	137
B.1 Beispiel für eine Ausnahmemeldung	137
B.2 Beispiele für „DescribeFeatureTypes“	137
B.2.1 Beispiel 1	137
B.2.2 Beispiel 2	140
B.3 Beispiele für „GetFeature“	144
B.3.1 Einleitung	144
B.3.2 Beispiel 1	144
B.3.3 Beispiel 2	144
B.3.4 Beispiel 3	145
B.3.5 Beispiel 4	145
B.3.6 Beispiel 5	147
B.3.7 Beispiel 6	147
B.3.8 Beispiel 7	147
B.3.9 Beispiel 8	149
B.3.10 Beispiel 9	150
B.3.11 Beispiel 10	153
B.3.12 Beispiel 11	154
B.3.13 Beispiel 12	154
B.3.14 Beispiel 13	155
B.3.15 Beispiel 14	159
B.3.16 Beispiel 15	159
B.3.17 Beispiel 16	160
B.3.18 Beispiel 17	160
B.3.19 Beispiel 18	161
B.3.20 Beispiel 19	162
B.4 Beispiele für „GetPropertyValue“	162
B.4.1 Einleitung	162
B.4.2 Beispiel 1	164
B.4.3 Beispiel 2	165
B.4.4 Beispiel 3	166
B.4.5 Beispiel 4	168
B.4.6 Beispiel 5	168
B.4.7 Beispiel 6	170
B.4.8 Beispiel 7	170
B.4.9 Beispiel 8	171
B.4.10 Beispiel 9	172
B.5 Beispiele für „LockFeature“	173
B.5.1 Beispiel 1	173
B.5.2 Beispiel 2	173
B.5.3 Beispiel 3	174
B.5.4 Beispiel 4	174
B.6 Transaktionsbeispiele	175
B.6.1 Einfügungsbeispiel	175
B.6.2 Aktualisierungsbeispiele	176
B.6.3 Beispiele für Löschungen	178
B.6.4 Beispiel für Mischtransaktionen	180
B.6.5 Beispiel für Transaktionsantworten	183
B.7 Beispiel für „GetCapabilities“	184
B.8 KVP-Beispiele	199
B.8.1 Konventionen	199
B.8.2 Beispiele für „DescribeFeatureType“	200
B.8.3 Beispiele für „GetPropertyValue“	200
B.8.4 Beispiele für „GetFeature“	202
B.8.5 Beispiele für „LockFeature“	208
Anhang C (informativ) Konsolidiertes XML-Schema	210
C.1 Einleitung	210

C.2	wfs.xsd	210
Anhang D (normativ)	Dienstbindungen	221
D.1	Einleitung	221
D.2	HTTP-GET- und -POST-Bindung	221
D.3	HTTP-Statuscodes	221
D.4	SOAP-Bindung	223
D.4.1	Einleitung	223
D.4.2	SOAP Envelope	223
D.4.3	SOAP Header	223
D.4.4	SOAP Body	224
D.4.5	Kodierung von XML-Schema in einem SOAP Body	224
D.4.6	SOAP Fault	225
D.4.7	SOAP-HTTP-Bindung	226
Anhang E (normativ)	Web Service Description Language (WSDL)	227
E.1	Einleitung	227
E.2	WFS-Operationen in WSDL	227
E.3	SOAP-Bindung	227
E.4	Bindungsstil	228
E.5	Dienst	229
E.6	Dienstbeschreibung mit Hilfe von WSDL	229
E.6.1	Einleitung	229
E.6.2	wfs-xml-interfaces.wsdl	230
E.6.3	wfs-kvp-interfaces.wsdl	231
E.6.4	wfs-responses.wsdl	233
E.6.5	wfs-http-bindings.wsdl	234
E.6.6	wfs-kvp-bindings.wsdl	236
E.6.7	wfs-soap-bindings.wsdl	237
E.6.8	Zusatzdateien	240
E.6.9	Beispiele (informativ)	244
Anhang F (informativ)	Abstraktes Modell	247
F.1	Überblick	247
F.2	Abstraktes Ressourcenmodell	247
F.2.1	Einleitung	247
F.2.2	Basiszugriffsfunktion	247
F.3	Abbildung des General Feature Model (GFM) auf das WFS Abstract Model	249
F.4	Identifikatoren	249
F.5	valueOf()-Funktion	249
F.6	WFS-Operationen	249
F.6.1	Einleitung	249
F.6.2	featureTypeNameList()-Funktion	250
F.6.3	featureType()-Funktion	250
F.6.4	Abfragefunktion	250
F.6.5	propertyValue()-Funktion	252
F.6.6	lock()-Funktion	252
F.6.7	transaction()-Funktion	252
F.6.8	Gespeicherte Abfrageoperationen	253
F.7	WFS-Operationen	254
F.8	Konzeptuelles Schema	254
	Literaturhinweise	256

Bilder

Bild 1 — UML-Notation in Klassendiagrammen	20
Bild 2 — Zusammenfassung von UML-Zustandsdiagramm-Notationen	22
Bild 3 — BaseRequest.....	28
Bild 4 — StandardPresentationParameters.....	30
Bild 5 — StandardResolveParameters	34
Bild 6 — StandardInputParameters	37
Bild 7 — StandardResponseParameters.....	40
Bild 8 — Ad-hoc-Abfrageausdruck	45
Bild 9 — Abfrageprojektionsklausel	49
Bild 10 — Abfragesortierklausel	54
Bild 11 — StoredQuery.....	55
Bild 12 — „GetCapabilities“-Anforderung	57
Bild 13 — „GetCapabilities“-Antwort	58
Bild 14 — „DescribeFeatureType“-Anforderung	68
Bild 15 — „GetPropertyValue“-Anforderung	71
Bild 16 — „GetPropertyValue“-Antwort.....	74
Bild 17 — „GetFeature“-Anforderung.....	77
Bild 18 — „GetFeature“-Antwort	79
Bild 19 — „LockFeature“-Anforderung.....	85
Bild 20 — Zustandsdiagramm für eine WFS-Sperre	88
Bild 21 — „LockFeature“-Antwort	89
Bild 22 — „GetFeatureWithLock“-Anforderung.....	90
Bild 23 — „ListStoredQueries“-Anforderung	97
Bild 24 — ListStoredQueriesResponse	98
Bild 25 — „DescribeStoredQueries“-Anforderung	99
Bild 26 — DescribeStoredQueriesResponse	100
Bild 27 — „CreateStoredQuery“-Anforderung	101
Bild 28 — „CreateStoredQuery“-Antwort	102
Bild 29 — „DropStoredQuery“-Anforderung	102
Bild 30 — „Transaction“-Anforderung	105
Bild 31 — „Transaction“-Antwort.....	112
Bild F.1 — Überblick über die Schnittstellen des Web Feature Service.....	255

Tabellen

Tabelle 1 — WFS-Konformitätsklassen	13
Tabelle 2 — Operationsanforderungskodierung	23
Tabelle 3 — WFS-Ausnahmecodes	27
Tabelle 4 — KVP-Kodierung des „Base Request“-Typs	29
Tabelle 5 — KVP-Kodierung von Standardpräsentationsparametern.....	31
Tabelle 6 — KVP-Kodierung von Standardauflösungsparametern.....	35
Tabelle 7 — Zusätzliche gemeinsame Schlüsselwörter für KVP-kodierte WFS-Anforderungen	39
Tabelle 8 — Schlüsselwörter zur KVP-Kodierung von Ad-hoc-Abfragen.....	46
Tabelle 9 — KVP-Kodierung einer Projektionsklausel	50
Tabelle 10 — Schlüsselwörter für die KVP-Kodierung gespeicherter Abfrageausdrücke	56
Tabelle 11 — Elemente zur Beschreibung von Feature-Typen	61
Tabelle 12 — Parameter-Wertebereiche für WFS-Operationen	62
Tabelle 13 — Diensteinschränkungen	64
Tabelle 14 — Operationseinschränkungen	65
Tabelle 15 — KVP-Kodierung der „DescribeFeatureType“-Anforderung.....	69
Tabelle 16 — Schlüsselwörter für die KVP-Kodierung von „GetPropertyValue“	72
Tabelle 17 — Schlüsselwörter zur KVP-Kodierung von „GetFeature“	78
Tabelle 18 — Schlüsselwörter zur KVP-Kodierung von „LockFeature“	86
Tabelle 19 — Zusätzliche Schlüsselwörter für die KVP-Kodierung von „GetFeatureWithLock“	91
Tabelle 20 — Schlüsselwörter für die KVP-Kodierung von „ListStoredQueries“	97
Tabelle 21 — Schlüsselwörter für die KVP-Kodierung von „DescribeStoredQueries“	99
Tabelle 22 — Schlüsselwörter für die KVP-Kodierung der „DropStoredQuery“-Operation	103
Tabelle D.1 — Anforderungskodierung und Transportverfahren	221
Tabelle D.2 — Korrelation von OWS- und WFS-Ausnahmecodes mit HTTP-Statuscodes.....	222
Tabelle F.1 — Abbildung der Operationen des abstrakten Modells auf WFS-Operationen	254

Vorwort

Dieses Dokument (EN ISO 19142:2010) wurde vom Technischen Komitee ISO/TC 211 „Geographic information/Geomatics“ in Zusammenarbeit mit dem Technischen Komitee CEN/TC 287 „Geoinformation“ erarbeitet, dessen Sekretariat vom BSI gehalten wird.

Diese Europäische Norm muss den Status einer nationalen Norm erhalten, entweder durch Veröffentlichung eines identischen Textes oder durch Anerkennung bis Juni 2011, und etwaige entgegenstehende nationale Normen müssen bis Juni 2011 zurückgezogen werden.

Es wird auf die Möglichkeit hingewiesen, dass einige Texte dieses Dokuments Patentrechte berühren können. CEN [und/oder CENELEC] sind nicht dafür verantwortlich, einige oder alle diesbezüglichen Patentrechte zu identifizieren.

Entsprechend der CEN/CENELEC-Geschäftsordnung sind die nationalen Normungsinstitute der folgenden Länder gehalten, diese Europäische Norm zu übernehmen: Belgien, Bulgarien, Dänemark, Deutschland, Estland, Finnland, Frankreich, Griechenland, Irland, Island, Italien, Kroatien, Lettland, Litauen, Luxemburg, Malta, Niederlande, Norwegen, Österreich, Polen, Portugal, Rumänien, Schweden, Schweiz, Slowakei, Slowenien, Spanien, Tschechische Republik, Ungarn, Vereinigtes Königreich und Zypern.

Anerkennungsnotiz

Der Text von ISO 19142:2010 wurde vom CEN als EN ISO 19142:2010 ohne irgendeine Abänderung genehmigt.

Einleitung

Mit dem Web Feature Service (WFS) lassen sich Geoinformationen auf neue Art erzeugen, modifizieren und über das Internet austauschen. Statt Geoinformationen etwa auf Dateiebene mittels File Transfer Protocol (FTP) weiterzugeben, bietet der WFS einen direkten, feinkörnigen Zugriff auf Geoinformationen auf Feature- und Feature-Eigenschaftsebene. Mit Hilfe von Web Feature Services können Kunden speziell die gesuchten Daten abrufen oder modifizieren und müssen nicht mehr eine Datei abrufen, die neben den gewünschten Daten möglicherweise auch zahlreiche andere Elemente enthält. Die Daten können dann für viele verschiedene Zwecke verwendet werden, darunter auch für andere als vom Erzeuger vorgesehene Zwecke.

In der in ISO 19119 definierten Dienste-Taxonomie ist der WFS primär ein Feature-Zugriffsdienst, umfasst aber auch Elemente eines Feature-Typ-Dienstes, eines Dienstes zur Koordinatenumwandlung/-transformation und eines Dienstes zur Umwandlung geographischer Formate.

1 Anwendungsbereich

Diese Internationale Norm legt das Verhalten eines Dienstes fest, der Transaktionen auf geographischen Features und den Zugriff auf diese unabhängig von dem zugrunde liegenden Datenspeicher bietet. Die Norm legt Ermittlungsoperationen, Abfrageoperationen, Sperroperationen, Transaktionsoperationen sowie Operationen zum Speichern parametrisierter Abfrageausdrücke fest.

Durch Ermittlungsoperationen kann der Dienst abgefragt werden, um seine Fähigkeiten festzustellen und das Anwendungsschema abzurufen, das die vom Dienst angebotenen Feature-Typen definiert.

Mit Abfrageoperationen können Features oder Werte von Feature-Eigenschaften aus dem zugrunde liegenden Datenspeicher auf der Basis von vom Client definierten Einschränkungen der Feature-Eigenschaften abgerufen werden.

Sperroperationen ermöglichen einen exklusiven Zugriff auf Features, um diese zu modifizieren oder zu löschen.

Mit Transaktionsoperationen lassen sich Features erzeugen, ändern, ersetzen und aus dem zugrunde liegenden Datenspeicher löschen.

Durch gespeicherte Abfrageoperationen sind Clients in der Lage, parametrisierte Abfrageausdrücke, die vom Server gespeichert werden und wiederholt mit Hilfe unterschiedlicher Parameterwerte aufgerufen werden können, zu erzeugen, zu löschen, aufzulisten und zu beschreiben.

ANMERKUNG Fragen der Zugriffsüberwachung sind in dieser Internationalen Norm nicht behandelt.

Diese Internationale Norm definiert die folgenden 11 Operationen:

- GetCapabilities (Ermittlungsoperation);
- DescribeFeatureType (Ermittlungsoperation);
- GetPropertyValue (Abfrageoperation);
- GetFeature (Abfrageoperation);
- LockFeature (Sperroperation);
- GetFeatureWithLock (Abfrage- und Sperroperation);
- Transaction (Transaktionsoperation);
- CreateStoredQuery (gespeicherte Abfrageoperation);
- DropStoredQuery (gespeicherte Abfrageoperation);
- ListStoredQueries (gespeicherte Abfrageoperation);
- DescribeStoredQuery (gespeicherte Abfrageoperation).

2 Konformität

In Tabelle 1 sind die in dieser Internationalen Norm definierten Konformitätsklassen aufgeführt sowie die in Anhang A festgelegten Prüfungen, die für eine Übereinstimmung mit jeder Klasse erfüllt sein müssen.

Darüber hinaus ist in Tabelle 1 folgendes aufgeführt:

- a) welche Konformitätsprüfungen zur Filtercodierung (siehe ISO 19143:2010, Abschnitt 2) gegebenenfalls für jede WFS-Konformitätsklasse bestanden werden müssen;
- b) welche GML-Konformitätsprüfungen (GML siehe ISO 19136:2007) gegebenenfalls für jede WFS-Konformitätsklasse bestanden werden müssen.

