Omniscale

Geodaten absichern mit MapProxy

Oliver Tonnhofer
Omniscale GmbH & Co. KG

Omniscale

Über uns

- Omniscale GmbH & Co. KG, Oldenburg, DE
- OpenSource WebGIS- und Serverentwicklung
- OpenStreetMap Kartendienste
- MapProxy und Imposm Entwicklung, Support und Schulungen

Omniscale

Inhalt

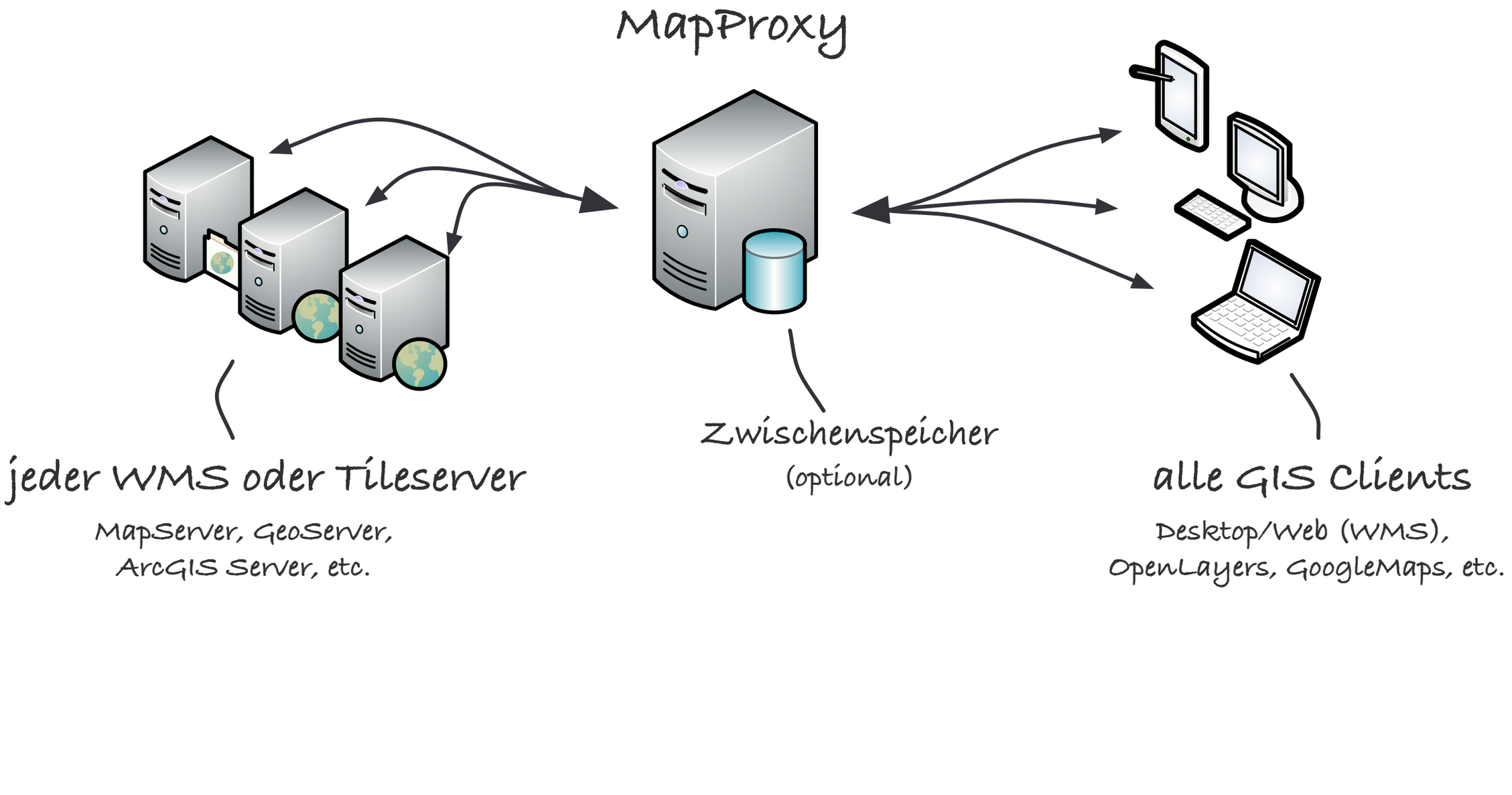
Was ist MapProxy?

Was ist Absicherung?

Absicherung mit MapProxy

Möglichkeiten

Was ist MapProxy?


MapProxy
jeder WMS oder Tileserver
MapServer, GeoServer,
ArcGIS Server, etc.
Zwischenspeicher
(optional)
alle GIS Clients
Desktop/Web (WMS),
OpenLayers, GoogleMaps, etc.

Was ist Absicherung?

Was ist Absicherung?

Nicht jeder darf alles!

Benutzer A darf alles

Benutzer B darf etwas

Benutzer C darf nichts

Benutzer D nur manchmal

Authentifizierung

- Wer? (Identifizierung)
  - Benutzer
  - Benutzergruppe
  - Rechner
  - Anwendung
  - Netzwerk

Autorisierung

- Was darf X?
  - WMS Anfragen
    - GetMap/GetFeatureInfo/GetLegend
  - Kachel Anfragen (TMS/WMTS/KML)
  - Layers
  - Geographische Ausdehnung

Omniscale

Wer

Authentication/Authentifizierung

…darf was?

