Workshop WS13

Running INSPIRE Download Services on Docker with deegree and PostgreSQL

Schedule
Date: Monday, 2016-08-22
Time: 14:00 - 18:00 CEST

Location
Gustav-Stresemann-Institut e.V. / Europäische Tagungs- und Bildungsstätte
Room S 18, 1st floor
Langer Grabenweg 68
53175 Bonn

Instructors

Dirk Stenger (lat/lon GmbH) - stenger@lat-lon.de - +49 228 18496-0
Torsten Friebe (deegree OSGeo project) - friebe@lat-lon.de - +49 228 18496-0

Agenda

1. Setup the Docker infrastructure
2. Configure INSPIRE Direct Access Download Services based on deegree WFS 2.0
3. Import test data using deegree WFS-T interface
4. Retrieve data with different clients
5. Validate service and data

Online Document

https://goo.gl/dnW1hi

OSGeo-Live System Login
Username: `user`, Password: `user`

Part 1 - 

Install docker

https://docs.docker.com/engine/installation/linux/ubuntulinux/

```
sudo apt-get install docker-engine
```

Start docker daemon

```
sudo service docker start
```

Verify that docker is installed correctly

```
sudo docker run hello-world
```

Attention:
On LINUX the `docker` daemon binds on a UNIX socket which is owned by the user `root` and other users can access it with `sudo`. For this reason, `docker` daemon always runs as the `root` user.

Basic docker commands

General structure of the docker CLI:
`docker <command> [options] [arguments]`

Display help per docker command:
`docker <command> --help` - Show help per docker command

Commands and options used within this tutorial:

docker info - Display system-wide information
docker images - List images
docker pull - Pull an image or a repository from a registry (e.g. hub.docker.com)
docker ps - List containers
docker ps - List all containers
- -a - Show all containers, incl. **stopped** containers

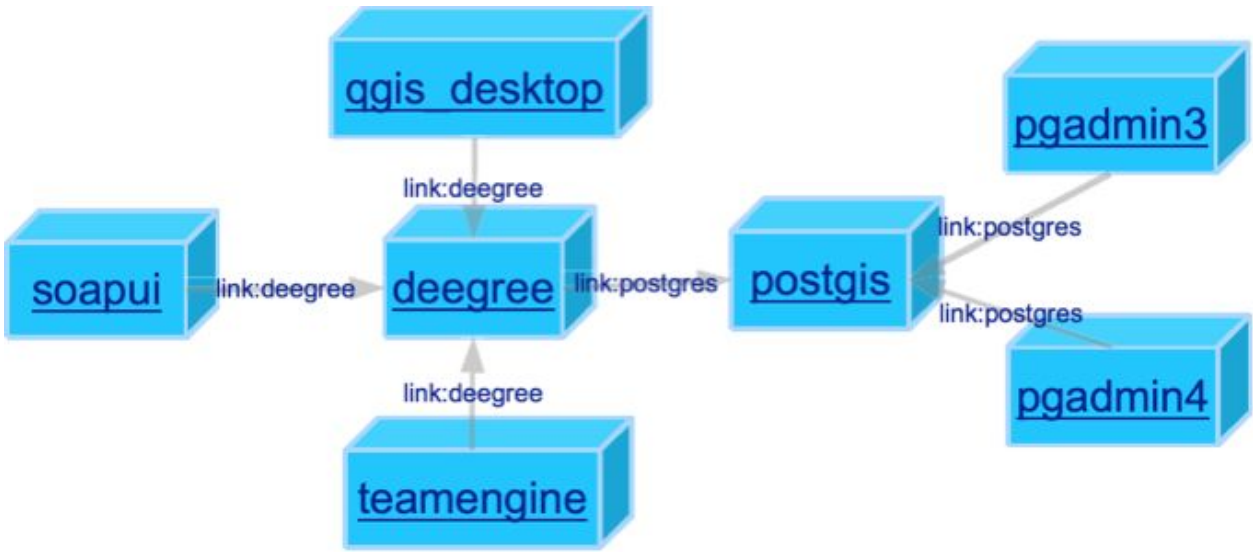
docker network ls - List all networks
docker run - Run a command in a **new** container
- -d, --detach Run container in background and print container ID
- -e, --env value Set environment variables (default [])

