



Erhöhe den Nutzen deines Dienstes

Qualitätskontrolle für OGC-konforme
Geodatendienste mit TEAM Engine

von Dirk Stenger (stenger@lat-lon.de)

Agenda

1. Einleitung
2. TEAM Engine
3. OGC Testsuites für die TEAM Engine
4. WFS 2.0-Referenzimplementierung in deegree
5. Docker nutzen: deegree WFS 2.0 mit der TEAM Engine testen
6. Fazit: So kann der Nutzen deines Dienstes erhöht werden

1. Einleitung

- **Dirk Stenger**, Diplom Geograph
- Seit 2012 bei lat/lon als Software-Entwickler und Experte für Geodateninfrastrukturen eingestellt.
- Interessen und Hauptarbeitsfelder sind die Implementierung von OGC Standards wie z.B. WMS, WFS, CSW, GML und WCS im Rahmen der deegree Initiative, welche Umsetzungen von OGC Standards auf Basis einer Open Source Lizenz (LGPL) anbietet.
- Mitglied des OGC CITE Teams.
 - Technische Leitung der Testsuites für WMS, WFS und WCS.

1. Einleitung

cite.openeospatial.org/teamengine/



OGC Validator



ogctestbed12.lat-lon.de/deegree/services/wfs?service

Mit dieser XML-Datei sind anscheinend keine Style-In

```
-<WFS_Capabilities version="2.0.0" xsi:schemaL
-<ows:ServiceIdentification>
  -<ows:Title>
    OGC Testbest 12 Candidate WFS 2.0 Referenc
  </ows:Title>
  -<ows:Abstract>
    This service is the Candidate WFS 2.0 Referen
  </ows:Abstract>
  <ows:ServiceType codeSpace="http://www.og
  <ows:ServiceTypeVersion>2.0.0</ows:Servic
  </ows:ServiceIdentification>
-<ows:ServiceProvider>
  <ows:ProviderName>lat/lon GmbH</ows:Pro
  <ows:ProviderSite xlink:href="http://www.lat
  -<ows:ServiceContact>
    <ows:IndividualName>Dirk Stenger</ows:
    <ows:PositionName>Software Engineer</o
  -<ows:ContactInfo>
    -<ows:Phone>
      <ows:Voice>0228/18496-0</ows:Voice>
      <ows:Facsimile>0228/18496-20</ows:F
```

Product Improvement and Differentiation

The [OGC](#) validator is an essential tool that helps organizations better implement service interfaces, encodings and clients that adhere to OGC standards. Passing the test and [getting OGC certified](#) helps organizations distinguishing their product in the market place.

“Achieving OGC certification is extremely important to us as an organization - we are proud to have more than 20 products compliant with OGC standards.”

Stan Tilman | Intergraph

Available Test Suites

OGC

Specification	Version	Test Suite Revision	Status
Catalogue Service - Web (CSW)	2.0.2	1.16	Final
Catalogue Service - Web (CSW)	3.0.0	1.0	Final
GeoPackage	1.0	1.0	Final
Geography Markup Language (GML)	3.2.1	1.25	Final

Community Tool

Developers, product and quality assurance managers have been using this free validator for over 8 years.



The validator can be used by OGC and non OGC members as often as they like to test their implementations of OGC standards.

The source of the engine and the tests are available at [GitHub](#). The [CITE forum](#) provides a place to ask questions and help developers pass the tests.

Features

The OGC Web Validator has the following features:

- Speed testing
- Detailed reporting
- Storing of sessions
- Validation of services
- Validation of clients
- Validation of schemas
- Validation of data

Get OGC Certified



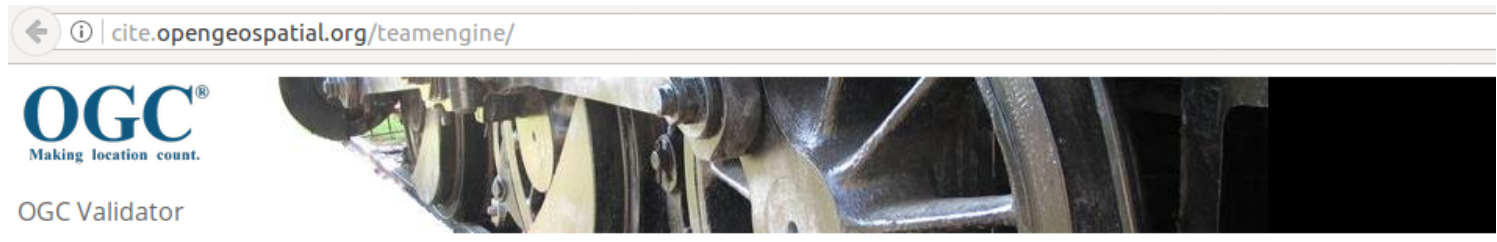
To apply for certification, visit the [OGC implementation database](#), register your product and provide details about the validation results.

