FOSS4G 2010

TileCache,GeowebCache and MapProxy - a technical and usability comparison

Johannes Weskamm, terrestris GmbH & Co. KG

What is a TileServer?

- *TileCacheServer* or *TileServer* save the content of a WMS as *Tiles* in a Cache
- Tiles can be pre-rendered (seeding) or generated dynamically on demand
- TileServer normally act as proxy between Client and the WMS-Server

What is a TileServer?

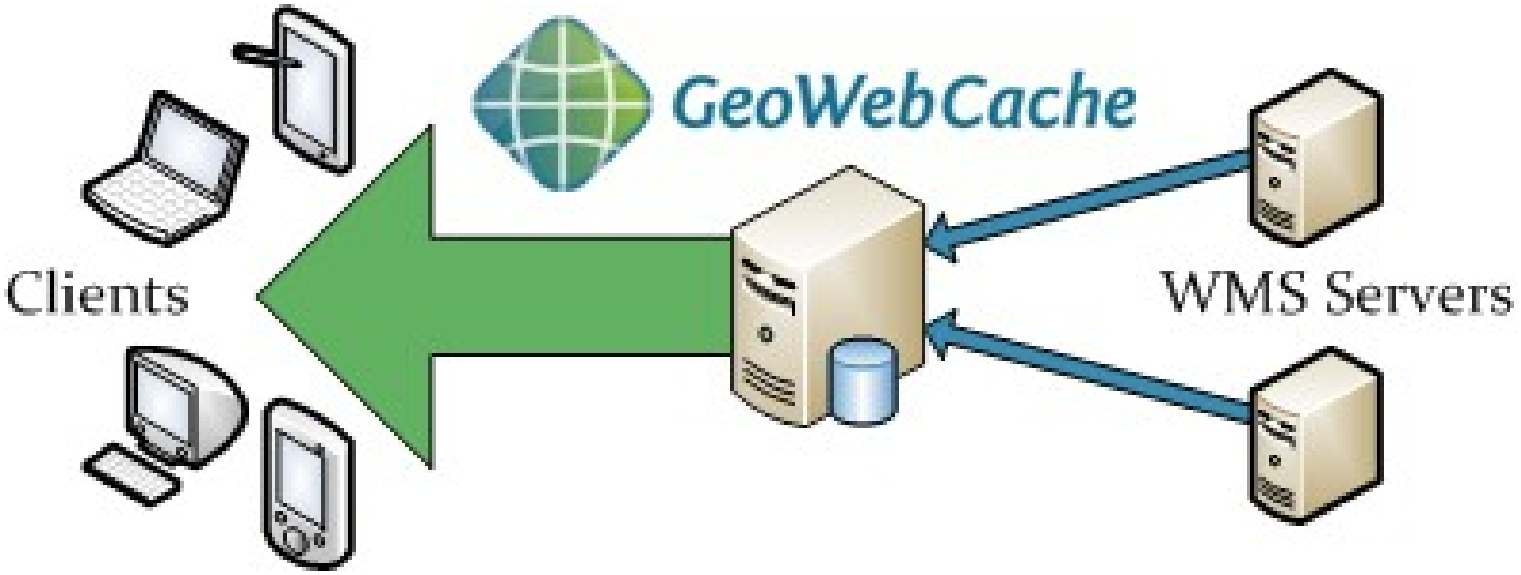

Figure 1: Typical TileCahce Infrastructure
(Source: http://geowebcache.org/trac/attachment/wiki/WikiStart/gwc_medium.png)

Why use a TileServer?

· Faster delivery of the WMS data to the client

· Improvements in usability

· Reduced load on the WMS-Server through pregenerated tiles
(rendering has only to be done once)

How does it work

- The TileServer requests the predefined data from the WMS-Server
- When using Metatiles, the response gets cut into tiles → *Tiling*
- Tiles get saved in the cache and are accessible for the client