Tabelle 1 — WFS-Konformitätsklassen

Name der Konformitäts-klasse	Operation oder Verhalten	WFS-Konformitätsprüfung	FES-Konformitätsprüfung(en)	GML-Konformitätsprüfung(en)
Simple WFS	Der Server muss die folgenden Operationen implementieren: GetCapabilities, DescribeFeatureType, ListStoredQueries, DescribeStoredQueries, GetFeature mit mindestens der Aktion „StoredQuery“. Es muss eine gespeicherte Abfrage verfügbar sein, die ein Feature mittels seiner ID holt, doch der Server darf auch zusätzliche gespeicherte Abfragen anbieten. Darüber hinaus muss der Server mindestens einer der folgenden Konformitätsklassen entsprechen: HTTP GET, HTTP POST oder SOAP.	A.1.1	ISO 19143:2010, A.1	ISO 19136:2007, A.1.1, A.1.4, A.1.5, A.1.7, B.3, B.5, B.2.3
Basic WFS	Der Server muss die Konformitätsklasse „Simple WFS“ und zusätzlich die Operation „GetFeature“ mit der Aktion „Query“ sowie die Operation „GetPropertyValue“ implementieren.	A.1.2	ISO 19143:2010, A.2, A.7, A.8, A.10, A.11, A.12, A.14	ISO 19136:2007, B.4
Transactional WFS	Der Server muss die Konformitätsklasse „Basic WFS“ und außerdem die Operation „Transaction“ implementieren.	A.1.3		
Locking WFS	Der Server muss die Konformitätsklasse „Transactional WFS“ und mindestens eine der Operationen „GetFeatureWithLock“ oder „LockFeature“ implementieren.	A.1.4		
HTTP GET	Der Server muss die Schlüssel/Wert-Paar-Kodierung für die vom Server angebotenen Operationen implementieren.	A.1.5		
HTTP POST	Der Server muss die XML-Kodierung für die vom Server implementierten Operationen implementieren.	A.1.6		
SOAP	Der Server muss XML-kodierte Anforderungen und Ergebnisse innerhalb von SOAP Envelopes implementieren.	A.1.7		
Inheritance	Der Server muss die Schema-Element()-Funktion in Xpath-Ausdrücken implementieren.	A.1.8	ISO 19143:2010, A.1.15	

Tabelle 1 (fortgesetzt)

Name der Konformitäts-klasse	Operation oder Verhalten	WFS-Konformitätsprüfung	FES-Konformitätsprüfung(en)	GML-Konformitätsprüfung(en)
Remote resolve	Der Server muss die Fähigkeit zur Auflösung von abgesetzten Ressourcenverweisen implementieren.	A.1.9		ISO 19136:2007, B.2.1
Response paging	Der Server muss die Fähigkeit implementieren, die Antwort-Features oder -Werte zu durchblättern.	A.1.10		ISO 19136:2007, B.3
Standard joins	Der Server muss Verknüpfungsprädikate mit Hilfe aller „Filter“-Operatoren, ausgenommen die räumlichen und zeitlichen Operatoren, implementieren.	A.1.11	ISO 19143:2010, A.8, A.10	
Spatial joins	Der Server muss Verknüpfungsprädikate mit Hilfe von räumlichen Operatoren implementieren.	A.1.12	ISO 19143:2010, A.11, A.12	
Temporal joins	Der Server muss Verknüpfungsprädikate mit Hilfe von zeitlichen Operatoren implementieren.	A.1.13	ISO 19143:2010, A.9, A.10	
Feature versions	Der Server muss die Fähigkeit zur Navigation von Feature-Versionen implementieren.	A.1.14	ISO 19143:2010, A.11	
Manage stored queries	Der Server muss die Operationen „CreateStoredQuery“ und „DropStoredQuery“ implementieren.	A.1.15	ISO 19143:2010, A.1	

3 Normative Verweisungen

Die folgenden zitierten Dokumente sind für die Anwendung dieses Dokuments erforderlich. Bei datierten Verweisungen gilt nur die in Bezug genommene Ausgabe. Bei undatierten Verweisungen gilt die letzte Ausgabe des in Bezug genommenen Dokuments (einschließlich aller Änderungen).

ISO/TS 19103:2005, *Geographic information — Conceptual schema language*

ISO 19136:2007, *Geographic information — Geography Markup Language (GML)*

ISO 19143:2010, *Geographic information — Filter Encoding*

IETF RFC 2616, *Hypertext Transfer Protocol — HTTP/1.1* (June 1999)

IETF RFC 4646, *Tags for Identifying Languages* (September 2006)

OGC 06-121r3, *OGC Web Services Common Specification*, OGC® Implementation Specification (9 February 2009)

OGC 07-092r3, *Definition identifier URNs in OGC namespace*, OGC® Best Practices (15 January 2009)

W3C SOAP, *Simple Object Access Protocol (SOAP) 1.2*, W3C Note (27 April 2007)

W3C WSDL, *Web Services Description Language (WSDL) 1.1*, W3C Note (15 March 2001)

W3C XML Namespaces, *Namespaces in XML*, W3C Recommendation (14 January 1999)

W3C XML Path Language, *XML Path Language (XPath) 2.0*, W3C Recommendation (23 January 2007)

W3C XML Schema Part 1, *XML Schema Part 1: Structures*, W3C Recommendation (2 May 2001)

W3C XML Schema Part 2, *XML Schema Part 2: Datatypes*, W3C Recommendation (2 May 2001)

4 Begriffe

Für die Anwendung dieses Dokuments gelten die folgenden Begriffe.

4.1

Attribut <XML> (en: attribute)

in einem **Element** enthaltenes Name/Wert-Paar (4.6)

[ISO 19136:2007, Definition 4.1.3]

ANMERKUNG Sofern nicht anders festgelegt, sind die Attribute in dieser Internationalen Norm XML-Attribute.

4.2

Client (en: client)

Softwarekomponente, die eine **Operation** (4.17) von einem **Server** (4.28) aufrufen kann

[ISO 19128:2005, Definition 4.1]

4.3

Koordinate (en: coordinate)

eine Zahl aus einer Folge von n Zahlen, die die Position eines Punktes im n -dimensionalen Raum bezeichnet

[ISO 19111:2007, Definition 4.5]

4.4

Koordinatenreferenzsystem (en: coordinate reference system)

Koordinatensystem (4.5), das durch ein Datum auf ein Objekt bezogen ist

[ISO 19111:2007, Definition 4.8]

4.5

Koordinatensystem (en: coordinate system)

Menge mathematischer Regeln zur Festlegung, wie **Koordinaten** (4.3) Punkten zuzuordnen sind

[ISO 19111:2007, Definition 4.10]

4.6

Element <XML> (en: element)

grundlegendes Informationselement eines XML-Dokuments, das abgeleitete Elemente, **Attribute** (4.1) und Zeichendaten enthält

[ISO 19136:2007, Definition 4.1.23]

4.7

Feature (en: feature)

Abstraktion eines Phänomens der realen Welt

[ISO 19101:2002, Definition 4.11]

ANMERKUNG Ein Feature kann als ein Typ oder eine Instanz vorkommen. Feature-Typ oder Feature-Instanz ist zu verwenden, wenn nur eines gemeint ist.

4.8

Feature-Identifikator (en: feature identifier)

Identifikator, der eine **Feature**-Instanz (4.7) eindeutig bezeichnet

4.9

Filterausdruck (en: filter expression)

mit Hilfe von XML kodierter Prädikatausdruck

[ISO 19143:2010, Definition 4.11]

4.10

Schnittstelle (en: interface)

benannte Menge von **Operationen** (4.17), die das Verhalten einer Entität charakterisieren

[ISO 19119:2005, Definition 4.2]

4.11

Verknüpfungsprädikat (en: join predicate)

Filterausdruck (4.9), der eine oder mehrere Klauseln umfasst, welche Eigenschaften von zwei unterschiedlichen Entitätstypen einschränken

[ISO 19143:2010, Definition 4.16]

ANMERKUNG In dieser Internationalen Norm handelt es sich bei den Entitätstypen um Feature-Typen.

4.12

Verknüpfungstupel (en: join tuple)

Menge von zwei oder mehr Objektinstanzen, die einem Filter entsprechen, der **Verknüpfungsprädikate** enthält

ANMERKUNG In dieser Internationalen Norm handelt es sich bei den Objektinstanzen um **Feature**-Instanzen.

4.13

lokale Ressource (en: local resource)

Ressource, die direkt von einem System gesteuert wird

ANMERKUNG In dieser Internationalen Norm handelt es sich bei dem System um einen Web Feature Service, und die Ressource wird in einem Datenspeicher gehalten, der von diesem Web Feature Service direkt gesteuert wird.

4.14

Lokalisatorattribut (en: locator attribute)

Attribut (4.1), dessen Wert eine Verweisung auf eine **lokale Ressource** (4.13) oder **abgesetzte Ressource** (4.20) ist

ANMERKUNG In XML wird dieses Attribut im Allgemeinen href genannt und enthält einen URI-Verweis auf die abgesetzte Ressource (siehe W3C XLink).

4.15

MIME-Typ (en: Multipurpose Internet Mail Extensions type)

Medientyp und Untertyp von Daten im Hauptteil einer Nachricht, der die ursprüngliche Darstellung (kanonische Form) dieser Daten darstellt

[IETF RFC 2045:1996]

4.16

Namensraum <XML> (en: namespace)

Sammlung von Namen, identifiziert durch einen URI-Verweis, die in XML-Dokumenten als **Elementnamen** (4.6) und **Attributnamen** (4.1) verwendet werden

[W3C XML Namespaces:1999]

4.17

Operation (en: operation)

Festlegung einer Transformation oder Abfrage, zu deren Ausführung ein Objekt aufgerufen werden kann

[ISO 19119:2005, Definition 4.3]

4.18

Eigenschaft (en: property)

Facette oder Merkmal eines Objekts, referenziert durch einen Namen

[ISO 19143:2010, Definition 4.21]

4.19

Ressource (en: resource)

Komponente oder Mittel, die bzw. das eine Anforderung erfüllt

[ISO 19115:2003, Definition 4.10]

ANMERKUNG In dieser Internationalen Norm handelt es sich bei der Ressource um ein **Feature** (4.7) oder eine identifizierbare Komponente eines Features (z. B. eine Eigenschaft eines Features).

4.20

abgesetzte Ressource (en: remote resource)

Ressource, die nicht direkt von einem System gesteuert wird

ANMERKUNG In dieser Internationalen Norm handelt es sich bei dem System um einen Web Feature Service. Die Ressource wird in keinem von diesem Web Feature Service direkt gesteuerten Datenspeicher gehalten und kann somit nicht direkt von diesem Dienst abgerufen werden.

4.21

Anforderung (en: request)

Aufruf einer **Operation** (4.17) durch einen **Client** (4.2)

[ISO 19128:2005, Definition 4.10]

4.22

Verschieben <Verweisung> (en: relocate)

Aktualisieren einer Verweisung auf eine Ressource, die an einen neuen Ort bewegt oder kopiert wurde

BEISPIEL Ein **Server** (4.28) generiert eine **Antwort** (4.24) auf eine GetFeature-**Anforderung** (4.21); er muss ein referenziertes **Feature** (4.7) in das Antwortdokument kopieren, und der Server muss die im referenzierenden Feature enthaltene ursprüngliche Verknüpfung in die im Antwortdokument abgelegte Kopie „verschieben“.

4.23

auflösen (en: resolve)

Abrufen einer referenzierten Ressource und Einfügen dieser Ressource in ein servergeneriertes Antwortdokument

ANMERKUNG Das Einfügen darf entweder durch Inline-Ersetzen (eingefügtes Ersetzen) der Verweisung durch eine Kopie der Ressource erfolgen oder durch Verschieben der Verweisung so, dass sie auf eine Kopie der Ressource zeigt, die in das Antwortdokument eingebracht wurde.

4.24

Antwort (en: response)

Ergebnis einer **Operation** (4.17), das von einem **Server** (4.28) an einen **Client** (4.2) zurückgegeben wird

[ISO 19128:2005, Definition 4.11]

4.25

Antwortmodell (en: response model)

Schema (4.26), das die Eigenschaften jedes **Feature**-Typs (4.7) definiert, der in der **Antwort** (4.24) auf eine Abfrageoperation (4.17) erscheinen kann

ANMERKUNG Hierbei handelt es sich um das Schema von Feature-Typen, die ein **Client** mit der Operation DescribeFeatureType (siehe Abschnitt 9) beziehen kann.

4.26

Schema (en: schema)

formale Beschreibung eines Modells

[ISO 19101:2002, Definition 4.25]

ANMERKUNG Ein Schema ist im Allgemeinen eine abstrakte Darstellung der Merkmale eines Objekts und seiner Beziehungen zu anderen Objekten. Ein XML-Schema stellt die Beziehung zwischen den **Attributen** (4.1) und **Elementen** (4.6) eines XML-Objekts dar (zum Beispiel eines Dokuments oder Teils eines Dokuments).

4.27

Schema (en: schema)

<XML-Schema> Sammlung von **Schemakomponenten** (4.26) innerhalb desselben **Zielnamensraums** (4.16)

[ISO 19136:2007, Definition 4.1.54]

BEISPIEL Schemakomponenten von XML-Schema nach W3C sind Typen, **Elemente** (4.6), **Attribute** (4.1), Gruppen usw.

4.28

Server (en: server)

bestimmte Instanz eines **Dienstes** (4.29)

[ISO 19128:2005, Definition 4.12]

4.29

Dienst (en: service)

Service

charakteristischer Teil der Funktionalität, der von einer Entität durch **Schnittstellen** (4.10) bereitgestellt wird

[ISO 19119:2005, Definition 4.1]

4.30

Dienst-Metadaten (en: service metadata)

Metadaten, die die auf einem **Server** (4.17) verfügbaren **Operationen** (4.28) und geographischen Informationen beschreiben

[ISO 19128:2005, Definition 4.14]

4.31

Durchquerung <XML> (en: traversal)

die Nutzung oder das Folgen einer XLink-Verknüpfung zu beliebigen Zwecken

[W3C XLink:2001]

4.32

Tupel (en: tuple)

geordnete Liste von Werten

[ISO 19136:2007, Definition 4.1.63]

ANMERKUNG In dieser Internationalen Norm handelt es sich bei der geordneten Liste im Allgemeinen um eine endliche Folge von **Features** (4.7) mit jeweils einem spezifischen Feature-Typ.

4.33

Uniform Resource Identifier

URI

eindeutiger Identifikator für eine Ressource, der in Übereinstimmung mit IETF RFC 2396 aufgebaut ist

[ISO 19136:2007, Definition 4.1.65]

ANMERKUNG Die allgemeine Syntax ist <scheme>::<scheme-specified-part>. Die hierarchische Syntax mit einem **Namensraum** ist <scheme>://<authority><path>?<query>.

5 Konventionen

5.1 Abkürzungen

CGI	Common Gateway Interface
CRS	Coordinate Reference System
DCP	Distributed Computing Platform
EPSG	European Petroleum Survey Group
FES	Filter Encoding Specification
GML	Geography Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Secure Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
KVP	Keyword-Value Pairs (Schlüsselwort/Wert-Paare)
MIME	Multipurpose Internet Mail Extensions
OGC	Open Geospatial Consortium
OWS	OGC Web Service
SQL	Structured Query Language
SOAP	Simple Object Access Protocol
UML	Unified Modelling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
VSP	Vendor Specific Parameter
WFS	Web Feature Service
WSDL	Web Services Description Language
XML	Extensible Markup Language

5.2 Verwendung von Beispielen

In dieser Internationalen Norm werden zahlreiche XML-Beispiele verwendet. Diese sollen die verschiedenen Aspekte eines in dieser Internationalen Norm festgelegten Web Feature Service veranschaulichen. Ein Großteil der Beispiele findet sich in Anhang B, einige Beispiele sind auch im Hauptteil der Norm enthalten. Alle Beispiele referenzieren fiktive Server und Daten. Diese Internationale Norm erhebt somit keinen Anspruch darauf, dass aus dieser Norm kopierte XML- oder Schlüsselwort/Wert-Paar-kodierte Beispiele eine korrekte Ausführung oder die Validierung mit einem bestimmten XML-Validierungswerkzeug bewirken.

5.3 XML-Schemata

In der gesamten Internationalen Norm werden Fragmente von XML-Schema (siehe W3C XML Schema, Part 1 und W3C XML Schema Part 2) verwendet, um die XML-Kodierung von WFS-Operationen zu definieren. Diese Fragmente sind in Anhang C zu einer validierten Schemadatei zusammengefasst.

5.4 UML-Notation

5.4.1 Klassendiagramme

Bild 1 beschreibt die in dieser Internationalen Norm für UML-Klassendiagramme verwendeten Notationen der Unified Modelling Language (UML).