2. TEAM Engine

- TEAM Engine (Test, Evaluation, And Measurement Engine) ist eine Testausführungs-Engine, mit welcher Webservices und andere Ressourcen getestet werden können.
- Ist in Java geschrieben.
- <https://github.com/opengeospatial/teamengine>

2. TEAM Engine

- Führt Testskripte aus, die in Compliance Test Language (CTL), TestNG und anderen Sprachen geschrieben sind.
- Kann als Webanwendung oder Kommandozeilentool verwendet werden.



Product Improvement and Differentiation

The [OGC](#) validator is an essential tool that helps organizations better implement service interfaces, encodings and clients that adhere to OGC standards. Passing the test and [getting OGC certified](#) helps

Community Tool

Developers, product and quality assurance managers have been using this free validator for over 8 years.



Features

The OGC Web Validator has the following features:

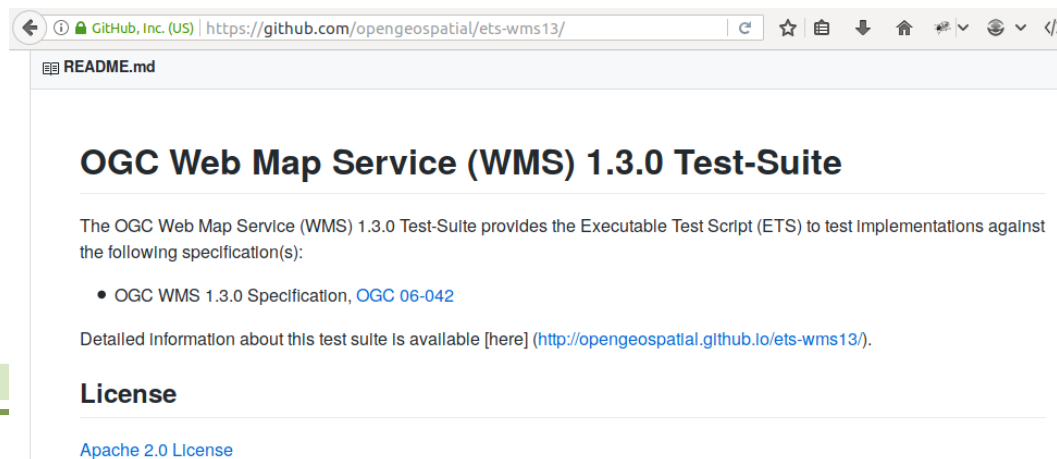
- Speed testing
- Detailed reporting
- Storing of sessions
- Validation of services
- Validation of clients
- Validation of schemas

3. OGC Testsuites für die TEAM Engine

- Das OGC stellt über 20 Testsuites bereit.
- Code der Testsuites befindet sich auf Github.
 - <https://github.com/opengeospatial/>
- Genutzte Sprachen:
 - CTL
 - TestNG

3. OGC Testsuites für die TEAM Engine

- Beispiele für Testsuites:
 - ETS-WFS11
 - <https://github.com/opengeospatial/ets-wfs11>
 - ETS-WFS20
 - <https://github.com/opengeospatial/ets-wfs20/>
 - ETS-WMS13
 - <https://github.com/opengeospatial/ets-wms13/>



The screenshot shows a web browser window displaying the GitHub repository page for 'OGC Web Map Service (WMS) 1.3.0 Test-Suite'. The browser's address bar shows the URL <https://github.com/opengeospatial/ets-wms13/>. The page content includes the title 'OGC Web Map Service (WMS) 1.3.0 Test-Suite', a description of the test suite, and a list of specifications it tests against. The license is identified as Apache 2.0 License.