- -i, --interactive Keep STDIN open even if not attached
- --link value Add link to another container (default [] /)
- -m, --memory string Memory limit (format: <number><unit>, where unit = b, k, m or g)
- --name string Assign a name to the container
- --network string Connect a container to a network (default "default" / [host, bridge]
- -p, --publish value Publish a container's port(s) to the host (default [] / host:container)
- --rm Automatically remove the container when it exits
- -t, --tty Allocate a pseudo-TTY
- -v, --volume value Bind mount a volume (default [] / host_dir:container_dir)

docker exec - Run a command in a **running** container
docker logs - Fetch the logs of a container
- -f, --follow - Follow log output

docker start - Start one or more stopped containers
docker stop - Stop one or more running containers
docker rm - Remove one or more containers
docker rmi - Remove one or more images

Get docker images and run docker infrastructure



Spatial Database

PostgreSQL PostGIS

Docker Hub: https://hub.docker.com/r/mdillon/postgis/

```
docker pull mdillon/postgis
docker run -d --name postgis -p 5432:5432 mdillon/postgis
```

Docker Hub: https://hub.docker.com/r/zfil/pgadmin3/

```
docker pull zfil/pgadmin3
xhost +
docker run -d -t -v /tmp/.X11-unix:/tmp/.X11-unix -v
~/.pgadmin:/home/pgadmin -e DISPLAY=unix:0
--name pgadmin3 --link postgis:postgres zfil/pgadmin3
```

Docker Hub: https://hub.docker.com/r/fenglc/pgadmin4/

```
docker pull fenglc/pgadmin4
docker run -d --name pgadmin4 -p 5050:5050 --link postgis:postgres
fenglc/pgadmin4
```

Open in browser: http://localhost:5050/browser/

Connection parameters for DBA

Hostname:	postgres
Port:	5432
User:	postgres

Database setup

Technical user for deegree with password ‘deegree’

```
CREATE ROLE deegree LOGIN
 ENCRYPTED PASSWORD 'md5b73ce574b23cf58ac77c8ca9ea0d2b5f'
 NOSUPERUSER INHERIT NOCREATEDB NOCREATEROLE NOREPLICATION;
COMMENT ON ROLE deegree IS 'technical user for deegree FeatureStore
config';
```

Hint:

Use persistent data volume container for productive systems, otherwise you may lose your data!

deegree Webservices

Docker Hub: https://hub.docker.com/r/tfr42/deegree/
Dockerfile: https://github.com/tfr42/deegree-docker

```
docker pull tfr42/deegree
docker run -d --name deegree -p 8080:8080 tfr42/deegree
```

Or with link to postgis container and attached to the deegree log console:

```
docker run --name deegree -p 8080:8080 --link postgis:db
tfr42/deegree
```

Open in browser: http://localhost:8080/deegree-webservices

Navigate to “connections > databases” and create a new connection of type “DataSource” with config template “PostgreSQL (minimal)”.
Change the JDBC URL to `jdbc:postgresql://db:5432/postgres`

Complete configuration file (saved inside the container in directory `/root/.deegree/`):

```
<DataSourceConnectionProvider configVersion="3.4.0"
  xmlns="http://www.deegree.org/connectionprovider/datasource"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.deegree.org/connectionprovider/datasource
http://schemas.deegree.org/jdbc/datasource/3.4.0/datasource.xsd">
  <!-- Creation / lookup of javax.sql.DataSource instance -->
  <DataSource javaClass="org.apache.commons.dbcp.BasicDataSource" />
  <!-- Configuration of DataSource properties -->
  <Property name="driverClassName" value="org.postgresql.Driver" />
  <Property name="url" value="jdbc:postgresql://db:5432/postgres" />
  <Property name="username" value="deegree" />
  <Property name="password" value="deegree" />
  <Property name="poolPreparedStatements" value="true" />
  <Property name="maxActive" value="10" />
  <Property name="maxIdle" value="10" />
</DataSourceConnectionProvider>
```

Build deegree docker container based on a Dockerfile

Readme: https://github.com/tfr42/deegree-docker/tree/master/deegree-webapp-tomcat
Dockerfile

```
git clone https://github.com/tfr42/deegree-docker.git
cd deegree-docker/deegree-webapp-tomcat/
docker build -t deegree/deegree-tomcat .
```

Use the branch “`feature/deegree3_4`” to build the container with deegree 3.4-RC2

Part 2 - configure WFS 2.0 deegree

Start deegree docker container with local deegree workspace directory

Download deegree workspace template for INSPIRE data themes Protected Sites and Cadastral Parcels:
https://dl.dropboxusercontent.com/u/4100192/deegree/deegree-workspace-bundle.zip

Create a new directory `.deegree` in the user home directory and unzip all files into the `~/.deegree` directory.