Tiling of Metatiles

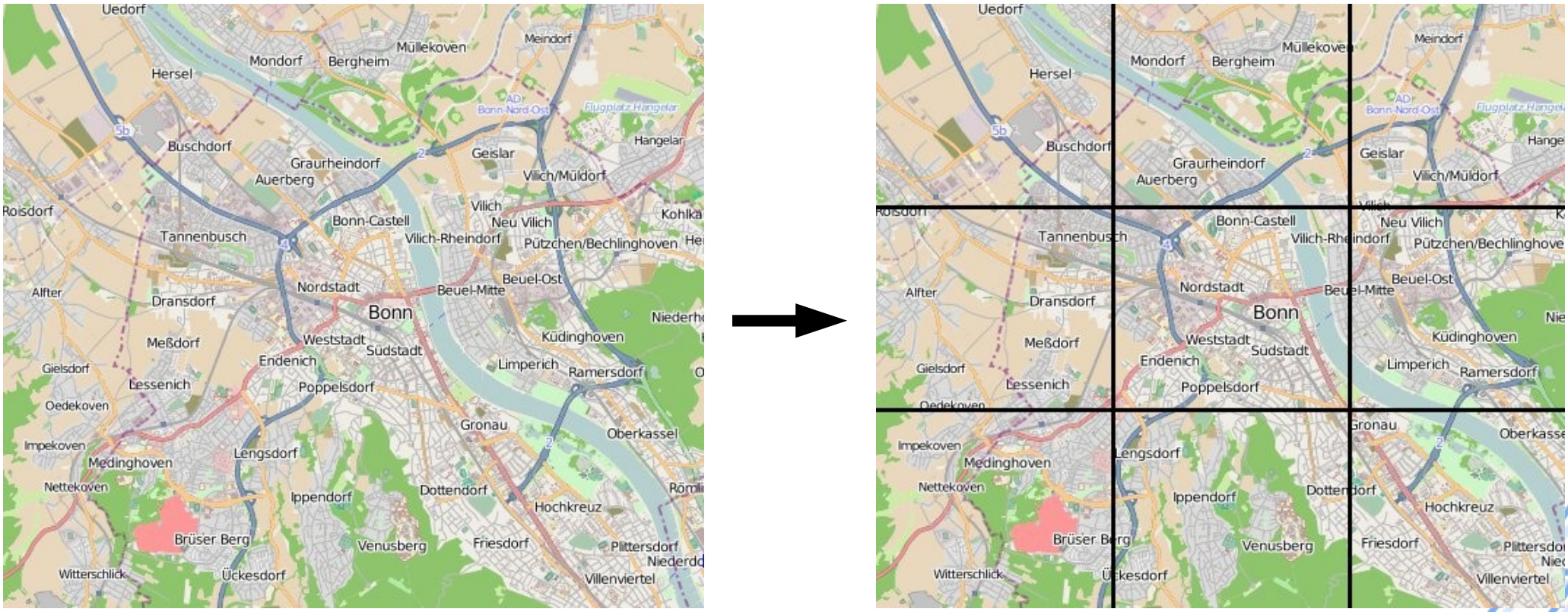

Figure 2: Tiling mechanism

Caching Methods

TileServer often support two types of caching the data:

·

· ***seeding*** (manually, requests and caches all tiles)

– temporarily high load on the WMS Server

– needs more time and diskspace

– client gets always a fast response

·

·

· **dynamic** (generating tiles only on demand)

– fills the cache over time through user requests

– longer responsetime for the client if requesting a non cached area (only for the first time)

Caching Methods

TileServer can use both approaches:

· Requests, which are not already in the (pregenerated) cache can be fullfilled through adding the missing data on demand

·These tiles will be added to the cache for further requests

Communication between Client and TileServer

WMS-C (Web Mapping Service – Cached)

· Results of discussions on the FOSS4G 2006
· Constrains requests to a predefined grid

Example:

http://wxs.ign.fr/geoportail/wmsc?LAYERS=ORTHOIMAGERY.ORTHOPHOTOS&
EXCEPTIONS=text/xml&FORMAT=image/jpeg&
SERVICE=WMS&VERSION=1.1.1&REQUEST=GetMap&STYLES=&
SRS=IGNF:GEOPORTALFXX&BBOX=189952,5432960,190080,5433088&
WIDTH=256&HEIGHT=256&TILED=true

Communication between Client and TileServer

TMS (Tile Map Service)