OGC Web Map Service (WMS) 1.3.0 Test-Suite

The OGC Web Map Service (WMS) 1.3.0 Test-Suite provides the Executable Test Script (ETS) to test Implementations against the following specification(s):

- OGC WMS 1.3.0 Specification, [OGC 06-042](http://www.ogc.org/standards/wms)

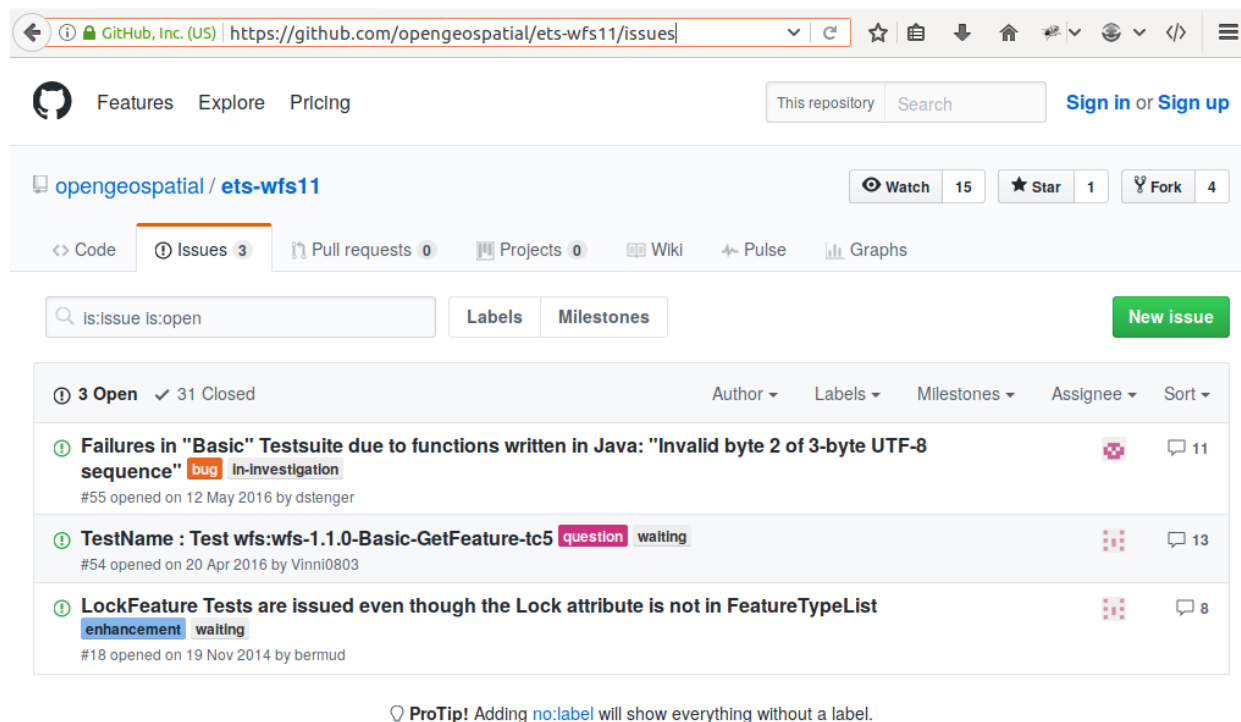
Detailed Information about this test suite is available [here] (<http://opengeospatial.github.io/ets-wms13/>).

License

[Apache 2.0 License](#)

3. OGC Testsuites für die TEAM Engine

- Issue Tracker befinden sich auf Github:
 - z.B. <https://github.com/opengeospatial/ets-wfs11/issues>



The screenshot shows the GitHub repository page for `opengeospatial/ets-wfs11`. The browser address bar displays the URL `https://github.com/opengeospatial/ets-wfs11/issues`. The repository page includes navigation links for Features, Explore, and Pricing, along with a search bar and options to sign in or sign up. The repository name `opengeospatial/ets-wfs11` is shown, along with statistics: 15 Watchers, 1 Star, and 4 Forks. The Issues tab is selected, showing 3 open issues. A search bar contains the text `is:issue is:open`. The issues list includes:

- 3 Open** (31 Closed) | Author | Labels | Milestones | Assignee | Sort
- Failures in "Basic" Testsuite due to functions written in Java: "Invalid byte 2 of 3-byte UTF-8 sequence"** (bug, in-investigation) | #55 opened on 12 May 2016 by dstenger | 11 comments
- TestName : Test wfs:wfs-1.1.0-Basic-GetFeature-tc5** (question, waiting) | #54 opened on 20 Apr 2016 by Vinni0803 | 13 comments
- LockFeature Tests are issued even though the Lock attribute is not in FeatureTypeList** (enhancement, waiting) | #18 opened on 19 Nov 2014 by bermud | 8 comments

ProTip! Adding `no:label` will show everything without a label.

3. OGC Testsuites für die TEAM Engine

OGC
Making location count.