Stop and delete the docker container `deegree`:

```
docker stop deegree
docker rm deegree
```

Start a **new** container with mounted directory `~/.deegree`:

```
docker run -d --name deegree -v ~/.deegree:/root/.deegree
-p 8080:8080 --link postgis:db tfr42/deegree
```

Create DA DLS serving INSPIRE data theme Protected Sites

Configuration steps needed:

1. Create the database
2. Add the GML application schema to workspace
3. Create the database connection configuration file
4. Create the FeatureStore configuration file
5. Create the WFS service configuration file

Database schema and deegree FeatureStore configuration derived from GML application schema (relational/canonical mode)

1. Create the database
    a. As user `postgres` - ~/.deegree/ddl/protectedsites/create_ps_canonical_db.sql
    b. As user `deegree` connected to `ps_canonical` database - ~/.deegree/ddl/protectedsites/create_ps_canonical_schema.sql
2. Add the GML application schema to workspace (source of XSD)
    a. ~/.deegree/workspace-ps/appschemas/ProtectedSites.xsd
3. Create the database connection configuration file
    a. ~/.deegree/workspace-ps/jdbc/postgresDS_canonical.xml
4. Create the FeatureStore configuration file
    a. ~/.deegree/workspace-ps/datasources/feature/ps_canonical.xml

5. Create the WFS service configuration file
    a. ~/.deegree/workspace-ps/services/wfs_ps_canonical.xml

Attention:
The wizard may skip complex element types. For the GML application schema for ProtectedSites (v.4.0) the element `legalFoundationDocument` is missing in the generated DDL and in the FeatureStore configuration file (see issue #742). More information how to generate the mapping and DDL in paragraph Supporting tools.

Database schema and deegree FeatureStore configuration based on simple GML application mapping (blob mode)

1. Create the database
    a. As user `postgres` - ~/.deegree/ddl/protectedsites/create_ps_blob_db.sql
    b. As user `deegree` connected to `ps_blob` database - ~/.deegree/ddl/protectedsites/create_ps_blob_schema.sql
2. ~~Add the GML application schema to workspace (source of XSD)~~
    a. ~~~/.deegree/workspace-ps/appschemas/ProtectedSites.xsd~~
3. Create the database connection configuration file
    a. ~/.deegree/workspace-ps/jdbc/postgresDS_blob.xml
4. Create the FeatureStore configuration file
    a. ~/.deegree/workspace-ps/datasources/feature/ps_blob.xml
5. Create the WFS service configuration file
    a. ~/.deegree/workspace-ps/services/wfs_ps_blob.xml

Supporting tools

Tools to create the SQL DDL scripts and the deegree FeatureStore configuration files:

- deegree Webservices console (in 3.4 the wizard is broken (see issue #471 and other), use deegree 3.3.18 or deegee CLI utility tool instead!)

Useful docker commands to monitor the container

`docker logs -f deegree` - follow the deegree console output
`docker attach deegree` - attach to the deegree container
You can detach from the container and leave it running with `CTRL-p CTRL-q`. Requires to pass `-it` option to the docker `run` command!
You can stop the container with `CTRL+c`.
`docker exec -it deegree '/bin/bash'` - opens a shell in the running deegree container.
Use exit to disconnect from the container
`docker stats deegree` - This will present the CPU utilization for the container, the memory used and total memory available to the container.
`docker network inspect bridge` - see the IP for each container