Authorization/Autorisierung

Omniscale

Authentifizierung	Autorisierung
Benutzer mit Cookie XYZ	Darf alle WMS Anfragen
Rechner mit IP 1.2.3.4	Keine FeatureInfo für Layer A
Nutzer des WebGIS ABC	Nur Kachelanfragen für Layer B
Desktop GIS mit Benutzername/Passwort	Nur Ausschnitt X

Omniscale

Benutzerdaten

Textdatei

.htpasswd

Datenbank

externer Dienst

Authentifizierungsmethoden

•

Autorisierungsmöglichkeiten

•

Benutzerdatenbanken

Authentifizierungsmethoden

•

Autorisierungsmöglichkeiten

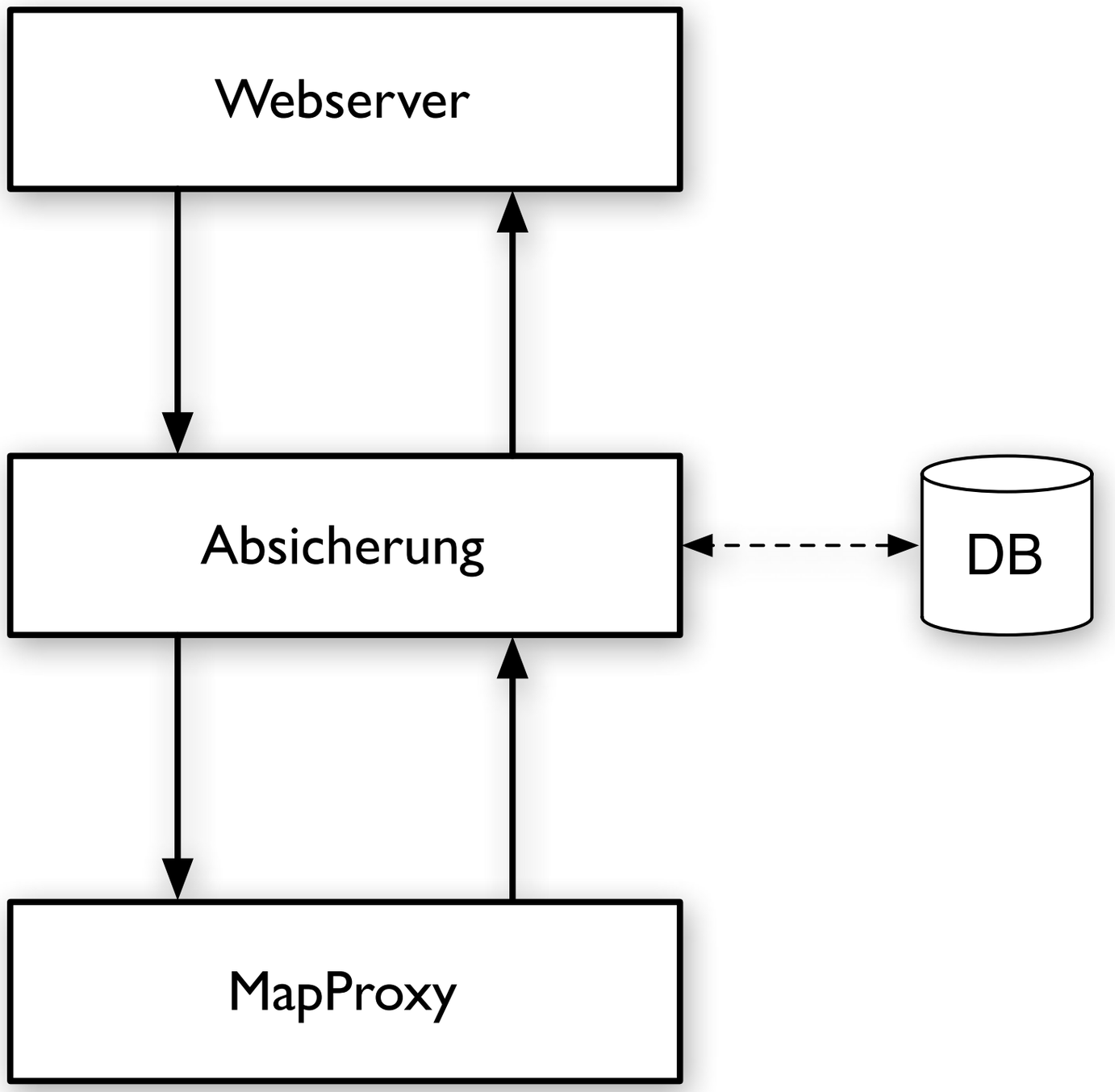
•

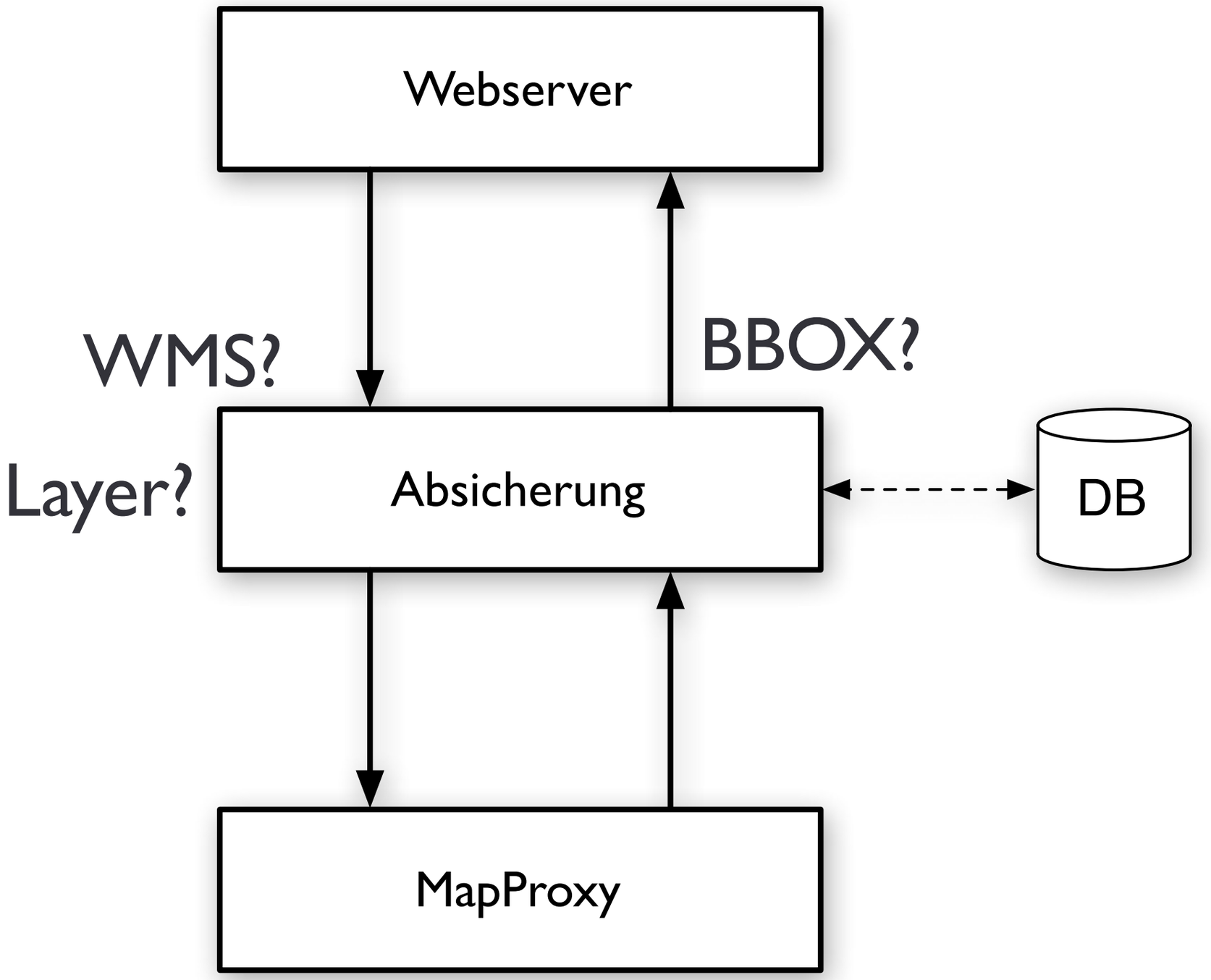
Benutzerdatenbanken

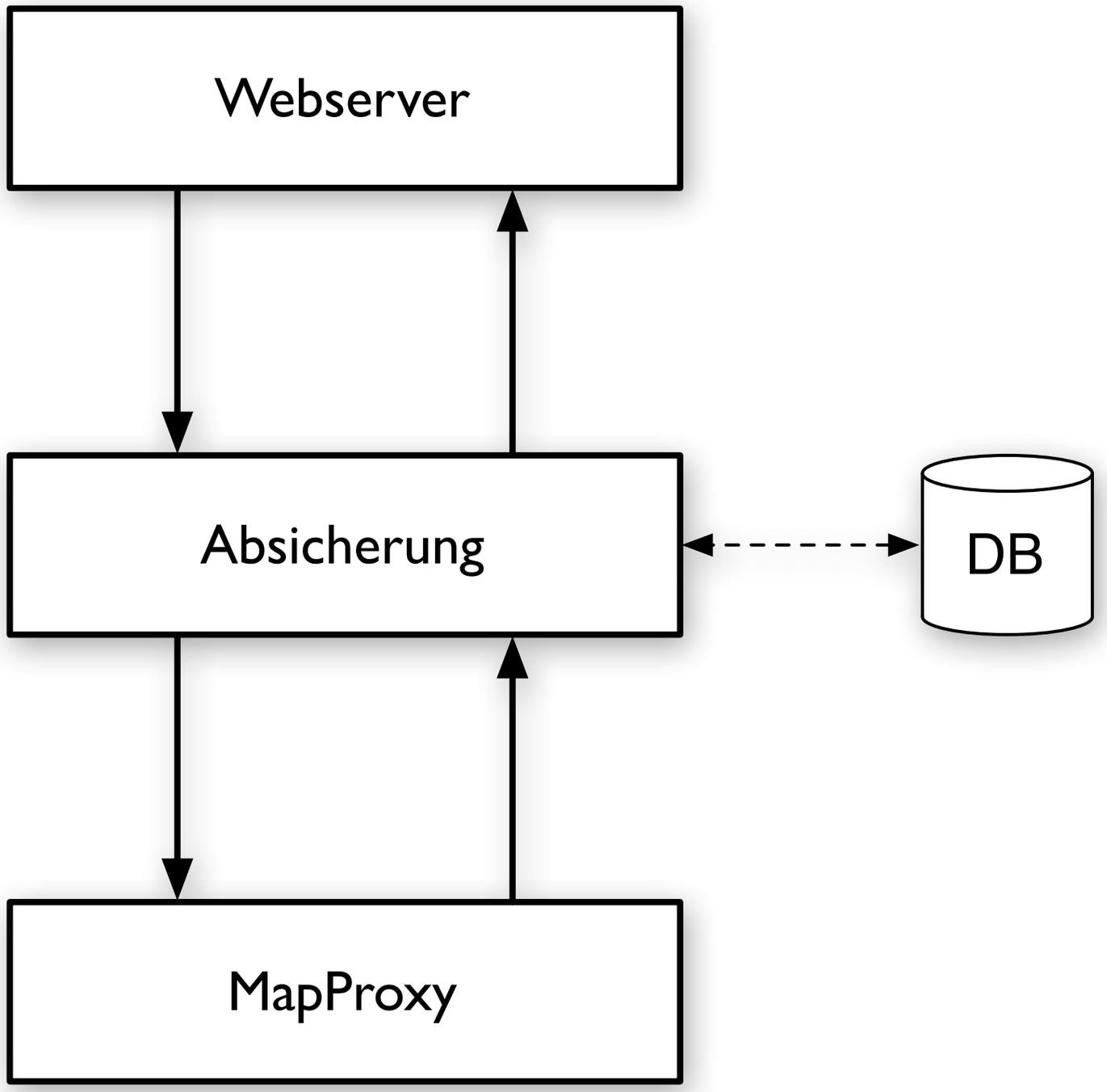
Unzählige Kombinationsmöglichkeiten
Keine Standardlösung