OGC Validator

Select a test suite:

Organization: OGC

Specification: Web Map Service (WMS) - 1.3.0 [1 ▼]

Description (Optional):

Start a new test session

TEAM Engine v4

Web Map Service 1.3.0

Test Overview and Reference Implementations

Complete information about this test suite can be found [here](#).
Reference implementations can be found [here](#).

Capabilities Setup

Enter a capabilities document URL below. The URL may point to a static capabilities document or a GetCapabilities request from a WMS. A typical GetCapabilities request will look as follows:

`http://hostname/path?SERVICE=WMS&REQUEST=GetCapabilities&VERSION=1.3.0`

`http://cite.deegree.org/deegree-webservices-3.4-RC1/services/wms130?service=WMS&request=GetCapabilities`

UpdateSequence Values

The WMS specification allows servers to use an UpdateSequence value for maintaining cache consistency as described in [Section 7.2.3.5 of the specification](#). If the server advertises an UpdateSequence value and the Automatic option is selected below, the test suite will attempt to test the UpdateSequence behavior automatically. However, the lexical ordering of UpdateSequence values is determined by the server, so the tests may not always be correct. If you suspect a problem, select the Manual option and enter the updateSequence values requested below.

Automatic - The updateSequence tests will use automatically generated updateSequence values
 Manual - The updateSequence tests will use the values supplied below

(Fill in these boxes if the Manual option is selected above)

<input type="text"/>	A value that is lexically higher than the current updateSequence value
<input type="text"/>	A value that is lexically lower than the current updateSequence value

TEAM Engine 4.10

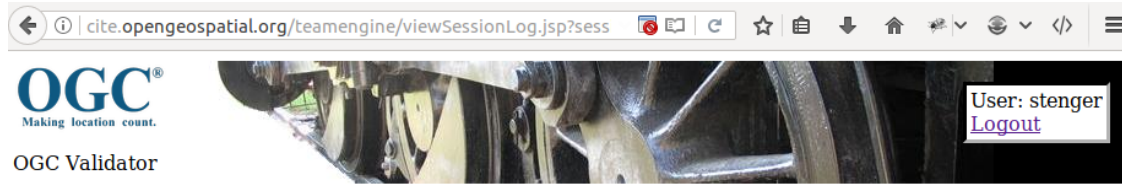
If you have any questions or suggestions contact O



lat/lon

www.lat-lon.de

3. OGC Testsuites für die TEAM Engine

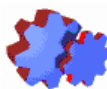


Results for session s0151

Test Suite: WMS 1.3 Conformance Test Suite

- [Test main:main \(View Details\)](#): Passed
 - [Test interactive:main \(View Details\)](#): Passed
 - [Test interactive:basic-polygons-sanity-check \(View Details\)](#): Passed
 - [Test interactive:blue-lake-sanity-check \(View Details\)](#): Passed
 - [Test interactive:layer-order \(View Details\)](#): Passed
 - [Test interactive:aspect-ratio \(View Details\)](#): Passed
 - [Test interactive:exceptions-inimage \(View Details\)](#): Passed
 - [Test interactive:fees-and-access-constraints \(View Details\)](#): Passed
 - [Test main:options-requirements \(View Details\)](#): Passed
 - [Test main:gif-or-png \(View Details\)](#): Passed
 - [Test main:std-data-present \(View Details\)](#): Passed
 - [Test main:getfeatureinfo-supported \(View Details\)](#): Passed
 - [Test main:std-data-queryable \(View Details\)](#): Passed
 - [Test basic_elements:main \(View Details\)](#): Passed
 - [Test basic_elements:version-negotiation \(View Details\)](#): Passed
 - [Test basic_elements:negotiate-no-version \(View Details\)](#): Passed
 - [Test basic_elements:negotiate-basic_elements-version \(View Details\)](#): Passed
 - [Test basic_elements:negotiate-higher-version \(View Details\)](#): Passed
 - [Test basic_elements:negotiate-lower-version \(View Details\)](#): Passed
 - [Test basic_elements:reserved-chars \(View Details\)](#): Passed
 - [Test basic_elements:escaped-chars \(View Details\)](#): Passed
 - [Test basic_elements:escaped-space \(View Details\)](#): Passed
 - [Test basic_elements:param-rules \(View Details\)](#): Passed
 - [Test basic_elements:extra-GetCapabilities-param \(View Details\)](#): Passed
 - [Test basic_elements:extra-GetMap-param \(View Details\)](#): Passed
 - [Test basic_elements:extra-GetFeatureInfo-param \(View Details\)](#): Passed
 - [Test getcapabilities:main \(View Details\)](#): Passed
 - [Test getcapabilities:requests \(View Details\)](#): Passed
 - [Test getcapabilities:each-format \(View Details\)](#): Passed
 - [Test getcapabilities:no-format \(View Details\)](#): Passed
 - [Test getcapabilities:invalid-format \(View Details\)](#): Passed
 - [Test getcapabilities:updatesequence-ignored \(View Details\)](#): Passed
 - [Test getcapabilities:updatesequence-current \(View Details\)](#): Passed
 - [Test getcapabilities:updatesequence-lower \(View Details\)](#): Passed
 - [Test getcapabilities:updatesequence-higher \(View Details\)](#): Passed
 - [Test getcapabilities:xml-validation \(View Details\)](#): Passed