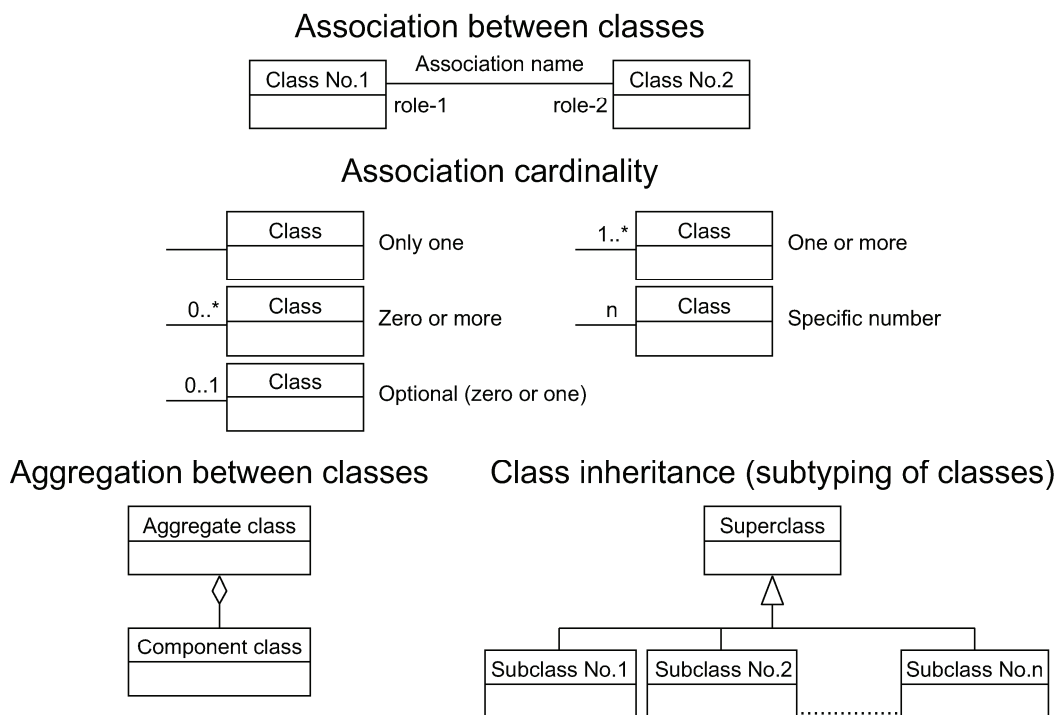


Bild 1 — UML-Notation in Klassendiagrammen

In diesen Klassendiagrammen werden die folgenden Stereotypen von UML-Klassen verwendet:

- a) <<DataType>> Ein Deskriptor einer Menge von Werten ohne Identität (unabhängige Existenz und Möglichkeit von Nebeneffekten). Ein „DataType“ ist eine Klasse ohne Operationen, deren Hauptzweck im Halten der Informationen besteht.

- b) <<Enumeration>> Ein Datentyp, dessen Instanzen eine Liste alternativer Literalwerte bilden. Enumeration bedeutet eine kurze Liste hinreichend verstandener potentieller Werte innerhalb einer Klasse.
- c) <<CodeList>> Eine flexible Enumeration zum Ausdrücken einer langen Liste potentieller alternativer Werte. Wenn die Listenalternativen vollständig bekannt sind, ist eine Enumeration zu verwenden; sind die einzig wahrscheinlichen Alternativen bekannt, ist eine Codeliste zu verwenden.
- d) <<Interface>> Eine Definition einer Menge von Operationen, die von Objekten mit dieser Schnittstelle unterstützt werden. Eine „Interface“-Klasse kann keine Attribute enthalten.
- e) <<Type>> Eine stereotypisierte Klasse, die zur Festlegung eines Wertebereichs von Instanzen (Objekten), zusammen mit den auf diese Objekte anwendbaren Operationen, verwendet wird. Eine „Type“-Klasse kann Attribute und Assoziationen aufweisen.
- f) <<Union>> Eine Liste alternativer Attribute, bei denen zu einem Zeitpunkt jeweils nur eines der Attribute vorhanden sein kann.

Siehe auch ISO/TS 19103:2005, 6.8.2 und D.8.3.

In dieser Internationalen Norm werden die folgenden Standarddatentypen verwendet:

- 1) `CharacterString` — eine Folge von Zeichen;
- 2) `LocalisedCharacterString` — ein mit einer Sprachumgebung (en: locale) assoziierter `CharacterString`;
- 3) `Boolean` — ein Wert, der WAHR (en: TRUE) oder FALSCH (en: FALSE) festlegt;
- 4) `URI` — ein Identifikator einer Ressource, die weitere Informationen bereitstellt;
- 5) `Integer` — eine ganze Zahl.

5.4.2 Zustandsmaschinennotation

Es wurde die im UML Reference Manual beschriebene Herangehensweise an die dynamische Modellierung angewendet. Das wesentliche Verfahren ist dabei die Zustandsmaschinensicht. Eine Zusammenfassung der UML-Notation für Zustandsdiagramme ist in Bild 2 dargestellt.

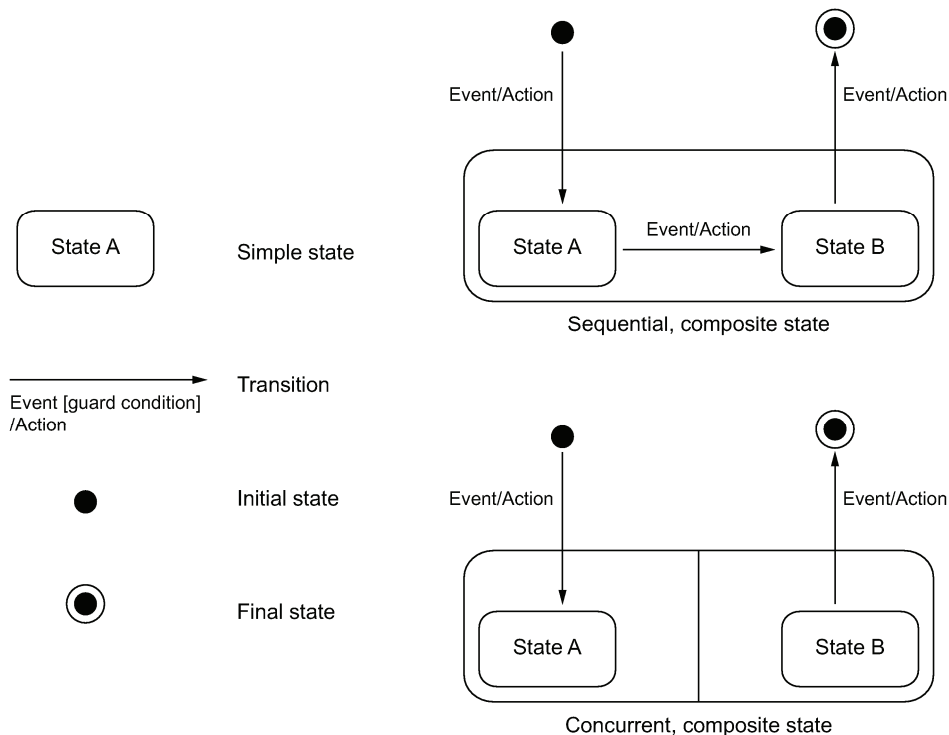


Bild 2 — Zusammenfassung von UML-Zustandsdiagramm-Notationen

6 Grundlegende Dienstelemente

6.1 Einleitung

Dieser Abschnitt beschreibt die Aspekte des Verhaltens eines Web Feature Service, die von bestimmten Operationen unabhängig sind oder mehreren Operationen oder Schnittstellen gemeinsam sind.

6.2 Versionsnummern und deren Verhandlung

6.2.1 Form und Wert der Versionsnummer

Versionsnummern müssen nach OGC 06-121r3:2009, 7.3.1, kodiert sein. Implementierungen dieser Internationalen Norm müssen den Wert „2.0.0“ als Protokollversionsnummer verwenden.

6.2.2 Vorkommen in Dienst-Metadaten und in Anforderungen

Ein mit dieser Internationalen Norm übereinstimmender Web-Feature-Server muss in seinen Dienst-Metadaten die Version „2.0.0“ (siehe 8.3) auflisten. Ein Server kann mehrere Versionen unterstützen, deren Werte von Clients entsprechend den Versionsverhandlungsregeln (siehe OGC 06-121r3:2009, 7.3.2) ermittelt werden können.

Die Versionsnummer, die in Anforderungen an einen Server verwendet wird, muss einer Versionsnummer entsprechen, die der Server laut seiner Deklaration unterstützt (außer während der Verhandlung, siehe Beschreibung in 6.2.3). Wenn der Server eine Anforderung mit einer von ihm nicht unterstützten Versionsnummer empfängt, muss er eine „InvalidParameterValue“-Ausnahme generieren (siehe 7.5).

Für Anforderungen, die mit dieser Internationalen Norm übereinstimmen, muss die Versionsnummer 2.0.0 festgelegt werden.

6.2.3 Verhandeln der Versionsnummer

Alle WFS-Implementierungen müssen eine Versionsverhandlung nach den in OGC 06-121r3:2009, 7.3.2, beschriebenen Regeln unterstützen.

6.2.4 Anforderungskodierung

Diese Internationale Norm definiert zwei Verfahren zur Kodierung von WFS-Anforderungen. Das erste Verfahren verwendet XML als Kodierungssprache. Das zweite Verfahren verwendet Schlüsselwort/Wert-Paare (en: keyword-value pairs, KVP) zur Kodierung der verschiedenen Parameter einer Anforderung.

BEISPIEL Ein Beispiel für ein Schlüsselwort/Wert-Paar ist „REQUEST=GetCapabilities“, wobei „REQUEST“ das Schlüsselwort und „GetCapabilities“ der Wert ist.

Die KVP-Kodierung ist eine Untergruppe der XML-Kodierung, da sich die KVP-Kodierung zum Kodieren bestimmter WFS-Operationen, wie zum Beispiel der Operation „Transaction“, nicht eignet. In beiden Fällen muss die Antwort auf eine Anfrage oder eine Ausnahmemeldung identisch sein.

Tabelle 2 ordnet WFS-Operationen ihrer entsprechenden Kodierungssemantik nach dieser Internationalen Norm zu.

Tabelle 2 —Operationsanforderungskodierung

Operation	Anforderungskodierung
GetCapabilities	XML und KVP
DescribeFeatureType	XML und KVP
GetPropertyValue	XML und KVP
GetFeature	XML und KVP
GetFeatureWithLock	XML und KVP
LockFeature	XML und KVP
Transaction	XML
CreateStoredQuery	XML
DropStoredQuery	XML und KVP
ListStoredQueries	XML und KVP
DescribeStoredQueries	XML und KVP

6.2.5 Regeln für die KVP-Parameterkodierung

6.2.5.1 Einleitung

Aspekte der KVP-Kodierung von WFS-Anforderungen sind in 6.2.5.2 und 6.2.5.3 beschrieben.

6.2.5.2 Parameterreihenfolge und Groß-/Kleinschreibung

Parameternamen dürfen nicht Groß-/Kleinschrift-empfindlich sein, Parameterwerte hingegen müssen dies sein. In dieser Internationalen Norm sind die in der KVP-Kodierung verwendeten Parameternamen aus Gründen der typographischen Verdeutlichung üblicherweise in Großbuchstaben dargestellt, was jedoch nicht als Anforderung zu verstehen ist.

Parameter in einer Anforderung dürfen in beliebiger Reihenfolge festgelegt werden.

Ein Web Feature Service muss darauf vorbereitet sein, auf Parameter zu treffen, die nicht Teil dieser Internationalen Norm sind. Für das Produzieren von Ergebnissen nach dieser Internationalen Norm muss ein Web Feature Service diese Parameter ignorieren.

6.2.5.3 Parameterlisten

Aus Listen bestehende Parameter sind nach ISO 19143:2010, 5.5, zu kodieren.

BEISPIEL Bei KVP-Kodierung mehrerer Verknüpfungsabfragen (siehe 7.9.2.5.3) werden Klammern verwendet, um die verschiedenen Parameter so voneinander zu trennen, dass sich ihre Werte korrekt ausrichten. Verdeutlicht wird dies durch das folgende Beispiel:

```
TYPENAMES=(ns1:F1,ns2:F2) (ns1:F1,ns1:F1) &ALIASES=(A,B) (C,D) &FILTER=(<Filter> ... for A,B ...  
</Filter>) (<Filter>...for C,D...</Filter>)
```

Dieses KVP-kodierte Fragment kodiert zwei Verknüpfungsabfragen. Die erste Abfrage verknüpft die Feature-Typen „ns1:F1“ und „ns2:F2“, die die Aliasnamen „A“ und „B“ erhalten haben. Die zweite Abfrage verknüpft die Feature-Typen „ns1:F1“ und „ns1:F1“ (eine Selbstverknüpfung), die die Aliasnamen „C“ und „D“ erhalten haben. Der FILTER-Parameter kodiert zwei durch Klammern eingegrenzte Filter, eines für jede Abfrage. Alle Parameterwerte richten sich 1:1 aus. Dieses Fragment ist äquivalent zu den folgenden beiden individuellen KVP-Kodierungsfragmenten:

```
TYPENAMES=ns1:F1,ns2:F2&ALIASES=A,B&FILTER=<Filter>...for A,B...</Filter>  
TYPENAMES=ns1:F1,ns1:F1&ALIASES=C,D&FILTER=<Filter>...for C,D...</Filter>
```

6.3 Namensräume

Namensräume (siehe W3C XML Namespaces) werden verwendet, um XML-Vokabulare von anderen Vokabularen zu unterscheiden. Für den WFS gibt es die folgenden vier normativen Namensraumdefinitionen:

- (<http://www.opengis.net/wfs/2.0>) – für das WFS-Schnittstellenvokabular;
- (<http://www.opengis.net/gml/3.2>) – für das GML-Vokabular (siehe ISO 19136:2007);
- (<http://www.opengis.net/fes/2.0>) – für das OGC-Filter-Vokabular (siehe ISO 19143);
- (<http://www.opengis.net/ows/1.1>) – für das OWS-Common-Vokabular (siehe OGC 06-121r3).

Darüber hinaus können WFS-Implementierungen eines oder mehrere GML-Anwendungsschemata nutzen, die wiederum einen oder mehrere Anwendungsnamensräume verwenden (z. B. <http://www.someserver.com/myns>). Während viele Beispiele in dieser Internationalen Norm einen einzigen Namensraum verwenden, dürfen auch mehrere Namensräume verwendet werden, wie in 11.3.3 gezeigt.

Bei XML-kodierten Anforderungen müssen die in der Anforderung verwendeten Namensräume mit der Notation „xmlns:prefix=namespace_uri“ im Wurzelement der Anforderung kodiert sein (siehe W3C XML Namespaces).

Bei KVP-kodierten Anforderungen muss der NAMESPACES-Parameter (siehe 7.6.6) verwendet werden, um die in der Anforderung verwendeten Namensräume zu deklarieren.

6.4 Dienstbindungen

Der Hauptteil dieser Internationalen Norm definiert die Kodierung von WFS-Anforderungs- und WFS-Antwortnachrichten unabhängig von einem bestimmten Kommunikationsprotokoll. Implementierungen dieser Norm müssen jedoch entweder „HTTP GET“, „HTTP POST“ oder „SOAP over HTTP POST“ (siehe Abschnitt 2) unterstützen. Anhang D enthält eine detaillierte Erläuterung dieser Dienstbindungen.