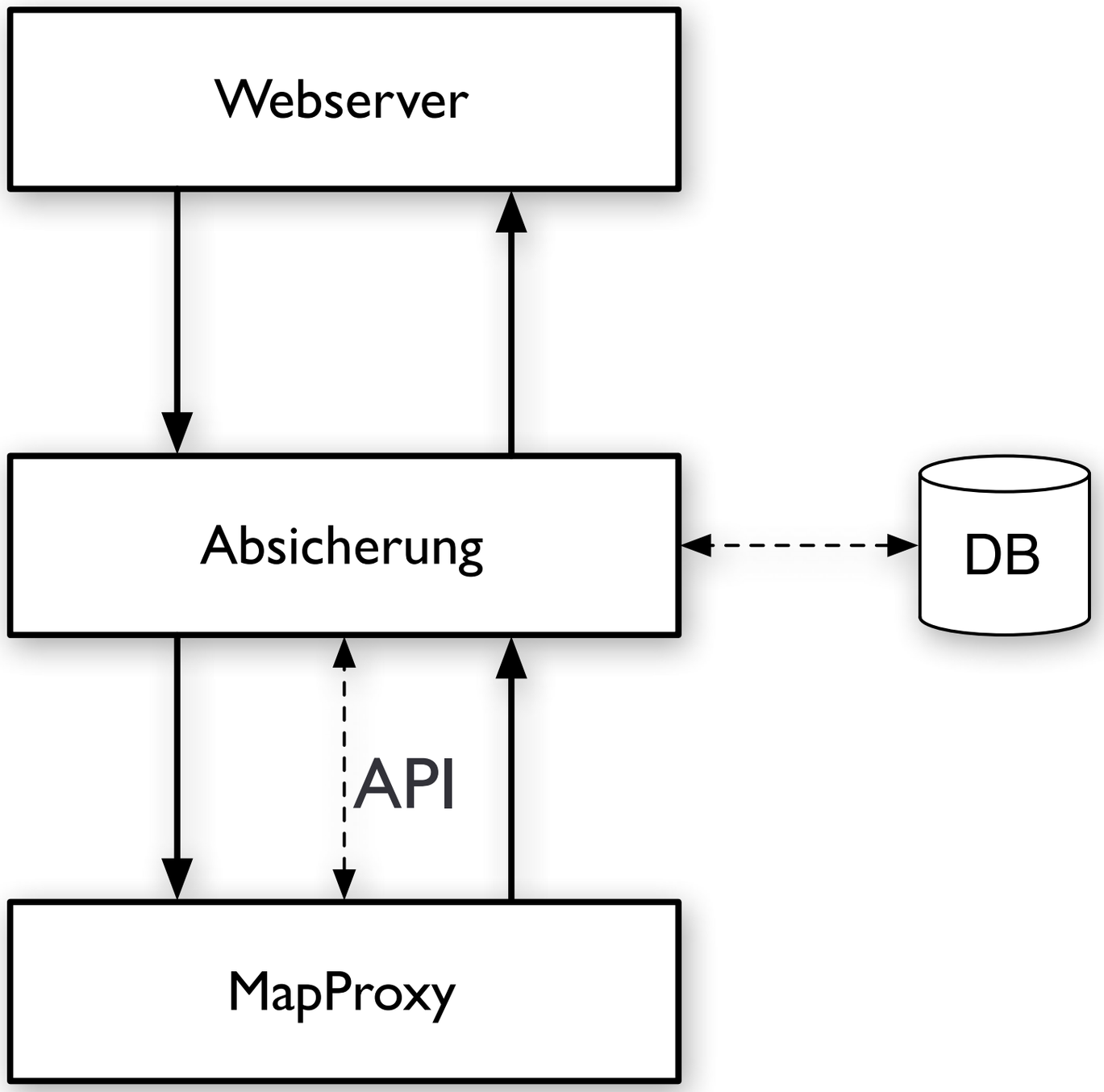
MapProxy

- Absicherung über Zwischenschicht
- Individuell anpassbar


Webserver
Absicherung
DB
MapProxy


Webserver
WMS?
BBOX?
Layer?
Absicherung
DB
MapProxy


Webserver
Absicherung
DB
MapProxy


Webserver
Absicherung
DB
API
MapProxy

Omniscale

Zwischenschicht MapProxy

Authentifizierung

Verarbeitung der Anfrage

Autorisierung

Durchsetzung der Einschränkungen

Autorisierung

- Teilweise Unterstützung durch MapProxy
- Zwischenschicht
  - Benutzerrechte ermitteln
- MapProxy
  - Benutzerrechte durchsetzen