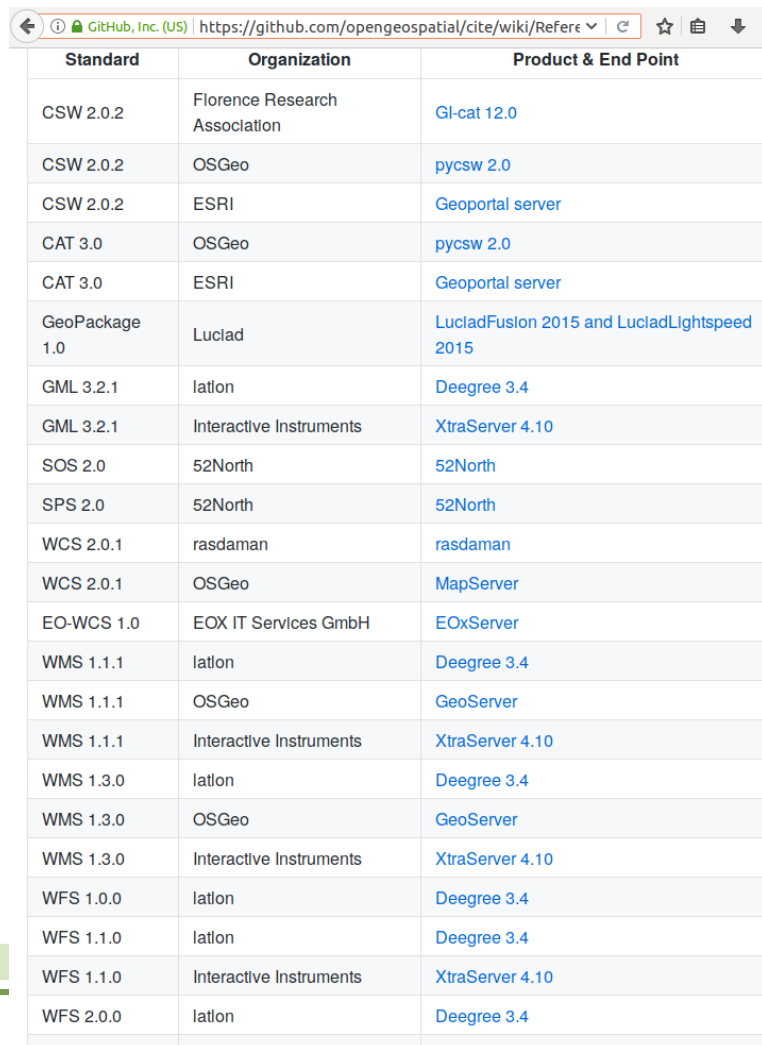



Executing tests...

4. WFS 2.0-Referenzimplementierung in deegree

- <https://github.com/opengeospatial/cite/wiki/Reference-Implementations>

- deegree
 - GML 3.2.1
 - WMS 1.1.1
 - WMS 1.3.0
 - WFS 1.0.0
 - WFS 1.1.0
 - WFS 2.0.0