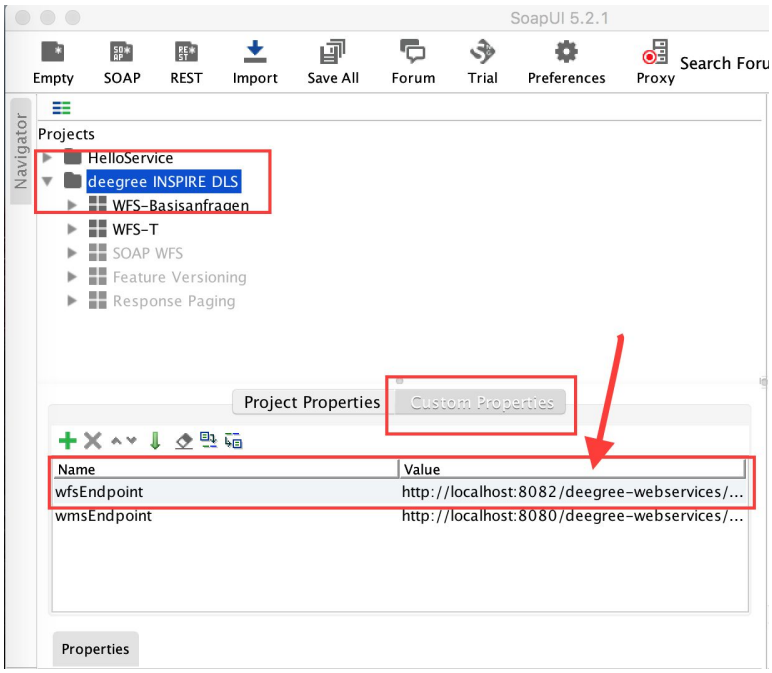
Part 3 - Import test data

Docker hub: https://hub.docker.com/r/tfr42/docker-soapui/
Dockerfile: -

```
docker pull tfr42/docker-soapui
xhost +
docker run --name soapui --rm -t -i -e DISPLAY=:0.0 -v
/tmp/.X11-unix:/tmp/.X11-unix -v ${HOME}/.deegree:/var/opt --link
deegree:deegree tfr42/docker-soapui '/opt/SoapUI/bin/soapui.sh'
```

Setting custom properties

Open the file `/var/opt/test/wfs200-soapui-project.x ml` with SoapUI and select the project root node.

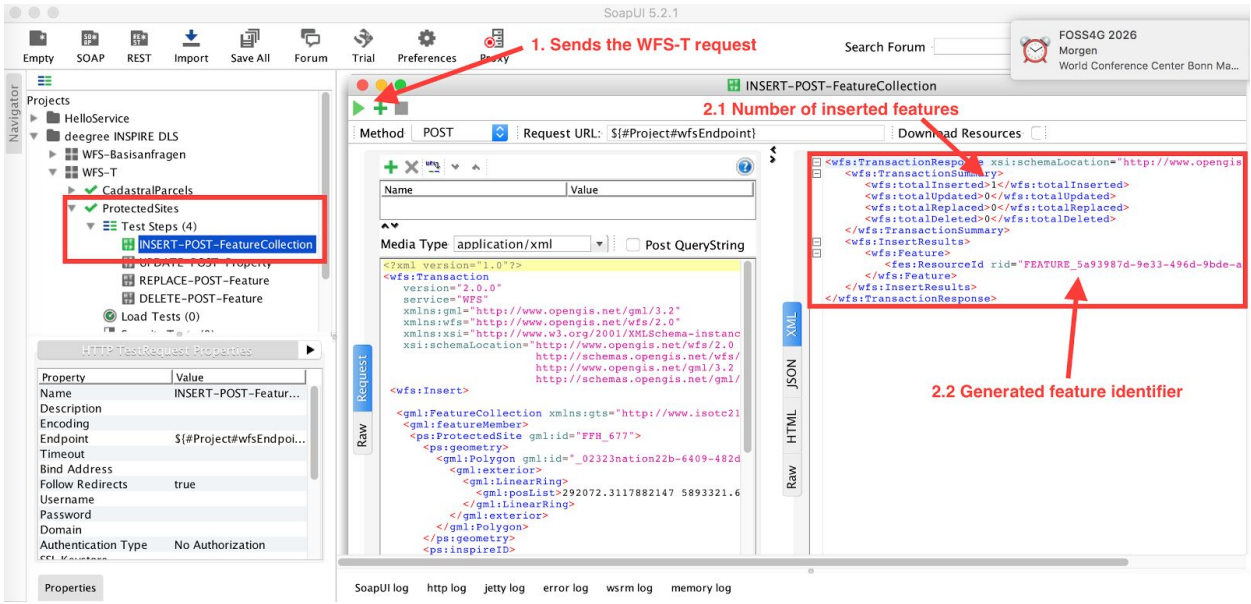


Switch to “Custom Properties” tab and set for property “`wfsEndpoint`”:

- To import into WFS configured with FeatureStore in blob modus use: http://deegree:8080/deegree-webservices/services/wfs_ps_blob
- WFS configured with FeatureStore in relational/canonical modus use: http://deegree:8080/deegree-webservices/services/wfs_ps_canonical

Import sample test data over WFS-T Insert operation

To send a WFS-T Insert submit the test step “WFS-T > ProtectedSites > INSERT - POST-FeatureCollection”:



Switch the `wfsEndpoint` property to the other endpoint and re-submit the WFS-T Insert request.