Omniscale

Zwischenschicht

WSGI

Web Service Gateway Interface (WSGI)
analog zur Servlet API in Java

Bindeglied zwischen beliebigen Python Webanwendung und Webservern

WSGI Anwendung

Serverkonfiguration für Apache mod_wsgi:

WSGI Anwendung

Serverkonfiguration für Apache mod_wsgi:

```
from mapproxy.wsgiapp import make_wsgi_app
application = make_wsgi_app('./mapproxy.yaml')
```

WSGI Middleware

Verschachteln von WSGI Anwendungen

Server ruft Middleware auf
Middleware ruft Anwendung auf

WSGI Middleware

Serverkonfiguration für Apache mod_wsgi:

WSGI Middleware

Serverkonfiguration für Apache mod_wsgi:

```
from mapproxy.wsgiapp import make_wsgi_app
mapproxy_app = make_wsgi_app('./mapproxy.yaml')
application = middleware(mapproxy_app)
```

Authentifizierung

Authentifizierung

- Keine Unterstützung in MapProxy
- Zwischenschicht (Middleware) übernimmt Authentifizierung

Omniscale

WSGI Authentication Middleware

```
def middleware(app):
    def application(environ, start_response):
        #
        # Authentifizierung
        #
        return app(environ, start_response)
    return application
```

Omniscale

WSGI Authentication Middleware

```
def middleware(app):
   def application(environ, start_response):
      user, passwd = get_basic_auth(environ)
      if check_passwd(user, passwd):
         environ['REMOTE_USER'] = user
      return app(environ, start_response)
   return application
```

Omniscale

WSGI Authentication Middleware

```
def middleware(app):
    def application(environ, start_response):
        user, passwd = get_basic_auth(environ)
        if check_passwd(user, passwd):
            environ['REMOTE_USER'] = user
        return app(environ, start_response)
    return application
```

repoze.who

- Identifizierungs- und Authentifizierungs-Framework für WSGI
- WSGI Middleware
- Zahlreiche Plug-Ins
- Erweiterbar

repoze.who

- Identifier
  - Session Cookie oder URL Token
- Challenger
  - Passwortabfrage, Weiterleitung zur Login-Seite
- Authenticator
  - Benutzername/Passwort abgleich mit DB

Omniscale

repoze.who Middleware

repoze.who Middleware

```
from mapproxy.wsgiapp import make_wsgi_app
mapproxy_app = make_wsgi_app('./mapproxy.yaml')

application = PluggableAuthenticationMiddleware(
     mapproxy_app,
     identifiers,
     authenticators,
     challengers,
)

application = middleware(mapproxy_app)
```

Omniscale

repoze.who Plug-Ins

repoze.who Plug-Ins

```
basicauth = BasicAuthPlugin('Abgesicherter Bereich')
identifiers = [('basic', basicauth)]
challengers = [(‘basic’, basicauth)]

htpasswd = HTPasswdPlugin(‘/etc/opt/mapproxy/htpasswd’)
authenticators = [('basic',htpasswd)]
```

repoze.who Plug-Ins

- Identifier
  - AuthTktCookiePlugin, BasicAuthPlugin
- Challenger
  - BasicAuthPlugin, RedirectorPlugin
- Authenticator
  - HTPasswdPlugin, SQLAuthenticatorPlugin

Eigenes Plugin

```
class BobAuthenticator():
    def authenticate(self, environ, identity):
        login = identity['login']
        password = identity['password']

        if login == ‘bob’ and password == ‘letmein’:
            return ‘bob’

        return None
```

Benutzerdatenbank

- Textdatei
- Datenbank (SQLAlchemy)
- LDAP/etc.

Omniscale

Autorisierung

Autorisierung

Was darf X?

- Autorisierungsfunktion entscheidet
- MapProxy setzt Einschränkungen durch

Autorisierung

- `full`
  - voller Zugriff
- `none`
  - kein Zugriff (HTTP 403 Forbidden)
- `unauthorized`
  - Benutzer muss sich noch Authentifizieren (HTTP 401 Unauthorized)
- `partial`
  - teilweiser Zugriff

Autorisierungsfunktion

```
def authorize(service, environ, layers=[], **kw):
    return {'authorized': 'full'}
```

Autorisierung

```
def middleware(app):
   def application(environ, start_response):
      environ['mapproxy.authorize'] = authorize
      return app(environ, start_response)
   return application
```

Autorisierungsfunktion

```
def authorize(service, environ, layers=[], **kw):
    if environ[“REMOTE_USER”] == “bob”:
        return {'authorized': 'full'}
    else:
        return {'authorized': 'none'}
```

Beispiel `partial`

- WMS Capabilities Dokument enthält nur Layer 1 & 2
- Nur Layer 2 ist Queryable
- GetMap für Layer 1 & 3 gibt nur Layer 1 zurück

Beispiel `partial`

```
{
   'authorized': 'partial',
   'layers': {
      'layer1': {
         'map': True,
         'featureinfo': False,
      },
      'layer2': {
         'map': True,
         'featureinfo': True,
      }
   }
}
```