· Specification of OSGEO

· Based on the WMS-C recommendation

· No offical Standard

· Supporting Servers:
TileCache, MapProxy and GeoWebCache,
on the client side:
WorldKit, Openlayers, Cadcorp SIS und Merkaartor

· Predecessor of WMTS
·Example:
http://tilecache.osgeo.org/wms-c/tilecache.py/1.0.0/basic/4/16/12.png

Communication between Client and TileServer

WMTS (Web Map Tile Service)

· Official Standard of OGC since April 2010
· No replacement for WMS, more a extension

· Communication is realized with a ServiceMetadata-Document
· Extends the request with a „getTile“ parameter

·Supporting Servers:
CubeSERV, MiraMon Map Server, GeoWebCache,
in the future: Esri, TileCache
·Supported clients: Openlayers, Esri
Example:
http://www.maps.cat/maps.cgi?service=WMTS&request=GetTile&version=1.0.0
&layer=etopo2&style=default&format=image/png&TileMatrixSet=WholeWorld_C
RS_84&TileMatrix=10m&TileRow=1&TileCol=3

Client-Support

· Besides TMS, MapProxy, TileCache and Geoserver can be accessed as a standard WMS-Server

· Problem: How to respond to requests with a scale not in the cache?

· Solution: MapProxy and GeoWebCache use resampling

· Picture quality and size of labels and symbol varies

Client-Support

· OpenLayers works fine with cached WMS / TMS when using fixed resolutions
(*resolutions: [0.703125, 0.3515625, 0.17578125, …]*)

· QGIS / ArcMap only work with WMS of MapProxy or GeoWebCache

· Support for WMTS is planned

TileCache

· Python-based

·

· Developed by MetaCarta Labs

· Installation really simple: Only requires a Webserver and Python-installation

· Acts as a CGI-Script

· Platformindependent

TileCache

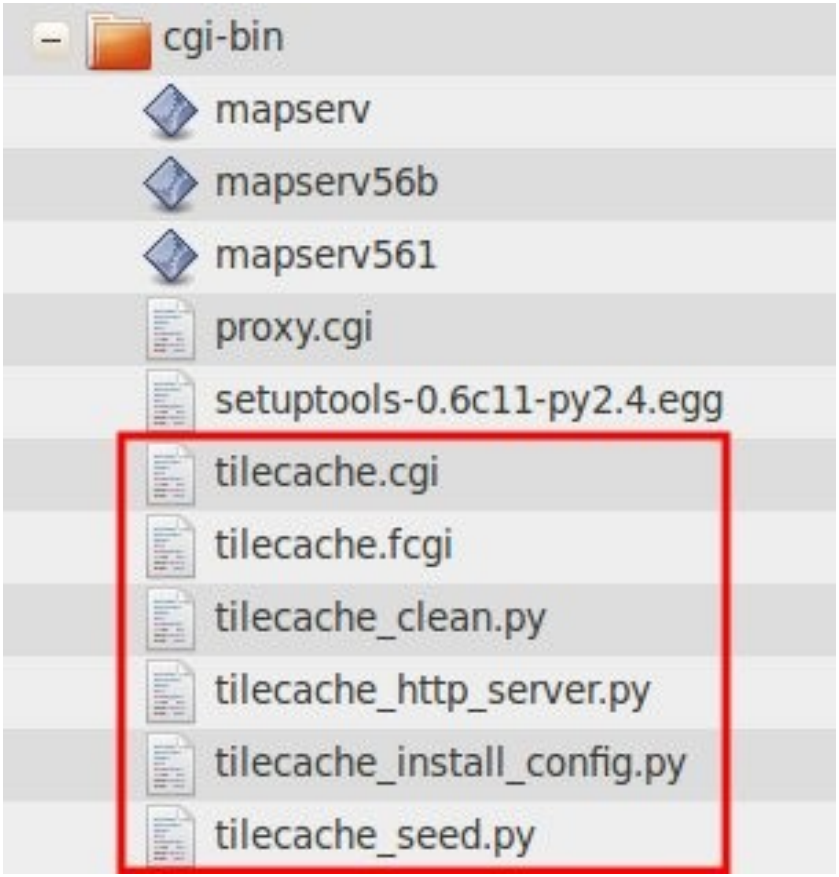

Figure 3: TileCache Python-Scripts

TileCache

· Tiles can be saved in a DiskCache or a MemoryCache

· Only WMS Clients with support for fixed scales can use the WMS

· TileCache does not support seeding on demand over WMS

· Implementation of WMTS is planned

TileCache

Typical configuration example (tilecache.cfg):

```
[Harbour]
type=WMS
size=256,256
url=http://wsvmapserv.wsv.bund.de/wms_ienc
mime_type=image/png
extension=png
Resolutions=0.175781,0.1,0.075,0.05,0.025,
0.007812
bbox=5.0,47.0,15.0,55.0
metaTile=true
metaBuffer=100
```