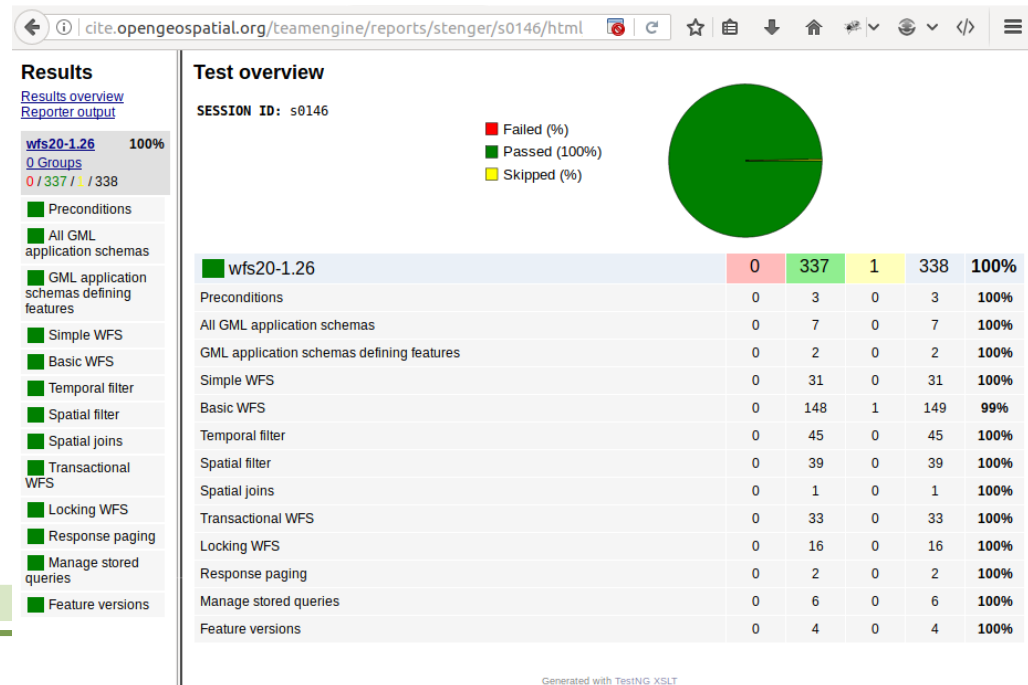
The screenshot shows a web browser window with the URL <https://github.com/opengeospatial/cite/wiki/Reference-Implementations>. The browser displays a table with three columns: Standard, Organization, and Product & End Point. The table lists various WFS 2.0 implementations from different organizations.

Standard	Organization	Product & End Point
CSW 2.0.2	Florence Research Association	GI-cat 12.0
CSW 2.0.2	OSGeo	pycsw 2.0
CSW 2.0.2	ESRI	Geoportal server
CAT 3.0	OSGeo	pycsw 2.0
CAT 3.0	ESRI	Geoportal server
GeoPackage 1.0	Luclid	LuclidFusion 2015 and LuclidLightspeed 2015
GML 3.2.1	latlon	Deegree 3.4
GML 3.2.1	Interactive Instruments	XtraServer 4.10
SOS 2.0	52North	52North
SPS 2.0	52North	52North
WCS 2.0.1	rasdaman	rasdaman
WCS 2.0.1	OSGeo	MapServer
EO-WCS 1.0	EOX IT Services GmbH	EOxServer
WMS 1.1.1	latlon	Deegree 3.4
WMS 1.1.1	OSGeo	GeoServer
WMS 1.1.1	Interactive Instruments	XtraServer 4.10
WMS 1.3.0	latlon	Deegree 3.4
WMS 1.3.0	OSGeo	GeoServer
WMS 1.3.0	Interactive Instruments	XtraServer 4.10
WFS 1.0.0	latlon	Deegree 3.4
WFS 1.1.0	latlon	Deegree 3.4
WFS 1.1.0	Interactive Instruments	XtraServer 4.10
WFS 2.0.0	latlon	Deegree 3.4

4. WFS 2.0-Referenzimplementierung in deegree

- Ergebnis des COM Threads aus OGC Testbed-12:
 - deegree WFS 2.0: `http://ogctestbed12.lat-lon.de/deegree/services/wfs?service=WFS&request=GetCapabilities`
- Konform zu folgenden „Conformance Classes“:

- Transactional WFS
- Locking WFS
- Response Paging
- Standard Joins
- Spatial Joins
- Temporal Joins
- Feature Versions
- Manage Stored Queries



4. WFS 2.0-Referenzimplementierung in deegree

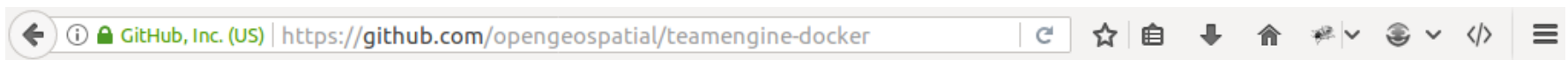
- Während der Entwicklung der RI wurde die Testsuite kontinuierlich genutzt, um die „Compliance“ sicherzustellen.
- Start der Tests über die Weboberfläche.
- Möglichkeit besteht, Tests in nächtlichen Builds in einer CI-Umgebung auszuführen.
 - Ausführung über das CLI der TEAM Engine.
 - Integration der Testausführung in Maven.