- WMS Capabilities Dokument enthält nur Layer 1 & 2
- Nur Layer 2 ist Queryable
- GetMap für Layer 1 & 3 gibt nur Layer 1 zurück

Omniscale

limited_to

```
{
  'authorized': 'partial',
  'layers': {
    'layer1': {
      'map': True,
      'limited_to': {
        'geometry':
           [-10, 0, 30, 50],
        'srs': 'EPSG:4326',
      },
    },
...
```

limited_to

```
{
   'authorized': 'partial',
   'layers': {
     'layer1': {
       'map': True,
       'limited_to': {
         'geometry':
            [-10, 0, 30, 50],
         'srs': 'EPSG:4326',
       },
     },
...
```

- WMS Capabilities mit beschränktem Layer Extent

limited_to

limited_to

```
...
        'layer2': {
          'map': True,
          'limited_to': {
            'geometry': 'POLYGON((...))',
            'srs': 'EPSG:4326',
          },
        },
...
```

limited_to


Hamburg
Hannover
Berlin
Kassel
Leipzig
Köln
Wiesbaden
Regensburg
Stuttgart
München

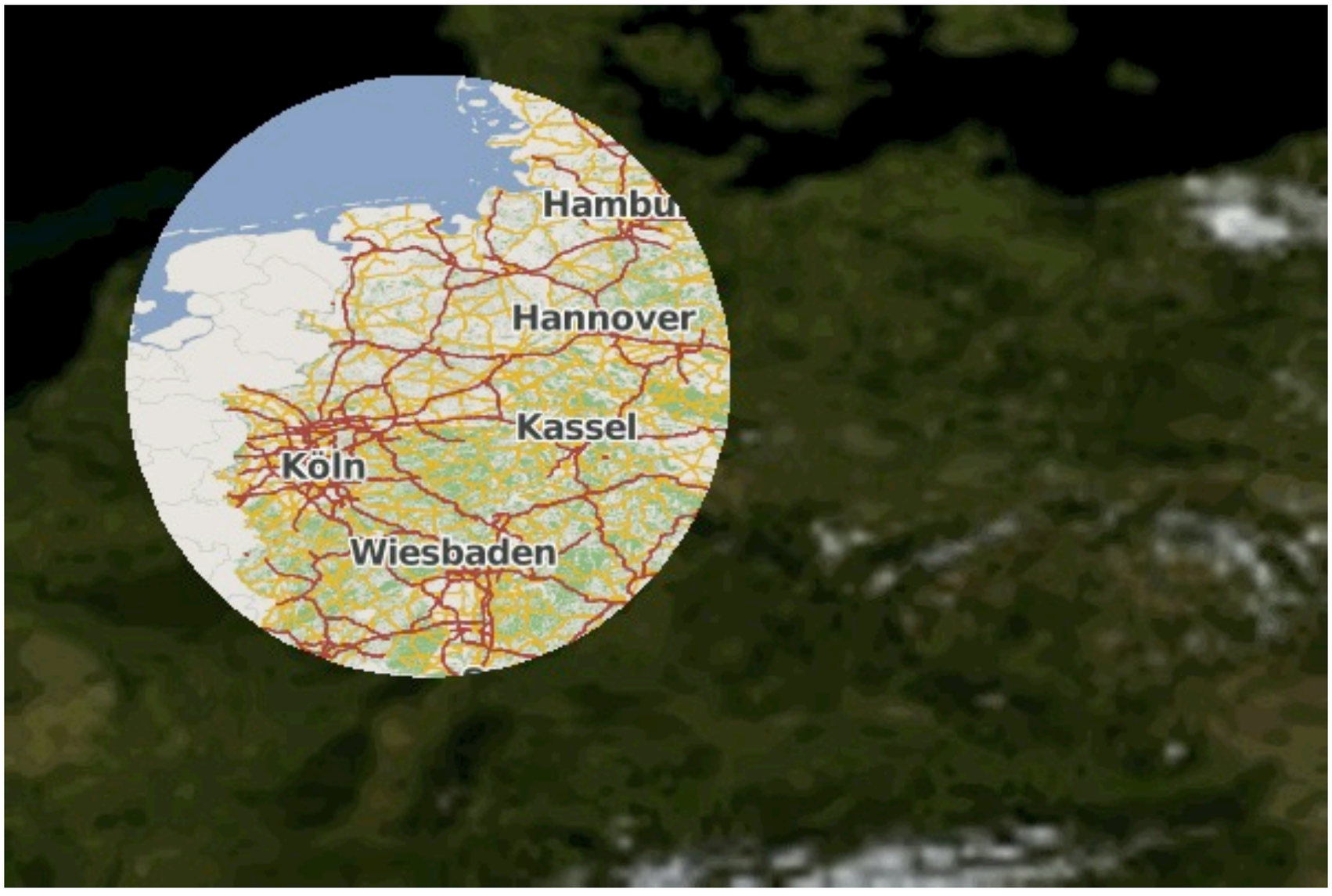
limited_to

```
...
        'layer2': {
          'map': True,
          'limited_to': {
            'geometry': postgis_query_result,
            'srs': 'EPSG:4326',
          },
        },
    ...
```

limited_to

```
...
        'layer3': {
          'map': True,
          'limited_to': {
            'geometry': Point(8, 53).buffer(2)
            'srs': 'EPSG:4326',
          }
        }
    ...
```