7 Gemeinsame Elemente

7.1 Kodierung von Features

Server, die dieser Internationalen Norm entsprechen, müssen auf GML-kodierten Features operieren. Bei der unterstützten GML-Version muss es sich dabei um die Version nach ISO 19136:2007 handeln. Die Operationen dieser Internationalen Norm sind jedoch so definiert, dass sie auch mit älteren und zukünftigen GML-Versionen verwendet werden können. Die Server dürfen also neben ISO 19136:2007 auch eine Unterstützung zusätzlicher GML-Versionen implementieren. Server müssen mit Hilfe der „inputFormat“- und „outputFormat“-Parameter-Wertebereiche (siehe Tabelle 12) alle unterstützten GML-Versionen in ihrem Fähigkeitendokument anzeigen.

Außerdem dürfen Server zusätzliche Nicht-GML-Feature-Kodierungen unterstützen, die ebenfalls im Fähigkeitendokument der Server aufgeführt sein müssen (siehe Tabelle 12). Diese Internationale Norm beschreibt jedoch nicht, wie ein Server auf derartigen Kodierungen operieren würde.

7.2 Ressourcen-Identifikatoren

7.2.1 Zuweisung von Ressourcen-Identifikatoren

Jeder Feature-Instanz in einem WFS muss ein persistenter eindeutiger Ressourcen-Identifikator zugewiesen werden, wobei die Zuweisung durch den Server erfolgt, wenn das Feature erzeugt wird.

Dieser Identifikator muss unter allen WFS-Operationen, einschließlich Löschen, unveränderlich sein, was bedeutet, dass ein Ressourcen-Identifikator nicht wieder verwendet werden kann, nachdem er einmal zugewiesen wurde.

Ressourcen-Identifikatoren sind nicht dazu bestimmt, WFS-Ressourcen mit Objekten der realen Welt zu assoziieren, und die Werte brauchen außerhalb des Gültigkeitsbereichs einer Web-Feature-Service-Instanz keine Bedeutung zu haben.

7.2.2 Kodierung von Ressourcen-Identifikatoren

Für mit GML kodierte Features muss der Ressourcen-Identifikator im XML-Attribut gml:id kodiert sein. Diese Internationale Norm beschreibt nicht, wie Ressourcen-Identifikatoren für andere Ausgabeformate kodiert sind.

Innerhalb von Filterausdrücken können spezifische Feature-Instanzen mit dem Element fes:ResourceId identifiziert werden. Wenn der Server eine Versionierung unterstützt, können bestimmte Versionen eines Features mit dem in einem fes:ResourceId-Element (siehe ISO 19143:2010, 7.11.2) enthaltenen Attribut versionAction referenziert werden.

7.2.3 Versionsidentifizierung

Wenn der Server die Versionierung von Features unterstützt, muss er Versionsinformationen zu jeder Feature-Instanz vorhalten. In dieser Internationalen Norm wird keine Annahme dazu getroffen, wie diese Versionsinformationen vorgehalten werden. Die in der Norm zur Filterkodierung (siehe ISO 19143:2010, 7.11.2) definierten Funktionen erlauben eine Versionsnavigation auf der Basis des Ressourcen-Identifikators.

7.3 Eigenschaftsverweisungen

7.3.1 XPath-Teilmenge

GML ermöglicht Features komplexe oder aggregierte nicht geometrische Eigenschaften. Daraus ergibt sich das Problem, wie Komponenten des komplexen Wertes dieser Eigenschaften an den verschiedenen Stellen, an denen Wertverweisungen erforderlich sind (z. B. Abfrage- und Filterausdrücke) referenziert werden. Wenn der von einem WFS angebotene Feature-Inhalt mit Hilfe von XML kodiert ist, muss der WFS XPath-Ausdrücke (siehe W3C XML Path Language) zum Referenzieren der Eigenschaften und Komponenten

des Wertes der Eigenschaften eines Features verwenden. Die obligatorische mindestens erforderliche Teilmenge von XPath, die Server unterstützen müssen, ist in ISO 19143:2010, 7.4.4, beschrieben.

Die Unterstützung der Schema-Element()-XPath-Funktion ist optional. Wenn der Server jedoch der Konformitätsklasse „Inheritance“ entspricht (siehe Tabelle 1), dann muss er auch die Schema-Element()-Funktion unterstützen (siehe ISO 19143:2010, 7.4.4).

7.3.2 Zugriffsfunktion

In GML darf eine Feature-Eigenschaft ihren Wert als nach dem Inline-Verfahren kodierten Inhalt enthalten oder mit einer einfachen XLink-Verknüpfung referenzieren (siehe ISO 19136:2007, 7.2.3). Das bedeutet, dass im Laufe der Bewertung eines XPath-Ausdrucks ein Server möglicherweise eine Ressourcenverweisung auflösen muss. Um diese Anforderung zu unterstützen, müssen alle WFS-Implementierungen eine konkrete Implementierung einer XPath-Zugriffsfunktion mit dem Namen wfs:valueOf() bereitstellen.

Das Argument der Funktion muss der Name einer Eigenschaft eines Features sein, und die Antwort muss der Wert der Eigenschaft sein. Der Wert könnte einfach ein Textknoten oder eine Liste von Elementknoten sein, die der Wert der benannten Eigenschaft ist.

Die Funktion muss alle lokal referenzierten Ressourcen auflösen, und wenn der Server in seinem Fähigkeitendokument ankündigt, dass er abgesetzte Verweisungen auflösen kann (siehe Tabelle 13), muss die Funktion auch alle abgesetzten Ressourcenverweisungen auflösen. Wenn der Server nur lokal referenzierte Ressourcen unterstützt und auf eine abgesetzt referenzierte Ressource trifft, muss er eine „OptionNotSupported“-Ausnahme melden (siehe OGC 06-121r3:2009, Tabelle 25).

Wann die valueOf()-Funktion in einem XPath-Ausdruck zu verwenden ist, kann anhand des Anwendungsschemas des Servers festgestellt werden. Ist für eine Eigenschaft deklariert, dass sie Wertverweisungen zulässt (siehe ISO 19136:2007, 7.2.3.3 und 7.2.3.7), dann sollte die wfs:valueOf()-Funktion in einem XPath-Ausdruck festgelegt werden (siehe B.2.2 und B.4.5).

7.4 Kodierung von Prädikatausdrücken

Eine Anzahl von in dieser Internationalen Norm definierten Operationen enthält Prädikatausdrücke, die eine Teilmenge von Features identifizieren, auf denen zu operieren ist. Derartige Prädikatausdrücke können eine spezifische Menge von Features, auf denen zu operieren ist, enumerieren, oder eine Menge von Features kann durch Festlegen von Einschränkungen der Eigenschaften und/oder Komponenten des Wertes von Eigenschaften eines Feature-Typs definiert werden.

XML-kodierte Prädikatausdrücke müssen mit Hilfe des in ISO 19143:2010, 7.2, beschriebenen fes:Filter-Elementes kodiert werden.

KVP-kodierte Prädikatausdrücke müssen mit Hilfe der in ISO 19143:2010, Tabelle 2, beschriebenen Parameter definiert werden.

Die spezifische Menge von Prädikaten, die ein Web Feature Service implementiert, muss im Fähigkeitendokument des Servers im Filterfähigkeiten-Abschnitt (siehe 8.3.3) angegeben werden.

Sämtliche Implementierungen dieser Internationalen Norm müssen mindestens die Konformitätsklasse „Query“ (siehe ISO 19143:2010, Tabelle 1) implementieren.

Alle Implementierungen dieser Internationalen Norm, die die Konformitätsklasse „Basic WFS“ (siehe Tabelle 1) implementieren, müssen auch die Konformitätsklasse „Minimum Spatial Filter“ (siehe ISO 19143:2010, Tabelle 1) implementieren.

7.5 Ausnahmemeldungen

Trifft ein Web Feature Service beim Verarbeiten einer Anforderung auf einen Fehler oder empfängt er eine ungültige Anforderung, muss er ein XML-Dokument generieren, das angibt, dass ein Fehler aufgetreten ist. Das Format der XML-Fehlerantwort ist durch das in Abschnitt 8 der OWS Common Implementation

Specification (siehe OGC 06-121r3:2009) definierte Ausnahme-Antwortschema festgelegt und muss gegen dieses validieren.

Ein ows:ExceptionReport-Element darf eine oder mehrere WFS-Verarbeitungsausnahmen enthalten, die mit Hilfe des Elementes ows:Exception festgelegt werden. Das obligatorische „version“-Attribut (Versionsattribut) wird verwendet, um die Version des Dienstausnahme-Meldungsschemas anzugeben. Dieser Wert muss „2.0.0“ sein. Das optionale „language“-Attribut (Sprachattribut) darf verwendet werden, um die verwendete Sprache anzugeben. Die Codeliste für die Sprachparameter ist in IETF RFC 4646 definiert.

Individuelle Ausnahmenachrichten sind in dem ows:ExceptionText-Element enthalten. Das obligatorische „code“-Attribut (Codeattribut) muss verwendet werden, um einen Ausnahmecode mit der begleitenden Nachricht zu assoziieren.

Das optionale „locator“-Attribut (Lokalisatorattribut) darf verwendet werden um anzugeben, wo in der Anforderung auf die Ausnahme getroffen wurde, die den Fehler generiert hat. Tabelle 3 gibt an, welche Werte der Lokalisatorparameter für jeden Ausnahmecode haben sollte.

Tabelle 3 — WFS-Ausnahmecodes

exceptionCode-Werte	Bedeutung des Codes	„locator“-Wert	Konformitätsklasse
CannotLockAllFeatures	Eine Sperranforderung mit einer lockAction „ALL“ konnte nicht alle angeforderten Features sperren.	Wenn die Operation den optionalen Parameter „handle“ umfasst, ist der Wert als der Wert des Parameters „location“ zu melden.	Locking WFS
DuplicateStoredQueryIdValue	Der für einen gespeicherten Abfrageausdruck festgelegte Identifikator ist ein Duplikat.	Der „locator“-Parameter muss den Wert des zweifachen Identifikators enthalten.	Manage stored queries
DuplicateStoredQueryParameterName	Dieser festgelegte Name für einen gespeicherten Abfrageparameter wird bereits in derselben gespeicherten Abfragedefinition verwendet.	Der „locator“-Parameter muss den Namen des zweifachen gespeicherten Abfrageparameters auflisten.	Manage stored queries
FeaturesNotLocked	Bei Servern, die keine automatische Datensperre unterstützen (siehe 15.2.3.1) gibt diese Ausnahme an, dass eine „Transaction“-Operation Features modifiziert, die zuvor nicht mit einer LockFeature- (siehe Abschnitt 12) oder einer „GetFeatureWithLock“-Operation (siehe Abschnitt 13) gesperrt wurden.	Wenn die Operation den optionalen Parameter „handle“ umfasst, ist der Wert als der Wert des Parameters „location“ zu melden.	Locking WFS
InvalidLockId	Der Wert des „lockId“-Parameters auf einer „Transaction“-Operation ist ungültig, da er nicht vom Server generiert wurde.	Der „locator“-Parameter muss den Wert des ungültigen lockId enthalten.	Locking WFS
InvalidValue	Eine Transaktion (siehe Abschnitt 15) hat versucht, den Wert einer Datenkomponente so einzufügen oder zu verändern, dass gegen das Schema des Features verstoßen wird.	Der „locator“-Parameter muss den Namen der falsch modifizierten Eigenschaft enthalten.	Transactional WFS

Tabelle 3 (fortgesetzt)

exceptionCode-Werte	Bedeutung des Codes	„locator“-Wert	Konformitätsklasse
LockHasExpired	Der festgelegte Sperrenidentifikator auf einer „Transaction“- oder „LockFeature“-Operation ist abgelaufen und nicht länger gültig.	Der „locator“-Parameter muss den Wert des abgelaufenen Sperrenidentifikators enthalten.	Locking WFS
OperationParsingFailed	Die Abfrage ist falsch gebildet und wird vom Server nicht geparkt.	Der „locator“-Parameter muss den Wert des „handle“-Parameters enthalten, sofern verfügbar. Andernfalls muss der „locator“-Parameter den Namen der falsch gebildeten Operation enthalten.	Alle (siehe Tabelle 1)
OperationProcessingFailed	Beim Verarbeiten der Operation ist ein Fehler aufgetreten.	Der „locator“-Parameter muss den Wert des „handle“-Parameters enthalten, sofern verfügbar. Andernfalls muss der „locator“-Parameter den Namen der fehlgeschlagenen Operation enthalten.	Alle (siehe Tabelle 1)
ResponseCacheExpired	Der zur Unterstützung des Paging verwendete Antwort-Cache ist abgelaufen und die Ergebnisse sind nicht mehr verfügbar.	Wenn die Operation den optionalen Parameter „handle“ umfasst, ist der Wert als der Wert des Parameters „locator“ zu melden.	Response paging

Server müssen für alle Konformitätsklassen dieser Internationalen Norm die generischen Ausnahmecodes nach Tabelle 25 von OGC 06-121r3:2009 implementieren.

In einer einzelnen Ausnahmemeldung dürfen mehrere Ausnahmen angegeben werden, daher sollten Serverimplementierungen so viele Ausnahmen melden, wie zur eindeutigen Beschreibung eines Problems erforderlich.

BEISPIEL Schlägt das Parsen einer Operation fehl, weil der Wert eines Parameters ungültig ist, sollte der Server eine „OperationParsingFailed“-Ausnahme und eine „InvalidParameterValue“-Ausnahme melden.

Anhang B enthält Beispiele für Ausnahmemeldungen.

7.6 Gemeinsame Anforderungsparameter

7.6.1 Einleitung

Anforderungsparameter, die allen oder mehreren in dieser Internationalen Norm definierten Operationen gemeinsam sind, sind in 7.6.2 bis 7.6.6 beschrieben.

7.6.2 Basisanforderungstyp

7.6.2.1 Anforderungssemantik

Der Basisanforderungstyp („BaseRequest“-Typ, siehe Bild 3) ist ein abstrakter Typ, dem alle WFS-Operationen, ausgenommen die Operation „GetCapabilities“, untergeordnet sind.

<i>BaseRequest</i>
+ service : CharacterString = "WFS" {frozen}
+ version : CharacterString = "2.0.0" {frozen}
+ handle [0..1] : CharacterString

Bild 3 — BaseRequest

7.6.2.2 XML-Kodierung

Das folgende XML-Schemafragment legt die XML-Kodierung des Typs „BaseRequest“ fest:

```
<xsd:complexType name="BaseRequestType" abstract="true">
  <xsd:attribute name="service" type="xsd:string"
    use="required" fixed="WFS"/>
  <xsd:attribute name="version" type="xsd:string"
    use="required" fixed="2.0.0"/>
  <xsd:attribute name="handle" type="xsd:string"/>
</xsd:complexType>
```

7.6.2.3 KVP-Kodierung

Tabelle 4 definiert die KVP-Kodierung des „Base Request“-Typs.

Der Wert des obligatorischen REQUEST-Schlüsselworts muss angeben, welche Dienstoperation aufgerufen wird.

ANMERKUNG XML-kodierte Anforderungen besitzen keinen REQUEST-Parameter, da der Name des Wurzelelementes den Namen der aufgerufenen Dienstoperation kodiert.

Tabelle 4 — KVP-Kodierung des „Base Request“-Typs

URL-Komponente	Operation	O/M ^a	Beschreibung
SERVICE	alle Operationen	M	siehe 7.6.2.4
VERSION ^b (alle Operationen)	alle Operationen außer GetCapabilities	M	siehe 7.6.2.5
^a O = optional, M = obligatorisch (en: mandatory).			
^b VERSION ist für alle Operationen außer GetCapabilities verbindlich.			

7.6.2.4 „service“-Parameter

In XML muss dieser Parameter mit einem Attribut mit dem Namen „service“ kodiert werden (siehe 7.6.2.2).

Bei der KVP-Kodierung muss dieser Parameter mit dem SERVICE-Schlüsselwort kodiert werden (siehe 7.6.2.3).