TileCache

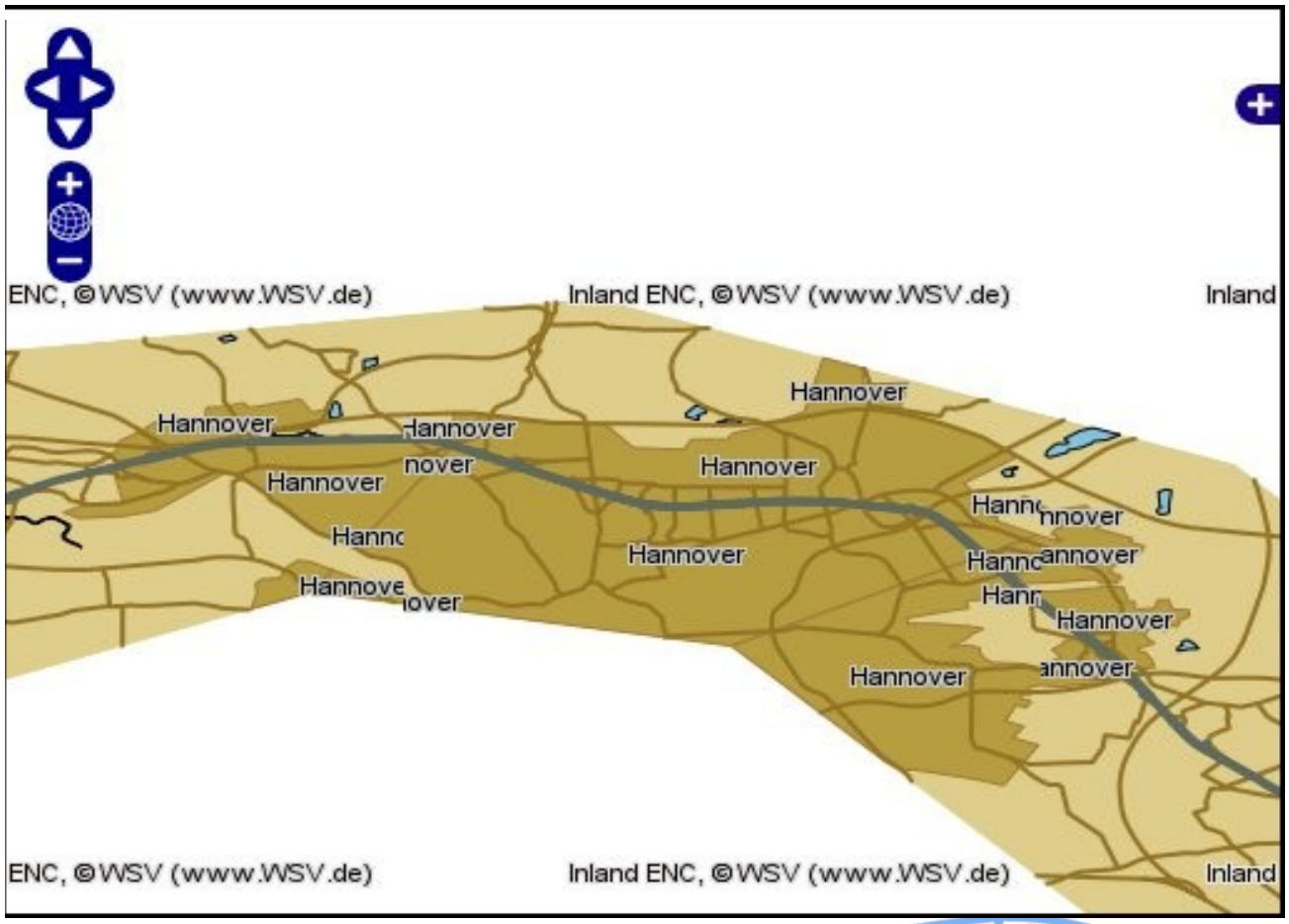

Figure 4: : TileCache w/o Metatile and Metabuffer

TileCache