limited_to



WMTS/TMS

Omniscale

Möglichkeiten

Layer nur Intern Freigeben

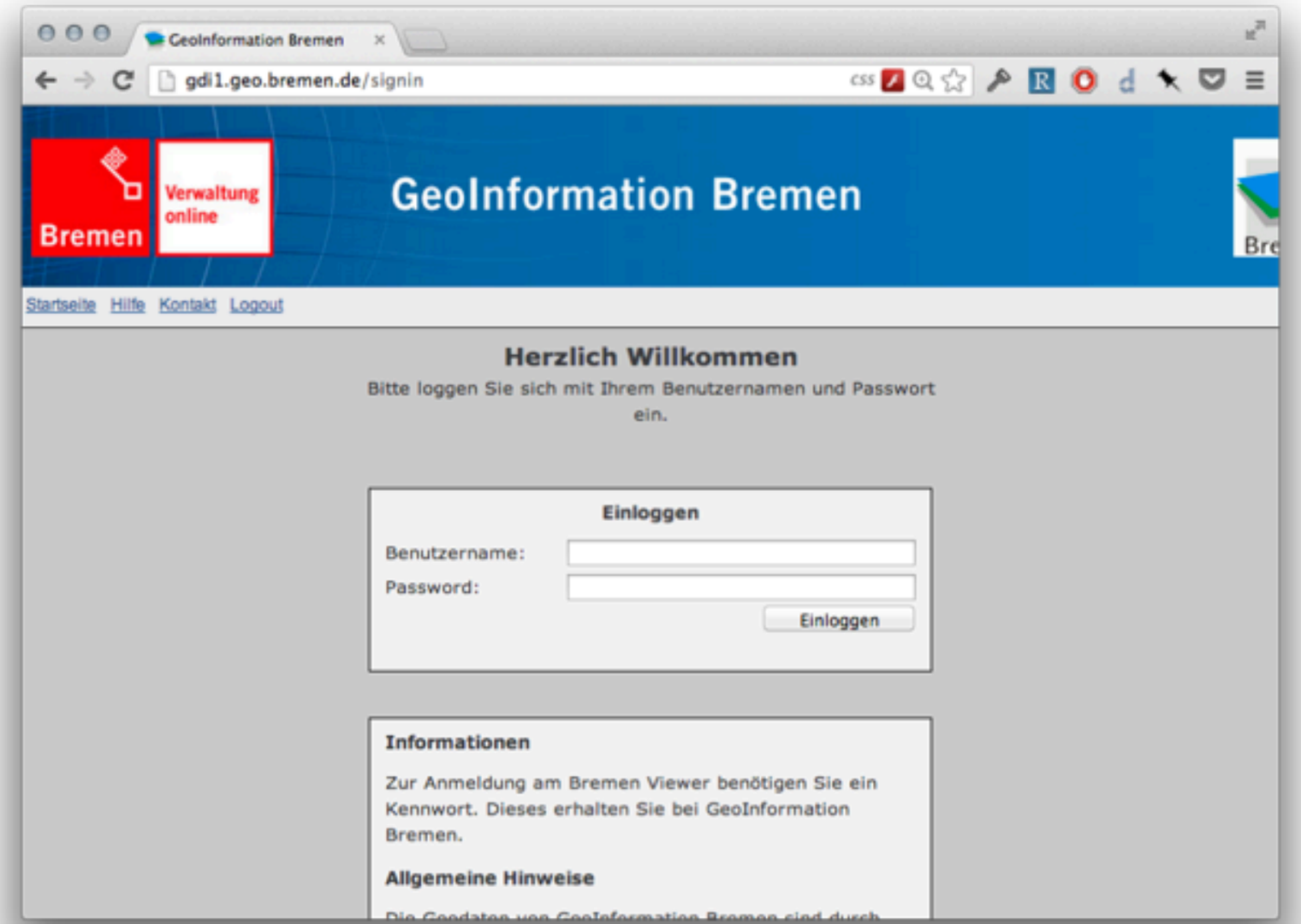
- Authentifizierung nach Anfrage IP
- Autorisierung
  - full für Authentifizierte Nutzer
  - partial ohne Layer für andere Nutzer

Leitungsauskunft

- Auskunftsanfrage über Webanwendung
- Authentifizierung über URL-Token
- Autorisierung
  - limited_to Auskunftsbereich, ggf. mit Puffer
  - Zeitliche Beschränkung