Der obligatorische „service“-Parameter ist zu verwenden, um anzugeben, welcher der verfügbaren Dienstypen an einem bestimmten Server aufgerufen wird. Beim Aufruf eines Web Features Service muss der Wert des „service“-Parameters „WFS“ sein.

7.6.2.5 „version“-Parameter

In XML muss dieser Parameter mit einem Attribut mit dem Namen „version“ kodiert werden (siehe 7.6.2.2).

Bei KVP-Kodierung muss dieser Parameter mit dem VERSION-Schlüsselwort kodiert werden (siehe 7.6.2.3).

Alle WFS-Anforderungen (außer die Operation „GetCapabilities“) müssen einen Parameter mit dem Namen „version“ enthalten.

Der „version“-Parameter ist zu verwenden um anzugeben, mit welcher Version der WFS-Spezifikation die Anforderungskodierung übereinstimmt, und er wird in den in 6.2.3 beschriebenen Versionsverhandlungen verwendet. Bei der Kodierung einer WFS-Anforderung nach dieser Internationalen Norm ist der Wert des „version“-Attributs auf 2.0.0 festzulegen, was der Versionsnummer dieser Internationalen Norm entspricht.

7.6.2.6 „handle“-Parameter

In XML muss dieser Parameter mit einem Attribut mit dem Namen „handle“ kodiert werden (siehe 7.6.2.2).

Dieser Parameter ist für eine KVP-Kodierung nicht definiert.

Der „handle“-Parameter kann optional auf einer Anforderung festgelegt werden.

Zweck des optionalen „handle“-Parameters ist es, einer Client-Anwendung zu erlauben, einen mnemonischen Namen mit einer Anforderung zu Fehlerbehandlungszwecken zu assoziieren.

Wird für eine Operation ein „handle“ festgelegt und tritt eine Ausnahme auf (siehe 7.5) beim Ausführen der Operation, muss ein Web Feature Service beim Verarbeiten der Operation dem „locator“-Attribut im ows:ExceptionText-Element (siehe OGC 06-121r3:2009, Abschnitt 8) den Wert des „handle“-Attributs zuweisen, um die Operation oder Aktion zu identifizieren, die die Ausnahme generiert hat. Ist kein „handle“ festgelegt, kann der Server das „locator“-Attribut im ows:ExceptionText-Element weglassen oder ein anderes Mittel, zum Beispiel Zeilennummern, nutzen, um die Ausnahme innerhalb der Operation aufzufinden. Das „handle“-Attribut ist besonders nützlich, wenn es zusammen mit der „Transaction“-Operation (siehe Abschnitt 15) verwendet wird, die viele Aktionen enthalten kann. Das Festlegen eines „handle“-Attributs für jede Aktion erlaubt dem Server das genaue Auffinden einer Ausnahme innerhalb einer „Transaction“-Operation.

7.6.3 Standardpräsentationsparameter

7.6.3.1 Parametersemantik

Standardpräsentationsparameter (en: standard presentation parameters, siehe Bild 4) werden verwendet, um zu steuern, wie Abfrageergebnisse in einem Antwortdokument präsentiert werden. Diese Parameter können in den Operationen „GetPropertyValue“ (siehe Abschnitt 10), „GetFeature“ (siehe Abschnitt 11) und „GetFeatureWithLock“ (siehe Abschnitt 13) vorkommen.

<i>StandardPresentationParameters</i>	<<Enumeration>> ResultType
+ resultType : ResultType = #results	+ results
+ outputFormat : CharacterString = "application/gml+xml; version=3.2"	+ hits
+ count[0..1] : Integer	
+ startIndex : Integer = 0	

Bild 4 — StandardPresentationParameters

7.6.3.2 XML-Kodierung

Das folgende Fragment definiert die XML-Kodierung der Standardpräsentationsparameter:

```
<xsd:attributeGroup name="StandardPresentationParameters">
  <xsd:attribute name="startIndex"
    type="xsd:nonNegativeInteger" default="0"/>
  <xsd:attribute name="count"
    type="xsd:nonNegativeInteger"/>
  <xsd:attribute name="resultType"
    type="wfs:ResultTypeType" default="results"/>
  <xsd:attribute name="outputFormat"
    type="xsd:string" default="application/gml+xml; version=3.2"/>
</xsd:attributeGroup>
<xsd:simpleType name="ResultTypeType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="results"/>
    <xsd:enumeration value="hits"/>
  </xsd:restriction>
</xsd:simpleType>
```

7.6.3.3 KVP-Kodierung

Tabelle 5 definiert die KVP-Kodierung der Standardpräsentationsparameter.

Tabelle 5 — KVP-Kodierung von Standardpräsentationsparametern

URL-Komponente	Operation	O/M ^a	Vordefiniert	Beschreibung
STARTINDEX	GetPropertyValue, GetFeature, GetFeatureWithLock	O	1	siehe 7.6.3.4
COUNT	GetPropertyValue, GetFeature, GetFeatureWithLock	O	1	siehe 7.6.3.5
OUTPUTFORMAT	DescribeFeatureType, GetPropertyValue, GetFeature, GetFeatureWithLock	O	application/gml+xml; version=3.2	siehe 7.6.3.7
RESULTTYPE	GetPropertyValue, GetFeature, GetFeatureWithLock	O	results	siehe 7.6.3.6.
^a O = optional, M = obligatorisch (en: mandatory)				

7.6.3.4 „startIndex“-Parameter

Bei XML-kodierten Anforderungen muss dieser Parameter mit einem Attribut mit dem Namen „startIndex“ kodiert werden (siehe 7.6.3.2).

Bei KVP-kodierten Anforderungen muss dieser Parameter mit dem STARTINDEX-Schlüsselwort kodiert werden (siehe 7.6.3.3).

Der optionale Parameter „startIndex“ gibt den Index in der Ergebnismenge an, aus der der Server mit der Präsentation der Ergebnisse im Antwortdokument beginnen muss.

7.6.3.5 „count“-Parameter

Bei XML-kodierten Anforderungen muss dieser Parameter mit einem Attribut mit dem Namen „count“ kodiert werden (siehe 7.6.3.2).

Bei KVP-kodierten Anforderungen muss dieser Parameter mit dem COUNT-Schlüsselwort kodiert werden (siehe 7.6.3.3).

Der optionale Parameter „count“ begrenzt die Anzahl explizit angeforderter Werte (d. h. Features oder Eigenschaftswerte), die in einem Antwortdokument präsentiert werden.

Im Zähler sind nur Werte der Typen zu zählen, die explizit als der Wert des „typeNames“-Parameters angefordert wurden. Verschachtelte Werte, die in den explizit angeforderten Werttypen enthalten sind, dürfen nicht gezählt werden.

BEISPIEL Der Zähler von Features im folgenden XML-Fragment ist 4. Die eingebettete Verweisung auf das Feature mit gml:id="2" und das Feature mit gml:id="5", die der Wert abc:Prop4 ist, ist nicht im Zähler enthalten.

```
<?xml version="1.0"?>
<wfs:FeatureCollection
  timeStamp="2010-08-01T22:47:02"
  numberMatched="4" numberReturned="4"
  xmlns:myns="http://www.someserver.com/myns"
```

```
xmlns:wfs="http://www.opengis.net/wfs/2.0"
xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.someserver.com/myns ./myns.xsd
                    http://www.opengis.net/wfs/2.0
                    http://schemas.opengis.net/wfs/2.0.0/wfs.xsd
                    http://www.opengis.net/gml/3.2
                    http://schemas.opengis.net/gml/3.2.1/gml.xsd">

  <wfs:member>
    <myns:Feature gml:id="1">
      ...
    </myns:Feature>
  </wfs:member>
  <wfs:member>
    <myns:Feature gml:id="2">
      ...
    </myns:Feature>
  </wfs:member>
  <wfs:member>
    <myns:Feature gml:id="3">
      ...
    </myns:Feature>
  </wfs:member>
  <wfs:member>
    <myns:Feature gml:id="4">
      <myns:Property1> ... <myns:Property1>
      <myns:Property2> ... <myns:Property2>
      <myns:Property3 xlink:href="#2"/>
      <myns:Property4>
        <myns:Feature gml:id="5">
          ...
        </myns:Feature>
      </myns:Property4>
    </myns:Feature>
  </wfs:member>
</wfs:FeatureCollection>
```

Bei einer Verknüpfungsabfrage (siehe 7.9.2.5.3.1) ist jedes Tupel explizit angeforderter Werttypen als Eins im Zähler zu zählen. Verschachtelte Wertmitglieder, die in einem der explizit angeforderten Werttypen des Tupels enthalten sind, zählen nicht.

Der Zählwert gilt für die gesamte Ergebnismenge (d. h. die Ergebnismenge, die durch Verarbeiten einer oder mehrerer „Query“-Aktionen generiert wird), und die Einschränkung muss auf die Werte in der Reihenfolge angewendet werden, in der diese präsentiert werden. Wenn die Zählgrenze erreicht ist, darf die Anforderungsverarbeitung enden und das Antwortdokument, das höchstens Zählwerte enthält, ist dem Client zu präsentieren.

Für den „count“-Parameter gibt es keinen vordefinierten Standardwert, und das Fehlen des Parameters muss bedeuten, dass alle Werte in der Ergebnismenge dem Client in Abhängigkeit von serverkonfigurierten Werten präsentiert werden. Besitzt der Server eine serverkonfigurierte Zählgrenze, muss diese Grenze mit Hilfe der „CountDefault“-Einschränkung (siehe Tabelle 14) im Fähigkeitendokument des Servers angezeigt werden.

7.6.3.6 „resultType“-Parameter

Bei XML-kodierten Anforderungen muss dieser Parameter mit dem Attribut mit dem Namen „resultType“ kodiert werden (siehe 7.6.3.2).

Bei KVP-kodierten Anforderungen muss dieser Parameter mit dem RESULTTYPE-Schlüsselwort kodiert werden (siehe 7.6.3.3).

Ein WFS kann auf eine Abfrageoperation auf zwei Arten antworten (außer mit einer Ausnahme-Antwort). Er kann entweder ein vollständiges Antwortdokument generieren, das Ressourcen enthält, die die Operation erfüllen, oder er kann einfach einen leeren Antwortcontainer generieren, der die Zählung der Gesamtanzahl von Ressourcen angibt, die die Operation zurückgeben würde. Welche dieser beiden Antworten ein WFS generiert, wird durch den Wert des optionalen „resultType“-Parameters bestimmt.

Die möglichen Werte für diesen Parameter sind „results“ (Ergebnisse) und „hits“ (Treffer).

Ist der Wert des „resultType“-Parameters auf „results“ gesetzt, muss der Server ein vollständiges Antwortdokument generieren, das die Ressourcen enthält, die die Operation erfüllen. Das Wurzelement des Antwortcontainers muss eine Zählung der Anzahl der im Antwortdokument tatsächlich präsentierten Ressourcen umfassen (7.7.4.3). Das Wurzelement des Antwortcontainers muss außerdem eine Zählung der Gesamtanzahl der von den Operationen tatsächlich gefundenen Ressourcen umfassen, welche immer gleich der Anzahl der im Antwortdokument präsentierten Ressourcen oder größer als diese ist (7.7.4.2).

Wenn der Wert des „resultType“-Attributs auf „hits“ gesetzt ist, muss der Server ein leeres Antwortdokument generieren, das keine Ressourceninstanzen enthält, und das Wurzelement des Antwortcontainers muss die Zählung der Gesamtanzahl der von der Operation gefundenen Ressourcen enthalten (siehe 7.7.4.2). Der Wert für die Anzahl der im Antwortdokument präsentierten Ressourcen (siehe 7.7.4.3) ist auf Null zu setzen.

7.6.3.7 „outputFormat“-Parameter

Bei XML-kodierten Anforderungen muss dieser Parameter mit einem Attribut mit dem Namen „outputFormat“ kodiert werden (siehe 7.6.3.2).

Bei KVP-kodierten Anforderungen muss dieser Parameter mit dem OUTPUTFORMAT-Schlüsselwort kodiert werden (siehe 7.6.3.3).

Der optionale Parameter „outputFormat“ legt das Format fest, das zur Kodierung von Ressourcen in der Antwort auf eine Abfrageoperation verwendet wird. Der Standardwert ist „application/gml+xml; version=3.2“ und gibt an, dass Ressourcen im Antwortdokument mit GML zu kodieren sind (siehe ISO 19136).

Jeder mit dieser Internationalen Norm übereinstimmende WFS muss diesen Standardwert unterstützen.

Ein Server darf zusätzliche Werte für den Parameter „outputFormat“ in seinem Fähigkeitendokument (siehe 8.3.3) anzeigen, wobei anzugeben ist, dass mehrere Ausgabeformate, einschließlich älterer GML-Versionen, unterstützt werden. Diese Internationale Norm weist diesen zusätzlichen Werten jedoch keine spezielle Bedeutung zu. Sind im Fähigkeitendokument des Servers zusätzliche „outputFormat“-Werte festgelegt, empfiehlt diese Internationale Norm die Aufnahme eines beschreibenden Kommentars für jeden aufgelisteten Wert.

7.6.4 Standardauflösungsparameter

7.6.4.1 Parametersemantik

Server, die mit dieser Internationalen Norm übereinstimmen, müssen die Fähigkeit zur Auflösung lokaler Ressourcenverweise implementieren.

Optional können Server die Fähigkeit implementieren, auf Anforderung Verweise abgesetzter Ressourcen aufzulösen und müssen diese Fähigkeit mit der „ImplementsRemoteResolve“-Einschränkung in ihrem Fähigkeitendokument anzeigen (siehe Tabelle 13).

Wie Ressourcenverweise von einem Server behandelt werden, wird durch die in 7.6.4.2 bis 7.6.4.7 beschriebenen Parameter „resolve“, „resolveDepth“ und „resolveTimeout“ gesteuert.

Die Standardauflösungsparameter (en: standard resolve parameters, siehe Bild 5) können in den Operationen „GetPropertyValue“ (siehe Abschnitt 10), „GetFeature“ (siehe Abschnitt 11) und „GetFeatureWithLock“ (siehe Abschnitt 13) vorkommen.

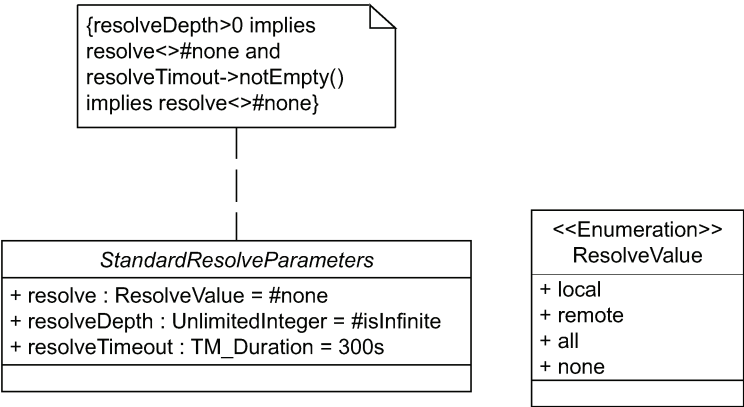


Bild 5 — StandardResolveParameters

7.6.4.2 XML-Kodierung

Das folgende Fragment definiert die XML-Kodierung für die Standardauflösungsparameter:

```
<xsd:attributeGroup name="StandardResolveParameters">
  <xsd:attribute name="resolve"
    type="wfs:ResolveValueType" default="none"/>
  <xsd:attribute name="resolveDepth"
    type="wfs:positiveIntegerWithStar" default="*"/>
  <xsd:attribute name="resolveTimeout"
    type="xsd:positiveInteger" default="300"/>
</xsd:attributeGroup>
<xsd:simpleType name="ResolveValueType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="local"/>
    <xsd:enumeration value="remote"/>
    <xsd:enumeration value="all"/>
    <xsd:enumeration value="none"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="positiveIntegerWithStar">
  <xsd:union memberTypes="xsd:positiveInteger wfs:StarStringType"/>
</xsd:simpleType>
<xsd:simpleType name="StarStringType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="*"/>
  </xsd:restriction>
</xsd:simpleType>
```

7.6.4.3 KVP-Kodierung

Tabelle 6 definiert die KVP-Kodierung der Standardauflösungsparameter.