4. WFS 2.0-Referenzimplementierung in deegree

- Problem aus der Praxis:
 - Wenn externe TEAM Engine Installation genutzt wird, muss der getestete Dienst auch von extern erreichbar sein.
 - Die TEAM Engine nutzt die DCP URL aus den Capabilities, um Requests abzuschicken.
 - Diese URL muss also vom Server, auf dem die TEAM Engine läuft, erreichbar sein.
- Lösung für nur intern erreichbare Dienste:
 - TEAM Engine plus gewünschte ETS selber aufsetzen.

5. Docker nutzen: deegree WFS 2.0 mit der TEAM Engine testen

- Installationsanleitung im Web vorhanden:
 - <http://opengeospatial.github.io/teamengine/installation.html>
- Einfache Installation mit Docker möglich.
- Eigenes Projekt für TEAM Engine + Docker vorhanden:
 - <https://github.com/opengeospatial/teamengine-docker>



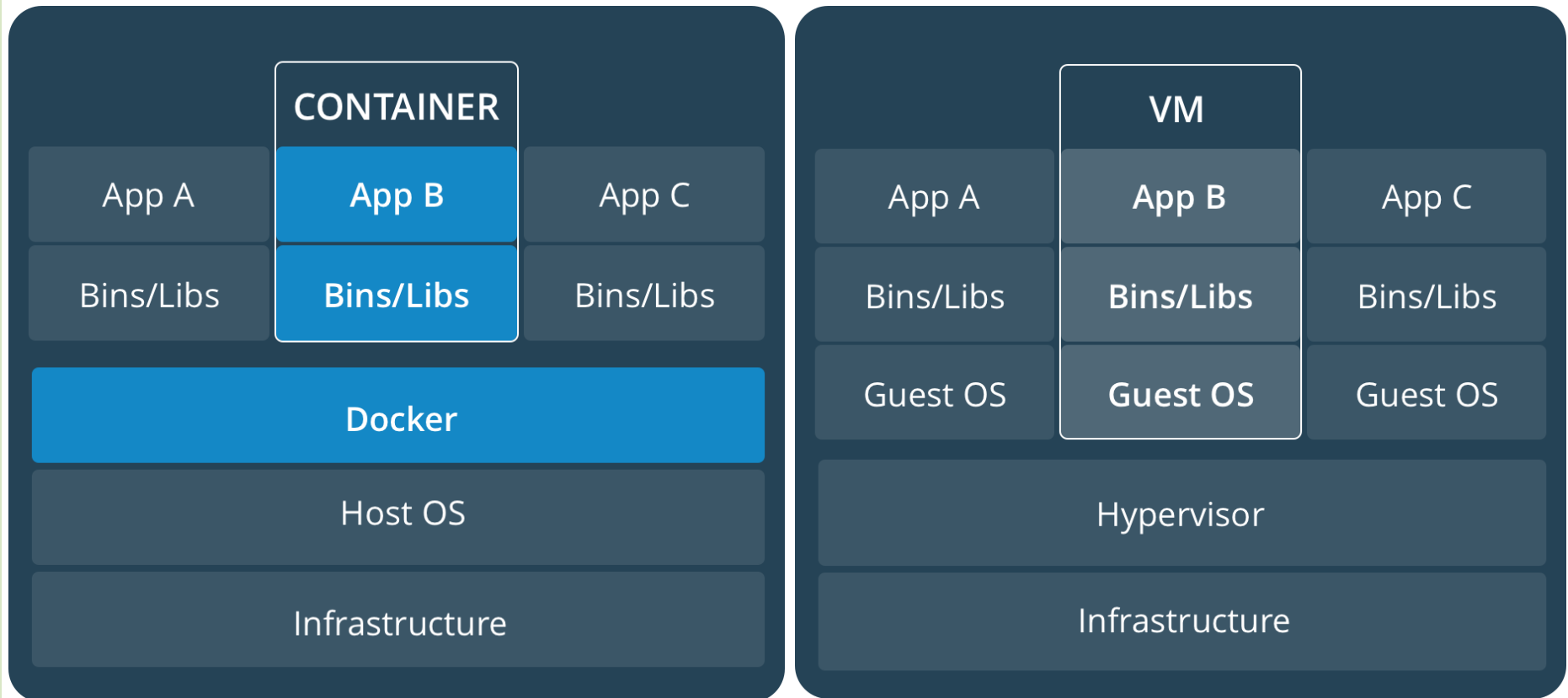
5. Docker nutzen: deegree WFS 2.0 mit der TEAM Engine testen

- Was ist Docker?
 - Mit Docker kann eine Anwendung in eine standardisierte Einheit gepackt werden: Den Docker Container.
 - Ein Docker Container beinhaltet die Anwendung und ein Dateisystem, welches alle benötigten Ressourcen beinhaltet.
 - Ein Docker Container kann auf Linux, Windows, macOS und den meisten Cloud Infrastrukturen ausgeführt werden.