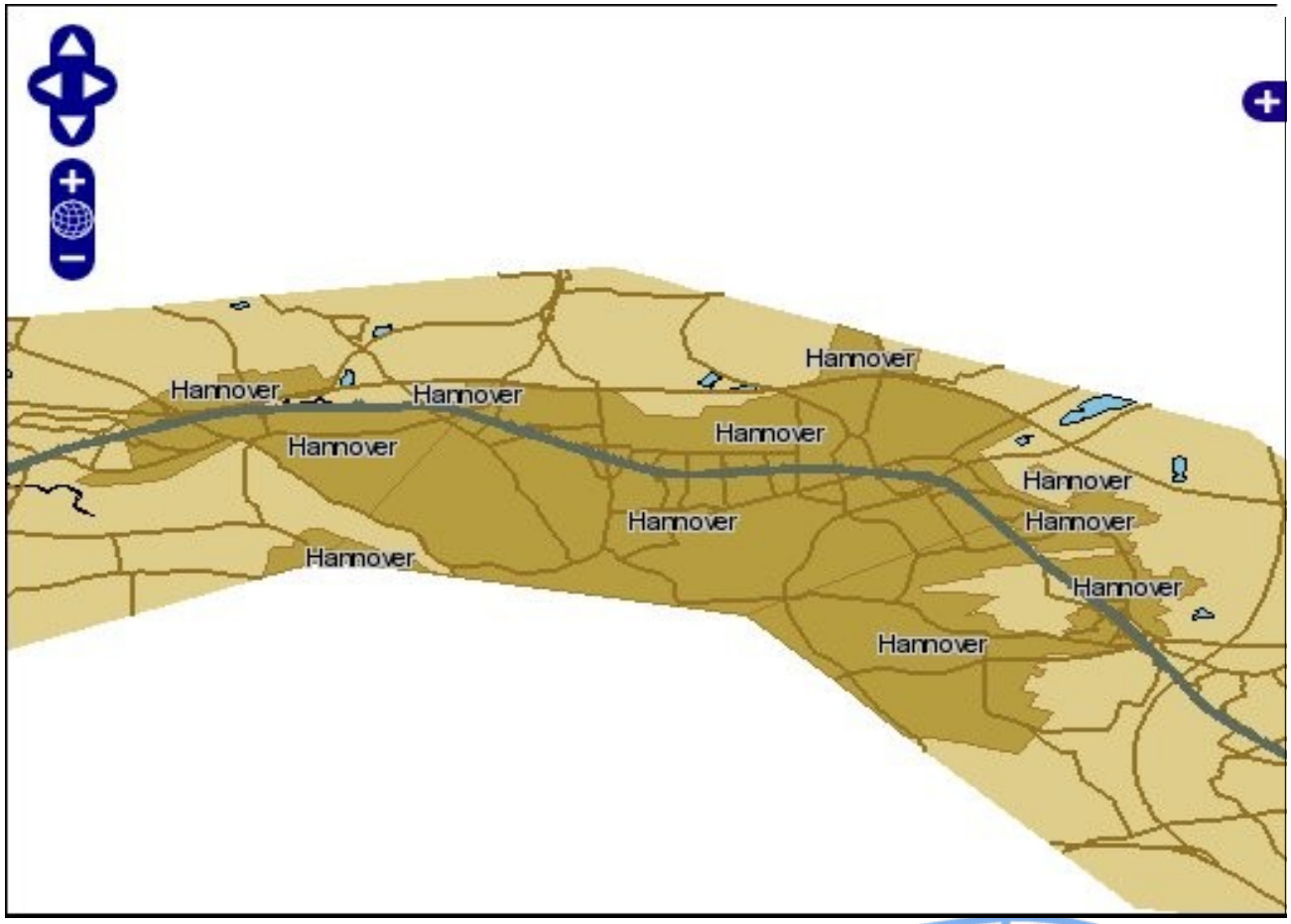

Figure 5: : TileCache with 3x3 Metatile and 100px Metabuffer

TileCache

· Seeding example:

tilecache_seed.py [options] <layer> [<zoom start> <zoom stop>]

tilecache_seed.py --bbox "7,50,8,52" basic 10 12

GeoWebCache

· Java-based, platformindependent

· Developed by OpenGEO

· Installation: bundled with GeoServer or stand alone

· requires Java and a servlet-server, i.e. Apache Tomcat

GeoWebCache

Supported Services (and clients)

WMTS,
WMS-C,
TMS,
KML / Google Earth,
Google Maps oder
Microsoft Virtual Earth

GeoWebCache

· Enabling resampling / recombining in GeoWebCache:

edit the geo-webcache-servlet.xml

<property name="fullWMS"><value>FALSE</value></property>

and set FALSE to TRUE

GeoWebCache

·Creating a cache manually with GeoWebCache

Figure 6: Webfrontend of Geowebcache

GeoWebCache

- Example configuration:

```
<gwcConfiguration>
        <layers>
            <wmsLayer>
                <name>topp:states</name>
                <wmsUrl><string>http://atlas.opengeo.org:8080/geoserver/wms</string>
                </wmsUrl>
                <metaWidthHeight>
                  <int>3</int>
                  <int>3</int>
                </metaWidthHeight>
                <gutter>100</gutter>
            </wmsLayer>
        </layers>
</gwcConfiguration>
```