Tabelle 6 — KVP-Kodierung von Standardauflösungsparametern

URL-Komponente	Operation	O/M ^a	Vordefiniert	Beschreibung
RESOLVE	GetPropertyValue, GetFeature, GetFeatureWithLock	O	none	Siehe 7.6.4.4.
RESOLVEDDEPTH	GetPropertyValue, GetFeature, GetFeatureWithLock	O	*	Siehe 7.6.4.5. RESOLVE-Parameter muss einen anderen Wert als „none“ haben.
RESOLVETIMEOUT	GetPropertyValue, GetFeature, GetFeatureWithLock	O	serverspezifisch (siehe ResolveTimeoutDefault, Tabelle 14)	Siehe 7.6.4.6. RESOLVE-Parameter muss einen anderen Wert als „none“ haben.
^a O = optional, M = obligatorisch (en: mandatory)				

7.6.4.4 „resolve“-Parameter

Bei XML-kodierten Anforderungen muss dieser Parameter mit einem Attribut mit dem Namen „resolve“ kodiert werden (siehe 7.6.4.2).

Bei KVP-kodierten Anforderungen muss dieser Parameter mit dem RESOLVE-Schlüsselwort kodiert werden (siehe 7.6.4.3).

Der optionale Parameter „resolve“ steuert, ob Ressourcenverweise von einer Operation aufgelöst werden und wenn ja, welche (d. h. lokale oder abgesetzte).

Der Wertebereich des „resolve“-Parameters ist: „local“ (lokal), „remote“ (abgesetzt), „all“ (alle) oder „none“ (keine). Je nachdem, welche Fähigkeiten ein Server implementiert, kann er einen Teil oder den gesamten Wertebereich unterstützen. Welche Auflösungsparameterwerte ein Server unterstützt, ist im Fähigkeitsdokument des Servers anzuzeigen (siehe 8.3.3).

Der Auflösungsparameterwert „local“ bedeutet, dass eine Operation nur lokale Verweise auflösen darf.

Der Auflösungsparameterwert „remote“ bedeutet, dass eine Operation nur abgesetzte Ressourcenverweise auflösen darf.

Der Auflösungsparameterwert „all“ bedeutet, dass eine Operation nur alle Ressourcenverweise auflösen darf.

Der Auflösungsparameterwert „none“ bedeutet, dass eine Operation keine Ressourcenverweise auflösen darf. Hierbei handelt es sich auch um den Standardwert, wenn der Auflösungsparameter nicht festgelegt ist.

7.6.4.5 „resolveDepth“-Parameter

Bei XML-kodierten Anforderungen muss dieser Parameter mit einem Attribut mit dem Namen „resolveDepth“ kodiert werden (siehe 7.6.4.2).

Bei KVP-kodierten Anforderungen muss dieser Parameter mit dem RESOLVEDDEPTH-Schlüsselwort kodiert werden (siehe 7.6.4.3).

Der optionale Parameter „resolveDepth“ gibt die Tiefe an, mit der verschachtelte Ressourcenverweise in einem Antwortdokument aufzulösen sind.

Der Bereich gültiger Werte für diesen Parameter besteht aus nicht negativen ganzen Zahl zuzüglich „*“.

Ein für den Parameter „resolveDepth“ festgelegter Wert darf nur verwendet werden, wenn der Auflösungsparameter (siehe 7.6.4.4) auf einer Operation oder Aktion festgelegt ist, bei der der „resolveDepth“-Parameter vorkommt und der Wert des Auflösungsparameters nicht auf „none“ gesetzt ist.

Wenn ein Wert für den Auflösungsparameter nicht festgelegt ist oder wenn der Wert des Auflösungsparameters auf „none“ gesetzt ist, muss der Server jeden für den „resolveDepth“-Parameter festgelegten Wert ignorieren.

Wenn der Wert des „resolveDepth“-Parameters als „0“ festgelegt ist, darf ein Server keine Ressourcenverweise auflösen.

Wenn der Wert des „resolveDepth“-Parameters als „1“ festgelegt ist, muss ein Server die unmittelbaren Ressourcenverweise auflösen und ihre Werte in das Antwortdokument mit aufnehmen. Enthalten diese aufgelösten Ressourcen jedoch verschachtelte Ressourcenverweise, dürfen diese verschachtelten Verweise nicht aufgelöst werden.

Ist der Wert des „resolveDepth“-Parameters als „*“ festgelegt, muss ein Server alle unmittelbaren Ressourcenverweise und alle verschachtelten Ressourcenverweise auflösen.

Nach Auflösen der Verweise mit der festgelegten Auflösungstiefe (resolveDepth) muss ein Server dafür sorgen, dass tiefer verschachtelte Verweise so verschoben werden, dass sie auf ihre vorgesehenen Ressourcen zeigen.

BEISPIEL 1 Dazu kann es erforderlich sein, dass die Verweise auf lokale Ressourcen im Datenspeicher eines WFS so umgewandelt werden, dass sie in einem Antwortdokument eine Fernverweisung zurück auf denselben Server vornehmen.

Für den Fall, dass vom Server ein Schleifenverweis festgestellt wird, bevor die festgelegte resolveDepth erreicht ist, muss der Server die weitere Auflösung verschachtelter Ressourcen abbrechen und dafür sorgen, dass der letzte Verweis in der Kette die entsprechende bereits aufgelöste Ressource referenziert.

BEISPIEL 2 Zu betrachten sei die folgende Kette von Ressourcenverweisen: $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow B$. Wenn der „resolveDepth“-Wert auf 8 gesetzt ist, würde ein Server alle Ressourcen bis E auflösen (d. h. 4 Ebenen) und anschließend dafür Sorge tragen, dass Ressource E auf die bereits aufgelöste Ressource B zeigt. An dieser Stelle muss der Server mit der Ressourcenauflösung aufhören, da die verbleibenden Ressourcen bis zu einer Tiefe von 8 Ebenen aufgrund des Schleifenverweises bereits aufgelöst wurden.

7.6.4.6 „resolveTimeout“-Parameter

Bei XML-kodierten Anforderungen muss dieser Parameter mit einem Attribut mit dem Namen „resolveTimeout“ kodiert werden (siehe 7.6.4.2).

Bei KVP-kodierten Anforderungen muss dieser Parameter mit dem RESOLVETIMEOUT-Schlüsselwort kodiert werden (siehe 7.6.4.3).

Der optionale Parameter „resolveTimeout“ steuert, wie lange ein Server bei der Auflösung von Ressourcenverweisen auf den Empfang einer Antwort warten muss.

Der „resolveTimeout“-Parameter ist vom Typ xsd:positiveInteger und legt die Ablaufzeit in Sekunden fest. Ist der „resolveTimeout“-Parameter nicht festgelegt, ist die Wartezeit des Servers implementierungsabhängig und muss im Fähigkeitendokument des Servers mit der „ResolveTimeoutDefault“-Einschränkung angezeigt werden (siehe Tabelle 14).

Ein für den Parameter „resolveTimeout“ festgelegter Wert darf nur verwendet werden, wenn der Auflösungsparameter auf dem Abfrageelement festgelegt ist, das den „resolveTimeout“-Parameter enthält, und wenn der Wert des Auflösungsparameters nicht auf „none“ gesetzt ist.

Ist der Wert des Auflösungsparameters auf „none“ gesetzt, muss der Server für den „resolveTimeout“-Parameter festgelegte Werte ignorieren.

7.6.4.7 Nicht auflösbare Verweise

Kann ein Server einen Ressourcenverweis nicht auflösen, muss er einfach den ursprünglichen, nicht aufgelösten URI in der Antwort angeben. Dies gilt nicht als Ausnahme.

7.6.5 Standardeingabeparameter

7.6.5.1 Parametersemantik

Standardeingabeparameter (en: standard input parameters, siehe Bild 6) sind eine Gruppe von Parametern, die verwendet werden, um die Kodierung von Ressourcen bei Eingabe sowie des CRS von Geometriewerten, die diese Ressourcen enthalten könnten, zu bestätigen.

Diese Parameter können auf den Aktionen „Insert“ (siehe 15.2.4), „Update“ (siehe 15.2.5) und „Replace“ (siehe 15.2.6) der Operation „Transaction“ (siehe Abschnitt 15) festgelegt werden.

<i>StandardInputParameters</i>
+ srsName [0..1] : SC_CRS
+ inputFormat : CharacterString = "application/gml+xml; version=3.2"

Bild 6 — StandardInputParameters

7.6.5.2 XML-Kodierung

Das folgende Fragment definiert die XML-Kodierung für die Standardeingabeparameter.

```
<xsd:attributeGroup name="StandardInputParameters">
  <xsd:attribute name="inputFormat" type="xsd:string"
    default="application/gml+xml; version=3.2"/>
  <xsd:attribute name="srsName" type="xsd:anyURI"/>
</xsd:attributeGroup>
```

7.6.5.3 KVP-Kodierung

Die Standardeingabeparameter sind für KVP-kodierte Anforderungen nicht definiert, da eine KVP-Kodierung für die „Transaction“-Operation nicht definiert ist (siehe Abschnitt 15).

7.6.5.4 „inputFormat“-Parameter

Bei XML-kodierten Anforderungen muss der „inputFormat“-Parameter mit einem Attribut mit dem Namen „inputFormat“ kodiert werden (siehe 7.6.5.2).

Der „inputFormat“-Parameter ist für KVP-kodierte Anforderungen nicht definiert.

Der Parameter „inputFormat“ darf verwendet werden, um die Feature-Kodierung zu bestätigen, die zum Ausdrücken von Features bei Eingabe mit Hilfe der Aktion „Insert“ der „Transaction“-Operation verwendet wird (siehe 15.2.4.1), oder wenn Features mit den Aktionen „Update“ (siehe 15.2.5.1) oder „Replace“ (siehe 15.2.6.1) aktualisiert werden.

Bei Servern, die die Konformitätsklasse „Transactional WFS“ (siehe Tabelle 1) implementieren, muss der Standardwert des Attributs „inputFormat“ „application/gml+xml; version=3.2“ sein, der angibt, dass Eingabe-Features mit GML kodiert sind (siehe ISO 19136).

Ein Server darf für den „inputFormat“-Parameter in seinen Fähigkeitendokumenten (siehe 8.3.3) zusätzliche Werte anzeigen, der angibt, dass mehrere Eingabeformate, einschließlich früherer GML-Versionen, unterstützt werden. Diese Internationale Norm weist diesen zusätzlichen Werten jedoch keine spezielle Bedeutung

zu. In Fällen, in denen zusätzliche „inputFormat“-Werte im Fähigkeitendokument des Servers festgelegt sind, empfiehlt diese Internationale Norm die Aufnahme eines beschreibenden Kommentars für jeden aufgelisteten Wert.

7.6.5.5 „srsName“-Parameter

Bei XML-kodierten Anforderungen muss der „srsName“-Parameter mit einem Attribut mit dem Namen „srsName“ kodiert werden.

Der Standardeingabeparameter „srsName“ ist für KVP-kodierte Anforderungen nicht definiert.

Der Parameter „srsName“ kann als ein Parameter der „Transaction“-Operation (siehe Abschnitt 15) oder als ein Parameter auf den Transaktionsaktionen „Insert“ (siehe 15.2.4.1), „Update“ (siehe 15.2.5.1) oder „Replace“ (siehe 15.2.6.1) festgelegt werden.

Bei Festlegung als ein Parameter der „Transaction“-Operation muss der Wert des Attributs „srsName“ das zur Kodierung von Feature-Geometrien in der Transaktion verwendete Standard-CRS bestätigen.

Ist ein „srsName“-Wert als ein Parameter der Transaktionsaktionen „Insert“, „Update“ oder „Replace“ festgelegt, muss der Wert jeden CRS-Wert ersetzen, der mit dem Parameter „srsName“ auf der „Transaction“-Operation festgelegt wird.

Der Parameter „srsName“ (siehe ISO 19136:2007, 10.1.3.1) kann auch verwendet werden, um das CRS einer einzelnen Geometrie zu bestätigen und ersetzt alle früheren CRS-Bestätigungen.

Sofern festgelegt, muss der Wert des „srsName“-Parameters gleich dem mit dem „wfs:DefaultCRS-Element“ im Fähigkeitendokument (siehe 8.3.3) festgelegten Standard-CRS-Wert sein oder gleich jedem der „wfs:OtherCRS“-Werte (siehe Tabelle 11) für den jeweiligen Feature-Typ.

Wird das festgelegte CRS für den festgelegten Feature-Typ nicht unterstützt, muss der WFS eine „InvalidParameterValue“-Ausnahme generieren (siehe Tabelle 3).

Ist der Parameter „srsName“ nicht festgelegt, muss der WFS dies in der Bedeutung interpretieren, dass Geometrien im Standard-CRS wie mit Hilfe des Elementes „wfs:DefaultCRS“ im Fähigkeitendokument (siehe 8.3.3) festgelegt kodiert werden.

7.6.6 Zusätzliche gemeinsame Schlüsselwörter für KVP-kodierte Anforderungen

Tabelle 7 definiert zusätzliche Schlüsselwörter für KVP-kodierte WFS-Anforderungen.

Tabelle 7 — Zusätzliche gemeinsame Schlüsselwörter für KVP-kodierte WFS-Anforderungen

URL-Komponente	Operation	O/M ^a	Beschreibung
NAMESPACES	Alle	O	Wird verwendet, um Namensräume und deren Präfixe festzulegen. Das Format muss <i>xmlns(prefix,escaped_url)</i> sein, wo <i>escaped_url</i> in OGC 06-121r3:2009, 11.3 definiert ist. Wenn das Präfix nicht festgelegt ist, ist der Standardnamensraum anzunehmen. Mehrere Namensräume können durch Festlegung einer durch Kommas getrennten Liste von <i>xmlns()</i>-Werten gebunden werden. Dieser Parameter ist nicht für XML-kodierte Anforderungen definiert, da XML über einen anderen Mechanismus zum Bestätigen von Namensräumen verfügt (siehe 6.3).
VSP		O	Ein Server kann zusätzliche KVP-Parameter implementieren, die nicht Teil dieser Internationalen Norm sind. Diese sind als herstellereigenspezifische Parameter VSP bekannt. Mit VSP können Hersteller zusätzliche Parameter festlegen, die die Ergebnisse von Anforderungen verbessern. Ein Server muss gültige Ergebnisse auch dann erzeugen, wenn die VSP fehlen oder falsch gebildet sind oder wenn VSP geliefert werden, die dem Server nicht bekannt sind. Unbekannte VSP sind zu ignorieren. Ein Server kann sich dafür entscheiden, einige oder alle VSP nicht anzuzeigen. Sind VSP in der „Capabilities“-XML enthalten (siehe 8.3), muss das Element <i>ows:ExtendedCapabilities</i> (siehe OGC 06-121r3:2009, 7.4.6) entsprechend erweitert werden. Es können zusätzliche Schemadokumente importiert werden, die die Erweiterung(en) des <i>ows:ExtendedCapabilities</i>-Elementes enthalten. Angezeigte VSP müssen zusätzliche Metadaten enthalten oder referenzieren, die ihre Bedeutung beschreiben (siehe 8.4). WFS-Implementierer sollten VSP-Namen mit Sorgfalt wählen, um Konflikte mit in dieser Internationalen Norm definierten WFS-Parametern zu vermeiden.
^a O = optional, M = obligatorisch (en: mandatory)			

7.7 Standardantwortparameter

7.7.1 Parametersemantik

Standardantwortparameter (en: standard response parameters, siehe Bild 7) sind Antwortparameter, die in der Definition der Antwortsammlung für die Operationen „GetPropertyValues“ (siehe Abschnitt 10), „GetFeature“ (siehe Abschnitt 11) und „GetFeatureWithLock“ (siehe Abschnitt 13) verwendet werden.