5. Docker nutzen: deegree WFS 2.0 mit der TEAM Engine testen

Container-

vs Betriebssystem-Virtualisierung



Quelle: <https://www.docker.com/what-container>

5. Docker nutzen: deegree WFS 2.0 mit der TEAM Engine testen

- Installationsanleitung auf Github Projekt Startseite (<https://github.com/opengeospatial/teamengine-docker>).
- Projekt baut ein Docker Image.
- Aus diesem Image kann ein Docker Container erstellt werden.
- Einzige nötige Befehle:
 - `git clone https://github.com/opengeospatial/teamengine-docker.git`
 - `mvn clean package docker:build`
 - `docker run -p 8088:8080 --name teamengine --rm opengis/teamengine`

5. Docker nutzen: deegree WFS 2.0 mit der TEAM Engine testen

- Einschränkungen:
 - Dependencies zu den Testsuites sind noch nicht zentral verfügbar. Somit müssen diese lokal gebaut werden (mvn clean install).
 - Wird in einem Issue diskutiert:
<https://github.com/opengeospatial/teamengine-docker/issues/2>
 - Momentan nur Testsuite für WFS 2.0 verfügbar.
- Geplante Aktivitäten für die kommenden Monate:
 - Dependencies zentral bereistellen (Central Maven Repository).
 - Weitere Testsuites in das Docker Projekt einbinden.

5. Docker nutzen: deegree WFS 2.0 mit der TEAM Engine testen

- (deegree) WFS 2.0 kann lokal aufgesetzt und mit der soeben mit Docker gestarteten TEAM Engine getestet werden.
- Auf Docker Hub existieren fertige Docker Images:
 - <https://hub.docker.com/r/tfr42/deegree/>
 - Beinhaltet deegree Docker Image.
- Dazugehöriges Github Repository:
 - <https://github.com/tfr42/deegree-docker>
 - Beinhaltet deegree Dockerfile.
 - Integration mit PostgreSQL Datenbank (auch über Docker-Compose YAML-Konfiguration).

5. Docker nutzen: deegree WFS 2.0 mit der TEAM Engine testen

- Start mit nur einem Befehl möglich (deegree ohne Datenbank):
 - `docker run -p 8080:8080 --name deegree --rm tfr42/deegree`
- Start von deegree plus PostgreSQL Datenbank:
 - `docker run --name postgis -p 5432:5432 -d mdillon/postgis`
 - `docker run --name deegree -p 8080:8080 --link postgis:db -d deegree/deegree`
 - Alternativ kann Docker-Compose verwendet werden (nur ein Befehl).

6. Fazit: So kann der Nutzen deines Dienstes erhöht werden

- Die TEAM Engine plus die dazugehörigen Testsuites ermöglichen schnelles Testen auf OGC Compliance.
- Bringt sowohl für Konfigurations- als auch Implementierungsarbeiten Gewinne.
- Bestehende Dienste können OGC Compliant „gemacht“ werden (Konfiguration, Bugfixes etc.).
- Testsuites können als Regressionstests verwendet werden.

6. Fazit: So kann der Nutzen deines Dienstes erhöht werden

- So wird der Nutzen deines Dienstes erhöht:
 - Hohe Qualität der Dienste wird gewährleistet.
 - Dienste können garantiert interoperabel genutzt werden.
 - Funktionsumfang eines Dienstes kann klar definiert werden (z.B. unterstützt Response Paging).

Fragen und Antworten

Vielen Dank für Eure Aufmerksamkeit!

Contact & Licence



© 2017 **lat/lon**

gesellschaft für raumbezogene
informationssysteme mbH

Aennchenstrasse 19

53177 Bonn

Tel: +49 +228 18496-0

Fax: +49 +228 18496-29

info@lat-lon.de

<http://www.lat-lon.de>

Twitter: [@latlon_de](https://twitter.com/latlon_de)