Part 4 - Retrieve data 

Docker hub: https://hub.docker.com/r/kartoza/qgis-desktop/
Dockerfile: https://github.com/kartoza/docker-qgis-desktop

```
docker pull kartoza/qgis-desktop
```

QGIS 2.14:

```
xhost +
docker run --name qgis-desktop_2_14 -i -t -v
/tmp/.X11-unix:/tmp/.X11-unix -v ${HOME}:/home/${USER} -e
DISPLAY=unix:0 --link deegree:deegree --rm kartoza/qgis-desktop:2.14
'/usr/bin/qgis'
```

QGIS 2.16 (DEV):

```
xhost +
docker run --name qgis-desktop_master -i -t  -v
/tmp/.X11-unix:/tmp/.X11-unix -v ${HOME}:/home/${USER} -e
DISPLAY=unix:0 --link deegree:deegree --rm
kartoza/qgis-desktop:latest '/usr/bin/qgis'
```

Part 5 - Validate deegree Webservice

Docker hub: https://hub.docker.com/r/tfr42/teamengine/
Dockerfile:
https://github.com/tfr42/teamengine/tree/feature/addDockerConfig/teamengine-docker

```
docker pull tfr42/teamengine
docker run -d --name teamengine -p 8088:8080 --link deegree:deegree
tfr42/teamengine
```

Open in browser: http://localhost:8088/teamengine

Use
http://deegree:8080/deegree-webservices/services/wfs_ps_blob?service=WFS&request=GetCapabilities
or
http://deegree:8080/deegree-webservices/services/wfs_ps_canonical?service=WFS&request=GetCapabilities
To run the validation.

Remark

In case TEAM Engine reports an error in the validation results for Simple WFS conformance class and with the fault that for `GetCapabilities` the value '`local`' for the `GetFeature` '`resolve`' parameter is missing, - this is a false negative, see https://github.com/opengeospatial/ets-wfs20/issues/39 for more information.

Troubleshooting

- Can't access docker -
  - check if the docker daemon is running, use `sudo`
- Error while starting docker container -
  - check system resources if memory is still available
  - Remove the container with `docker rm` and re-run the container
- For more hints and tipps check https://docs.docker.com/toolbox/faqs/troubleshoot/
  - For Mac OS : https://docs.docker.com/docker-for-mac/troubleshoot/
  - For Windows: https://docs.docker.com/docker-for-windows/troubleshoot/

Links

Slides

01_T_Introduction.pdf
02_T_INSPIRE-Download-Services.pdf
03_TP_Docker.pdf
04_P_deegree-on-Docker.pdf
05_TP_deegree.pdf
06_P_Configuration-of-a-deegree-INSPIRE-Download-Service.pdf
07_TP_Validation-of-service-and-data.pdf

Docker

https://www.docker.com
https://docs.docker.com
https://hub.docker.com
http://linoxide.com/linux-how-to/run-gui-apps-docker-container/

Talks about Docker and GIS

https://www.fossgis.de/konferenz/2015/programm/events/847.de.html
https://2016.foss4g-na.org/session/spatial-data-processing-docker
http://2016.foss4g.org/talks.html#146
http://training.runcloudrun.com/roadshow/

deegree

https://github.com/deegree/deegree3
http://www.deegree.org

deegree on Docker Hub

https://hub.docker.com/r/tfr42/deegree/

OGC CITE TEAM engine

https://github.com/opengeospatial/teamengine
http://opengeospatial.github.io/teamengine/
http://cite.opengeospatial.org
http://cite.opengeospatial.org/teamengine/

INSPIRE

http://inspire.ec.europa.eu/
http://inspire-geoportal.ec.europa.eu/validator2/
http://inspire-regadmin.jrc.ec.europa.eu/dataspecification/
http://inspire.ec.europa.eu/events/conferences/inspire_2012/presentations/69.pdf
http://www.slideshare.net/ChrisSchubert1/inspirehandsondatatransformation

OSGeo

https://live.osgeo.org/en/index.html
http://live.osgeo.org/en/overview/deegree_overview.html
http://geocontainers.org/
https://wiki.osgeo.org/wiki/DockerImages

Data and services

WMS with OSM data
http://ows.terrestris.de/osm/service?SERVICE=WMS&VERSION=1.1.1&REQUEST=GetCapabilities