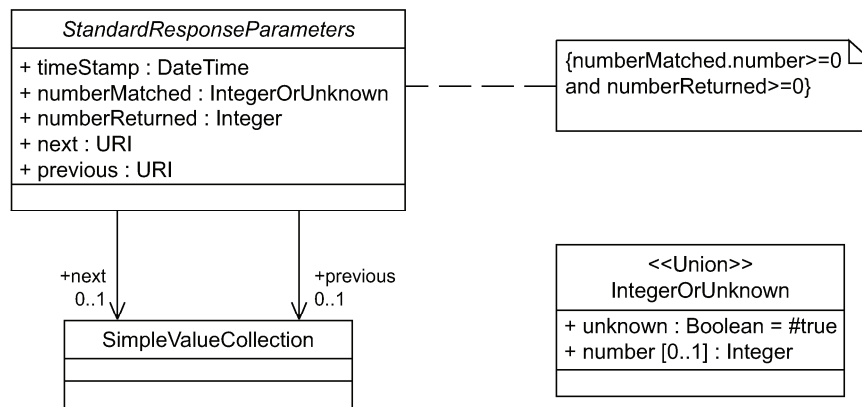


Bild 7 — StandardResponseParameters

7.7.2 XML-Kodierung

Das folgende XML-Schemafragment definiert die XML-Kodierung für die Standardantwortparameter:

```

<xsd:attributeGroup name="StandardResponseParameters">
  <xsd:attribute name="timeStamp" type="xsd:dateTime" use="required"/>
  <xsd:attribute name="numberMatched" type="xsd:nonNegativeIntegerOrUnknown" use="required"/>
  <xsd:attribute name="numberReturned" type="xsd:nonNegativeInteger" use="required"/>
  <xsd:attribute name="next" type="xsd:anyURI"/>
  <xsd:attribute name="previous" type="xsd:anyURI"/>
</xsd:attributeGroup>
<xsd:simpleType name="nonNegativeIntegerOrUnknown">
  <xsd:union>
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="unknown"/>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType>
      <xsd:restriction base="xsd:nonNegativeInteger"/>
    </xsd:simpleType>
  </xsd:union>
</xsd:simpleType>

```

7.7.3 KVP-Kodierung

Für WFS-Antwortsammlungen ist nur eine XML-Kodierung definiert.

7.7.4 Erläuterung der Parameter

7.7.4.1 „timeStamp“-Parameter

Der Parameter „timeStamp“ ist von einem WFS zu verwenden, um die Zeit und das Datum anzugeben, zu der/dem eine Antwortsammlung generiert wurde.

7.7.4.2 „numberMatched“-Parameter

Der Parameter „numberMatched“ ist in einem Antwortdokument zu verwenden, um die Gesamtanzahl von Features oder Werten, den in einer Operation angeforderten Typen zu melden, die in der Ergebnismenge vorliegen. Dieser Wert muss nicht notwendigerweise der Anzahl von Features oder Werten entsprechen, die im Antwortdokument tatsächlich präsentiert werden (siehe 7.7.4.3). Wenn ein Server nicht in der Lage ist, die Gesamtanzahl abgeglicherer Features oder Werte zu melden, muss er dies mit dem Wert „unknown“ (nicht bekannt) angeben.

Durch Setzen des Wertes des „resultType“-Parameters (siehe 7.6.3.6) auf „hits“ kann eine Zählung der Gesamtanzahl von Features oder Werten erhalten werden, die eine Operation zurückgeben würde, ohne dass die Kosten für die Übertragung der Ergebnismenge anfallen würden. Bei Servern, die Antwortpaging unterstützen (siehe 7.7.4.4) muss der Wert des „next“-Parameters so gesetzt werden, dass die erste Teilmenge der Antwort-Features oder -Werte geholt wird.

7.7.4.3 „numberReturned“-Parameter

Der Parameter „numberReturned“ muss verwendet werden um eine Zählung der Anzahl der in einem Antwortdokument präsentierten Features oder Werte anzuzeigen.

ANMERKUNG Siehe 7.6.3.4 zu den Zählregeln für Werte.

Ist für den „count“-Parameter ein Wert festgelegt (siehe 7.6.3.5), muss der Wert des Parameters „numberReturned“ gleich dem festgelegten Wert des Zählparameters oder kleiner als dieser sein.

Wenn der Wert des „count“-Parameters nicht explizit in der Operation festgelegt ist, darf ein Standardwert konfiguriert und im Fähigkeitendokument des Servers (siehe 8.3.3) angezeigt werden, der sich so auf „numberReturned“ auswirkt, als wäre er in der Operation festgelegt.

Ist der „count“-Parameter nicht explizit oder durch eine Serverkonfiguration festgelegt, muss der Wert des „numberReturned“-Antwortparameters gleich dem Wert des „numberMatched“-Antwortparameters sein.

7.7.4.4 Antwortpaging

7.7.4.4.1 Einleitung

Antwortpaging ist die Fähigkeit eines Clients, eine Menge von Antwort-Features oder -Werten zu durchblättern, wobei jeweils N Features oder Werte einzeln durchblättert werden, ähnlich wie beim seitenweisen Durchscrollen der Antwort einer Suchmaschine.

Server, die Antwortpaging unterstützen, müssen dies in ihrem Fähigkeitendokument mit der Einschränkung „ImplementsResultPaging“ (siehe Tabelle 13) angeben.

Antwortpaging wird erreicht durch Verwendung des auf den Antwortsammlungen definierten „previous“- und „next“-Parameters (siehe 10.3.1 und 11.3.1).

Damit das Paging ausgelöst wird, muss entweder der „count“-Parameter (siehe 7.6.3.5) auf der Abfrage gesetzt werden, oder der Server muss einen Standardwert für diesen Parameter implementieren, der im Fähigkeitendokument des Servers angezeigt werden muss (siehe Tabelle 14).

Die Werte des „previous“- oder „next“-Attributs müssen servergenerierte URI sein, die die entsprechende Ergebnismenge abrufen. Das spezifische Format dieser URI ist implementierungsabhängig, ebenso wie die Einzelheiten dazu, wie bzw. ob der Server die Ergebnisse einer Operation in den Cache kopiert, um sie dem Client in einzelnen Teilmengen präsentieren zu können.

Bei Auflösen der vorherigen oder der folgenden URI müssen Server entweder eine gültige Antwortsammlung generieren, die die folgende oder die vorherige Feature- oder Wertemenge enthält (oder Verknüpfungstapel im Falle einer Verknüpfung), oder eine Ausnahmenachricht, die angibt, dass die Ergebnismenge nicht mehr verfügbar ist (möglicherweise, weil der Client zu lange gewartet hat, bevor er der vorherigen oder folgenden Verknüpfung gefolgt ist und die Ergebnisse in der Zwischenzeit aus dem Cache des Servers gelöscht wurden). In diesem Fall muss der Server eine „ResponseCacheExpired“-Ausnahme generieren (siehe Tabelle 3).

Server müssen mit der „ResponseCacheTimeout“-Einschränkung in ihrem Fähigkeitendokument (siehe Tabelle 14) anzeigen, wie lange Ergebnismengen für ein Antwortpaging im Cache vorgehalten werden.

BEISPIEL Zur Veranschaulichung der Verwendung dieser Parameter sei folgende „GetFeature“-Anforderung (siehe Abschnitt 11) zu betrachten:

```
<GetFeature service="WFS" version="2.0.0" count="100"
xmlns="http://www.opengis.net/wfs/2.0"
xmlns:cw="http://www.someserver.com/cw"
xmlns:fes="http://www.opengis.net/ogc/1.1"
xmlns:gml="http://www.opengis.net/gml/3.2">
  <Query typeName="cw:MyFeatureType"/>
</GetFeature>
```

Die Interaktionsfolge mit dem Server gestaltet sich wie folgt:

- ein Client sendet die Anforderung an einen Server, der Paging unterstützt;
- der Server antwortet mit einem wfs:FeatureCollection-Element, das die ersten 100 Einträge in der Ergebnismenge enthält. Das Attribut *next* wird so gesetzt, dass der Client die nächsten 100 Features abrufen kann, das Attribut *previous* wird jedoch nicht gesetzt, da es sich um die erste Featuremenge in der Ergebnismenge handelt;
- der Client durchläuft den URI *next*;
- der Server antwortet mit einem weiteren wfs:FeatureCollection-Element, das die nächsten 100 Features in der Ergebnismenge enthält. In diesem Fall setzt der Server sowohl den Attributwert *previous* als auch *next*, sodass der Client die vorherigen 100 Features oder die nächsten 100 Features in der Ergebnismenge der ursprünglich in a) gestellten „GetFeature“-Anforderung abrufen kann;
- der Client durchläuft weiter jeden *next*-URI, bis eine <wfs:FeatureCollection>-Antwort ohne gesetztes *next*-Attribut empfangen wird. Dadurch wird angegeben, dass die letzte Feature-Menge, von der die ursprünglichen in a) gestellten Anforderung, abgerufen wurde;
- auf ähnliche Weise kann ein Client den *previous*-URI durchlaufen, bis eine <wfs:FeatureCollection>-Antwort empfangen wird, in der das *previous*-Attribut gesetzt ist, wodurch angegeben wird, dass die erste Menge von 100 Features abgerufen wurde.

7.7.4.4.2 Transaktionskonsistenz des Antwortpaging

7.7.4.4.2.1 Deklarieren der Transaktionskonsistenz

Server müssen mit der Einschränkung „PagingIsTransactionSafe“ (siehe Tabelle 14) in ihrem Fähigkeiten-dokument anzeigen, ob sie Transaktionskonsistenz für ein Antwortpaging unterstützen.

7.7.4.4.2.2 Antwortpaging mit Transaktionskonsistenz

Server, die eine Transaktionskonsistenz deklarieren, müssen diese zwischen Paging-Iterationen aufrechterhalten. Die Sicht der Daten, die ein Client während des Durchblätterns der Antwortmenge sieht, muss im Hinblick auf die Zeit, zu der die Ursprungsanforderung ausgeführt wurde, konsistent sein.

7.7.4.4.2.3 Antwortpaging ohne Transaktionskonsistenz

Server, die keine Transaktionskonsistenz deklarieren, müssen keine Transaktionskonsistenz bzw. keinen Transaktionszustand zwischen Paging-Iterationen aufrechterhalten. Dadurch ist es möglich, zwischen Iterationen neue Features hinzuzufügen, zu aktualisieren oder aus der vollständigen Ergebnismenge zu entfernen. Infolgedessen ist es möglich, zwischen Iterationen ein Ergebnismengenelement zu überspringen oder zu duplizieren.

7.8 Verwendung des Attributs „schemaLocation“

Jedes von einem WFS generierte GML-Dokument muss ein entsprechendes GML-Anwendungsschemadokument referenzieren, damit die Ausgabe validiert werden kann. Dazu ist das Attribut „schemaLocation“ zu verwenden, definiert in der XML-Schema-Spezifikation (siehe W3C XML Schema Part 1).

BEISPIEL Das folgende XML-Fragment zeigt die Verwendung des „schemaLocation“-Attributs auf dem Wurzelement, das den Ort eines XML-Schemadokuments anzeigt, das zur Validierung genutzt werden kann:

```
<?xml version="1.0" ?>
<wfs:FeatureCollection
  xmlns="http://www.someserver.com/myns"
  xmlns:myns="http://www.someserver.com/myns"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.someserver.com/myns
  http://www.someserver.com/wfs.cgi?request=DescribeFeatureType&typenames=TreesA_1M,RoadL_1M"> ...
```

In diesem Beispiel verwendet der Dienst eine „DescribeFeatureType“-Operation (siehe Abschnitt 9), um sich selbst wieder aufzurufen, um ein Schemadokument zu referenzieren, in dem die Elemente TreesA_1M und RoadL_1M deklariert sind.

7.9 Abfrageausdrücke

7.9.1 Einleitung

Ein Abfrageausdruck ist eine Aktion, die einen Server anweist, seinen Datenspeicher nach Ressourcen zu durchsuchen, die einige in der Abfrage kodierte Filterausdrücke erfüllen. Abfrageausdrücke suchen nach einem einzelnen Typ oder können zwei oder mehr Ressourcentypen verknüpfen und so nach Ressourcentupeln suchen.

In dieser Internationalen Norm sind die abfragbaren Ressourcen Features (siehe 7.1 und ISO 19143:2010, 6.3.3.1.1).

Ein Abfrageausdruck löst nach einer Menge von Features oder Feature-Tupeln auf, die den Filterausdruck erfüllen. Auf der Ergebnismenge wird wiederum mit einer in dieser Internationalen Norm definierten Operation operiert.

BEISPIEL Die Operation „GetFeature“ (siehe Abschnitt 11) verwendet einen Abfrageausdruck, um eine Teilmenge von Features zu identifizieren, um diese einem Client in einem Antwortdokument vorzulegen. Auf ähnliche Weise verwendet die Operation „LockFeature“ (siehe Abschnitt 12) einen Abfrageausdruck, um eine sperrende Feature-Teilmenge zu identifizieren.

Diese Internationale Norm definiert zwei Typen von Abfrageausdrücken: Ad-hoc-Abfrageausdrücke und gespeicherte Abfrageausdrücke.

Ad-hoc-Abfrageausdrücke werden mit Hilfe des Elementes wfs:Query in XML kodiert. Sie werden „ad hoc“ genannt, weil die Abfrage nicht vom Server gespeichert wird und nur während der Laufzeit bekannt ist.

Gespeicherte Abfrageausdrücke werden mit dem Element wfs:StoredQuery kodiert. Ein gespeicherter Abfrageausdruck ist eine Abfrage, die zuvor im Datenspeicher des Servers gespeichert wurde und jederzeit mit dem Identifikator der Abfrage ausgeführt werden kann.

Beide Abfrageausdruck-Typen leiten sich von den in ISO 19143:2010, Abschnitt 6, definierten abstrakten Abfrageausdrücken ab.

7.9.2 Ad-hoc-Abfrageausdruck

7.9.2.1 Abfragesemantik

Ein Ad-hoc-Abfrageausdruck kann in einer „GetPropertyValue“- (siehe Abschnitt 10), „GetFeature“- (siehe Abschnitt 11), „GetFeatureWithLock“- (siehe Abschnitt 13) oder „LockFeature“-Operation (siehe Abschnitt 12) verwendet werden, um die Menge von Features zu identifizieren, auf denen operiert werden soll.

Wie in Bild 8 gezeigt, enthält ein Ad-hoc-Abfrageausdruck einen „typeName“-Parameter, eine Projektionsklausel, eine Selektionsklausel und eine Sortierklausel.

Der obligatorische Parameter „typeName“ listet den Namen eines oder mehrerer abzufragender Feature-Typen auf.

Die optionale Projektionsklausel identifiziert eine Teilmenge optionaler Feature-Eigenschaften, die in der Ergebnismenge zu präsentieren sind. Die Projektionsklausel für XML-kodierte Ad-hoc-Abfrageausdrücke kann auch verwendet werden, um auf einer Eigenschaft-für-Eigenschaft-Basis zu steuern, wie Ressourcenverweise (siehe 7.6.4) im Antwortdokument aufgelöst werden.