GeoWebCache

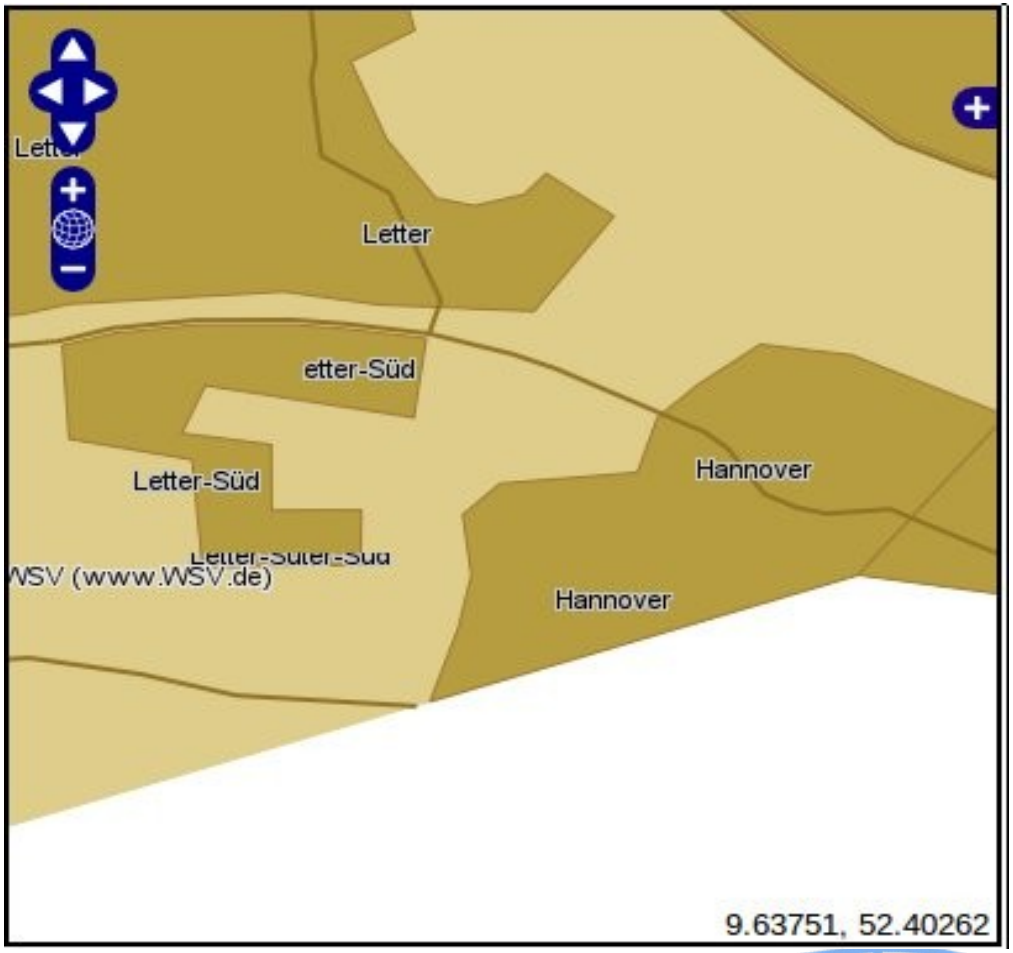

Figure 7: Geowebcache with 3x3 Metatiles and 100px Gutter

GeoWebCache

other features:

· Creation of individual CQL filters

· Can use time und elevation parameters of WMS 1.3.0

· Multiple styles for single layers

· Supports simple WFS caching with geoserver

·Metadata of tiles will be stored in H2 database which allows better cache handling

MapProxy

· Pythonbased, platformindependent

· Developed by Omniscale

· Installation relatively simple in a “virtual python environment”

· Configurationfiles use the YAML-Format

MapProxy

· Supported Services (and clients)
WMS (1.0.0, 1.1.1, 1.3.0),
TMS (1.0.0)
KML (2.2)

· Uses resampling to serve tiles in all scales

MapProxy

· Cache can be seeded or filled dynamically

· Example:

mapproxy_seed -c4 -f proxy.yaml seed.yaml

· GetFeatureInfo can be activated in the services.yaml:

```
type: cache_wms
  wms_opts:
  version: 1.3.0
  featureinfo: True
```

MapProxy

Configuration example:

```
osm:
·       md:
·           title: Omniscale OSM WMS - osm.omniscale.net
·       param:
·           format: image/png
·           srs: ['EPSG:4326', 'EPSG:900913']
·           res: [0.175781, 0.1,0.075, 0.05, 0.025, 0.007812 ]
·           tile_size: [256, 256]
·       sources:
·       - type: cache_wms
·         req:
·           url: http://osm.omniscale.net/proxy/service?
·           layers: osm
·
```