ANMERKUNG Wie Ressourcenverweise in einem Antwortdokument aufgelöst werden, kann für KVP-kodierte Anforderungen nur auf der Operationsebene gesteuert werden. Erreicht wird dies durch Verwendung der Schlüsselwörter RESOLVE, RESOLVEDDEPTH und RESOLVETIMEOUT (siehe 7.6.4.3). Dies steht im Gegensatz zu XML-kodierten Anforderungen, bei denen die Ressourcenauflösung auf der Operations- und der Eigenschaftsebene gesteuert werden kann.

Die optionale Selektionsklausel legt Kriterien fest, die Features unter bestimmten Bedingungen aus dem Datenspeicher eines Servers auswählen.

Die optionale Selektionsklausel legt fest, wie die Features im Antwortdokument geordnet sein sollten.

Eine Ad-hoc-Abfrage ermöglicht außerdem die Bestätigung und Verwendung eines CRS bei Präsentation von Feature-Geometrien in einem Antwortdokument.

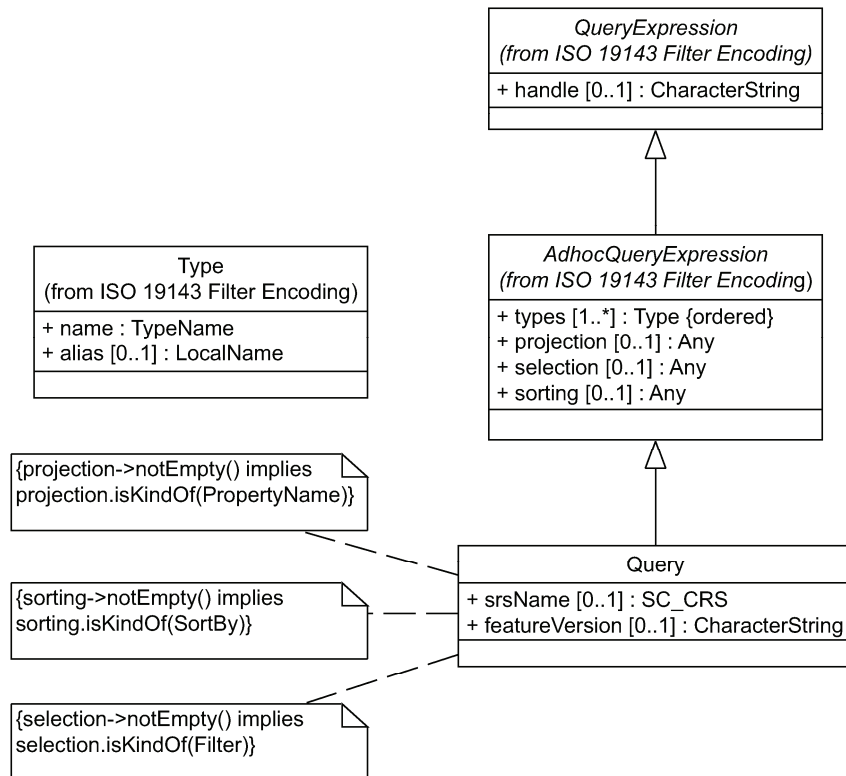


Bild 8 — Ad-hoc-Abfrageausdruck

7.9.2.2 XML-Kodierung

Das folgende XML-Schemafragment definiert die XML-Kodierung für einen Ad-hoc-Abfrageausdruck:

```

<xsd:element name="Query" type="wfs:QueryType"
  substitutionGroup="fes:AbstractAdhocQueryExpression"/>
<xsd:complexType name="QueryType">
  <xsd:complexContent>
    <xsd:extension base="fes:AbstractAdhocQueryExpressionType">
      <xsd:attribute name="srsName" type="xsd:anyURI"/>
      <xsd:attribute name="featureVersion" type="xsd:string"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
  
```

Das Element wfs:Query umfasst den vom Typ fes:AbstractAdhocQueryExpressionType geerbten „typeNames“-Parameter (siehe ISO 19143:2010, 6.3.2).

7.9.2.3 KVP-Kodierung

Tabelle 8 definiert die KVP-Kodierung für Ad-hoc-Abfrageausdrücke.

Tabelle 8 enthält die zur KVP-Kodierung eines Ad-hoc-Abfrageausdrucks erforderlichen Parameter aus 7.9.2.4 und ISO 19143:2010, Tabelle 2.

Tabelle 8 — Schlüsselwörter zur KVP-Kodierung von Ad-hoc-Abfragen

URL-Komponente	O/M ^a	Beschreibung
TYPENAMES	M ^b	siehe 7.9.2.4.1
ALIASES	O	siehe 7.9.2.4.3
SRSNAME	O	siehe 7.9.2.4.4
Projektionsklausel	O	siehe Tabelle 9
FILTER	O	siehe ISO 19143:2010, 6.3.3
FILTER_LANGUAGE	O	siehe ISO 19143:2010, 6.3.3
RESOURCEID	O	siehe ISO 19143:2010, 6.3.3
BBOX	O	siehe OGC 06-121r3
SORTBY	O	siehe ISO 19143:2010, Abschnitt 8 Der Parameter SORTBY wird verwendet, um eine Liste von Eigenschaften festzulegen, deren Werte verwendet werden sollten, um (bei Präsentation) die Menge von Feature-Instanzen zu ordnen, die die Abfrage erfüllen. Der Wert des SORTBY-Parameters muss die Form „PropertyName [ASC DESC][,PropertyName [AASC DESC],...]“ besitzen, wobei die Buchstaben ASC verwendet werden, um eine aufsteigende Sortierung anzugeben und die Buchstaben DESC, um eine absteigende Sortierung anzugeben. Sind weder ASC noch DESC festgelegt, muss die Standard-Sortierreihenfolge aufsteigend sein. Ein Beispielwert könnte „SORTBY=Field1 DESC,Field2 DESC,Field3“ sein. In diesem Fall werden die Ergebnisse nach Feld 1 absteigend sortiert, nach Feld 2 absteigend und nach Feld 3 aufsteigend.
^a O = Optional, M = obligatorisch (en: Mandatory) ^b Der Parameter TYPENAMES ist in allen Fällen obligatorisch, außer, wenn der Parameter RESOURCEID festgelegt ist (siehe 7.9.2.4.1).		

Wenn mehrere KVP-kodierte Abfrageausdrücke in einer WFS-Anforderung vorkommen, müssen die Parameterwerte für jeden Abfrageausdruck voneinander mit Klammern getrennt werden (siehe 6.2.5.3). Ist für einen Abfrageausdruck ein optionaler Parameter festgelegt, muss für diesen Parameter für alle in der Anforderung kodierten Abfrageausdrücke ein Wert festgelegt werden.

Aus einer Menge sich gegenseitig ausschließender Parameter darf nur ein Parameter in einer KVP-kodierten Anforderung, die mehrere Abfrageausdrücke kodiert, festgelegt werden; für den gewählten Parameter muss ein Wert für jeden kodierten Abfrageausdruck festgelegt werden.

BEISPIEL Wenn eine KVP-kodierte Anforderung mehrere Abfrageausdrücke enthält und einer dieser Abfrageausdrücke zur Kodierung des Prädikats das Schlüsselwort FILTER verwendet, dann müssen alle Abfrageausdrücke in der Anforderung das Schlüsselwort FILTER verwenden. Eine KVP-kodierte Anforderung mit mehreren Abfrageausdrücken, bei der eine Abfrage zur Kodierung des Prädikats das Schlüsselwort FILTER verwendet und ein anderer Abfrageausdruck RESOURCEID verwendet, ist zum Beispiel nicht möglich.

Bei Verwendung in einer Anforderung darf das Schlüsselwort BBOX nur eine einzelne Zeichenbox kodieren (siehe OGC06-121r3:2009, 10.2.3). Dieses Zeichenbox-Prädikat muss jedoch für alle in der Anforderung kodierten Abfrageausdrücke gelten (siehe B.8.5.4). Dies steht im Gegensatz zu den Schlüsselwörtern FILTER

und RESOURCEID, die mehrere Prädikate kodieren können, die jedem in einer KVP-kodierten Anforderung kodierten Abfrageausdruck entsprechen (siehe B.8.4.15).

Wie bei den XML-kodierten Anforderungen sind auch mehrere in einer einzelnen WFS-Operation kodierte KVP-kodierte Abfrageausdrücke als voneinander unabhängig zu betrachten.

7.9.2.4 Erläuterung der Parameter

7.9.2.4.1 „typeNames“-Parameter

Bei XML-kodierten Ad-hoc-Abfrageausdrücken muss der Parameter „typeNames“ mit dem „typeNames“-Attribut auf dem wfs:Query-Element kodiert werden. Dieses Attribut wird durch das abstrakte Element

`fes:AbstractAdhocQueryExpressionType`

vererbt (siehe ISO 19143, 6.3.2).

Bei KVP-kodierten Ad-hoc-Abfrageausdrücken muss der „typeNames“-Parameter mit dem TYPENAMES-Schlüsselwort kodiert werden (siehe Tabelle 8).

Der Parameter TYPENAMES ist in allen Fällen obligatorisch, es sei denn, der Parameter RESOURCEID ist festgelegt. In diesem Fall darf der Parameter TYPENAMES weggelassen werden, da jede Feature-Instanz durch allein durch ihren Ressourcen-Identifikator identifiziert werden kann (siehe 7.2). Sind sowohl der Parameter TYPENAMES als auch der Parameter RESOURCEID festgelegt, müssen alle Feature-Instanzen, die durch den Parameter RESOURCEID identifiziert sind, dem durch den Parameter TYPENAME festgelegten Typ entsprechen; andernfalls muss der Server eine „invalidParameterValue“-Ausnahme melden (siehe OGC 06-121r3:2009, Tabelle 25), wobei der Wert des „locator“-Attributs (siehe OGC-06-121r3:2009, 8.4) auf „RESOURCEID“ zu setzen ist.

Der Parameter „typeNames“ (siehe ISO 19143:2010, 6.3.3.1.1) muss in einem Ad-hoc-Abfrageausdruck verwendet werden, um die Namen eines oder mehrerer in Wechselbeziehung stehender abzufragender Feature-Typen zu kodieren. Einzelne Feature-Typ-Namen sind als QName zu kodieren (siehe W3C XML Schema Part 2).

Der Wert jedes als ein Wert des „typeNames“-Parameters aufgelisteten QName muss einem der im Fähigkeitendokument des Servers (siehe 8.3.3) angezeigten Feature-Typen entsprechen.

7.9.2.4.2 „schema-element()“-Funktion

Wenn die Liste der Werte für die „typeNames“-Parameter einen einzelnen Namen enthält, kann die „schema-element()“-Funktion verwendet werden, um eine Folge von Abfragen auf dem festgelegten Feature-Typ und jedem Feature-Typ, dessen Objektelemente sich in der Substitutionsgruppe des festgelegten Feature-Typs befinden, auszulösen.

BEISPIEL `typeNames="schema-element(ns1:Vehicle)"` könnte zusammen mit `ns1:Vehicle` die Feature-Typen `ns1:Cars`, `ns1:Boats` usw. abfragen.

7.9.2.4.3 „aliases“-Parameter

Bei XML-kodierten Ad-hoc-Abfrageausdrücken muss der Parameter „aliases“ mit dem „aliases“-Attribut auf dem wfs:Query-Element kodiert werden. Das „aliases“-Attribut wird durch das abstrakte Element

`fes:AbstractAdhocQueryExpressionType`

vererbt (siehe ISO 19143, 6.3.2).

Bei KVP-kodierten Ad-hoc-Abfrageausdrücken muss der „aliases“-Parameter mit dem ALIASES-Schlüsselwort kodiert werden.

Der optionale Parameter „aliases“ kann in einem Abfrageausdruck verwendet werden, um eine Liste alternativer Namen für die Namen der Feature-Typen festzulegen, festgelegt als der Wert des „typeNames“-Parameters. Ein Feature-Typ-Alias darf überall dort verwendet werden, wo der Feature-Typ-Name im Kontext eines Abfrageausdrucks verwendet werden darf.

Die Anzahl von Listenelementen im Wert des „aliases“-Parameters muss mit der Anzahl entsprechender Feature-Typ-Namen im Wert des „typeNames“-Parameters übereinstimmen und muss 1:1 korreliert sein.

BEISPIEL 1

TYPENAMES=(ns1:FeatureType1,ns2:FeatureType2)(ns2:FeatureType2,ns2:FeatureType3)&ALIASES=(A,B)(C,D)

Dieses KVP-kodierte Beispiel kodiert zwei Ad-hoc-Abfrageausdrücke, die jeweils eine Verknüpfung vornehmen. Der erste Ausdruck verknüpft die Feature-Typen ns1:FeatureType1 und ns2:FeatureType2 ab, die in einer Alias-Beziehung zu A und B stehen. Der zweite Ausdruck verknüpft die Feature-Typen ns2:FeatureType2 und ns2:FeatureType3, die in einer Alias-Beziehung zu C und D stehen.

Jedes im Wert des „aliases“-Parameters festgelegte Alias muss im Kontext eines einzelnen Abfrageausdrucks eindeutig sein.

Wird der Parameter „aliases“ verwendet, muss ein Alias für jeden Feature-Typ-Namen festgelegt werden, der als Wert des „typeNames“-Parameters aufgelistet ist.

Alias-Beziehungen werden üblicherweise in Abfrageausdrücken verwendet, die eine Verknüpfungsoperation ausführen (siehe 7.9.2.5.3.1), um Selbstverknüpfungen zu unterstützen. Dabei handelt es sich um eine Verknüpfung eines Feature-Typs mit sich selbst.

BEISPIEL 2 typeName="myns:Feat1 myns:Feat1" aliases="a b"

Dieses XML-kodierte Beispiel zeigt die Kodierung des Parameters „typeName“. Der erste Feature-Typ, myns:Feat1 mit dem Namen „a“ in Beziehung gesetzt, und der zweite Feature-Typ, myns:Feat1, mit dem Namen „b“. Somit können Eigenschaften der ersten Instanz von myns:Feat1 in einer Anforderung als „/a/property_name“ referenziert werden, und Eigenschaften der zweiten Instanz von myns:Feat1 können in einer Abfrage als „/b/property_name“ referenziert werden, wobei das Token „property_name“ als Platzhalter für den Namen einer beliebigen Eigenschaft des Features myns:Feat1 verwendet wird.

7.9.2.4.4 „srsName“-Parameter

Das optionale Attribut „srsName“ kann verwendet werden, um eine bestimmte WFS-unterstützte CRS-Transformation zu bestätigen, die auf die Geometrien der in einem Antwortdokument zurückgegebenen Features anzuwenden ist.

Der Wert des Parameters „srsName“ kann der wfs:DefaultCRS-Wert sein oder jeder der für den Feature-Typ im Fähigkeitendokument des Servers (siehe 8.3.3) aufgelisteten wfs:OtherCRS-Werte. Wird kein srsName-Wert geliefert, müssen die Feature-Geometrien im Antwortdokument mit dem angezeigten wfs:DefaultCRS-Wert kodiert werden.

Dieses Attribut hat für Feature-Typen ohne raumbezogene Eigenschaften keine Bedeutung und ist zu ignorieren.

Server, die mehr als einen wfs:OtherCRS-Wert in ihrem Fähigkeitendokument anzeigen (siehe 8.3.3), müssen in der Lage sein, zwischen dem zum Speichern von Features verwendeten und jedem angeforderten CRS mit dem Attribut „srsName“ zu transformieren.

Server, die diese Internationale Norm implementieren, müssen in der Lage sein, „srsName“-Attributwerte mit folgendem Formatmodell zu verarbeiten:

urn:ogc:def:objectType:authority:version:<EPSG code> (siehe OGC 07-092r3)