MapProxy

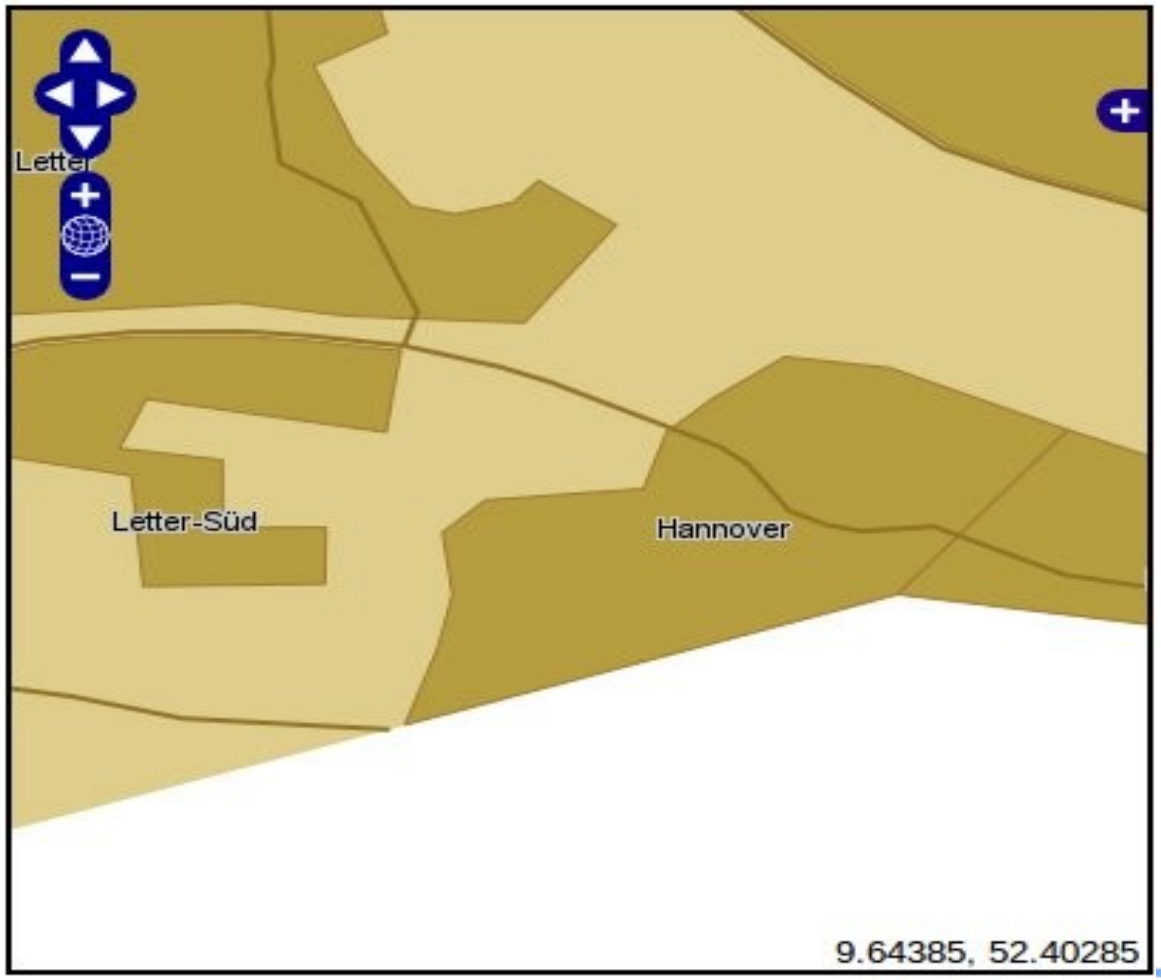

Figure 8: MapProxys with 3x3 Metatiling and 100px MetaBuffer

MapProxy

other Features

· Reprojection of the cache on the fly

· Seeding can be limited to a shapefile (or WKT)

· New PIL makes PNG8 transparent

· Old tile only gets deleted when recreation of the new one is completed

· White / empty tiles can be linked instead of saved

· Creation of watermarks

Comparison Chart

	TileCache	GeoWebCache	MapProxy
Supported clients	TMS Capable Clients, WMS-Clients only with support for fixed scales i.e. OpenLayers; WorldWind	all TMS / WMS-Clients Stretching / Cropping not configurable, only supports PNG	all TMS/ WMS-Clients Resampling configurable
Developers	Metacarta Labs	OpenGeo	Omniscale
Support	Mailinglist, Tracker	Mailinglist, Tracker, IRC	Mailinglist, Tracker, commercial support
Supported services	WMS, WorldWind, TMS 1.0.0	WMTS, WMS-C, WMS, TMS 1.0.0, KML	WMS 1.0.0, 1.1.1, 1.3.0 TMS 1.0.0 KML 2.2
Functionality of the WMS: GetFeatureInfo GetLegend	 Not possible Not possible	 possible possible	 possible Not possible

Comparison Chart

	TileCache	GeoWebCache	MapProxy
Displayproblems with labels and symbols	Can be eliminated witth Metatiles and MetaBuffer, but only in JPEG / GIF-Format → low quality	Can be eliminated witth Metatiles and Gutter, problems in practice → still cutoffs in labels and symbols	Can be eliminated with Metatiles and MetaBuffer, works well
Installation and configuration	simple	moderate	simple / moderate
Avoiding white tiles	no	no	yes
Reprojection on the fly	no	yes, if used in Geoserver-Bundle	yes, native
Seeding	yes	yes	yes
Caching on demand (over WMS)	no	yes	yes

TileCaching problems

Refilling the cache needs time!

·Select layers wisely

Cutoff Labels and Symbols:

Can be *minimized* with Metatiling and using a Metabuffer

· Problems can be solved if you have access to the WMS configuration:

·Set the dynamic positioning to fixed

Live Demo

Circumstances:

· OpenLayers-Client

·

· TileServers installed on the same machine

· TileServers use the same configuration:

Tilesize:	*256x256*
SRS:	*EPSG:900913 / 31466*
Metatiling:	*1x1 – 1 Tile*
MetaBuffer / gutter:	*0 Pixel*
Imageformat:	*JPEG / PNG*