Omniscale

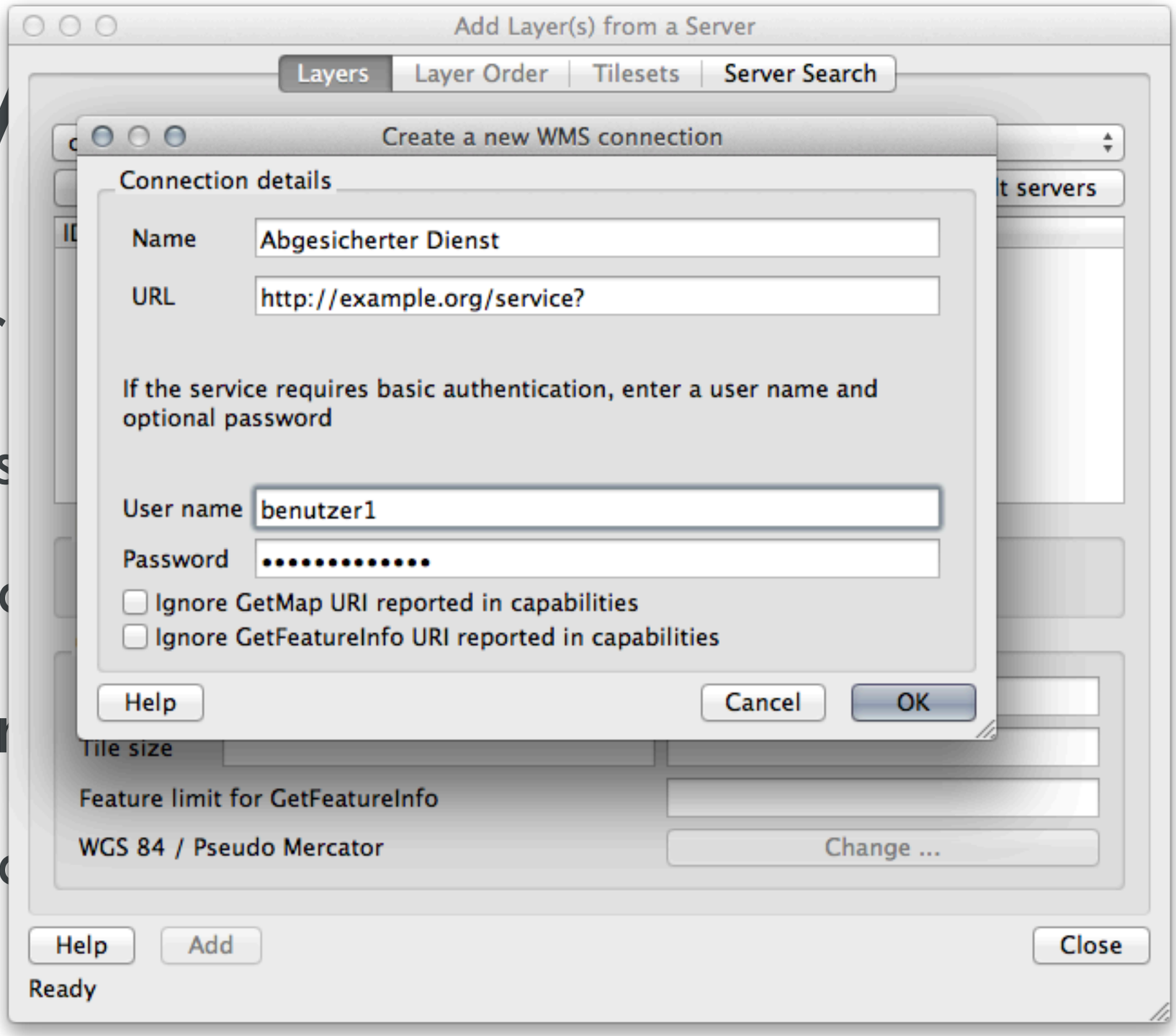
WebGIS mit Nutzergruppen



WebGIS mit Nutzergruppen

- Bestehende Webanwendung mit Benutzern und Gruppen
- Authentifizierung über Cookie der Webanwendung
- Autorisierung
  - Rechte aus Datenbank der Webanwendung

- Inter
  - Freis
  - Basi
- Exter
  - Basi

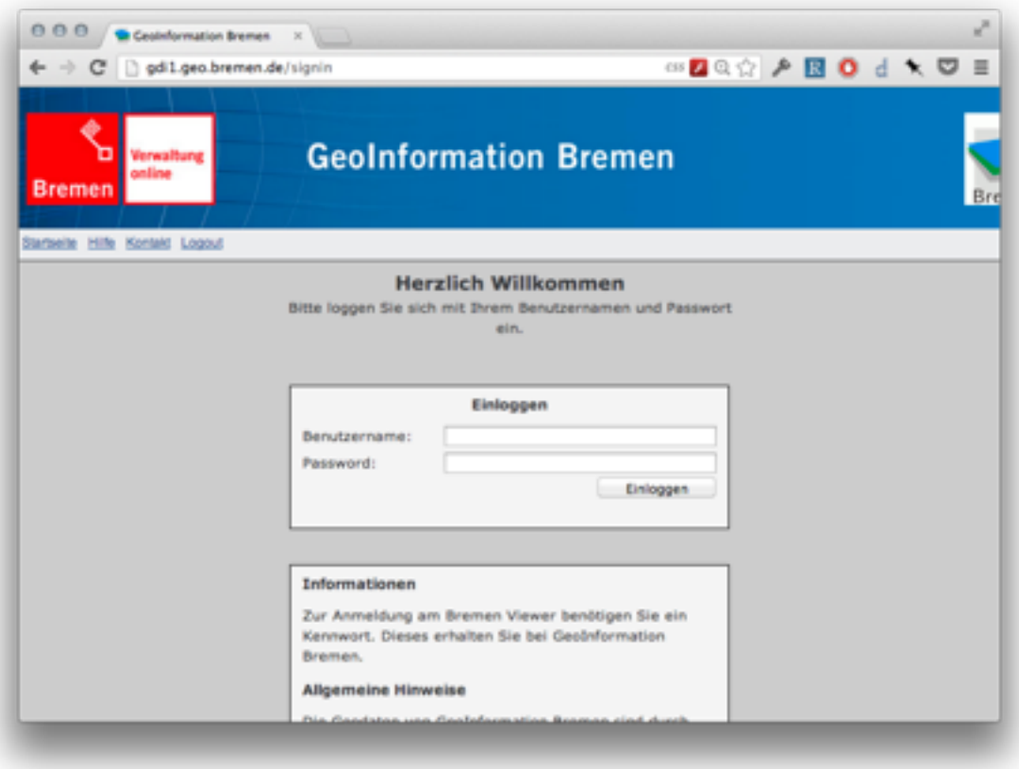


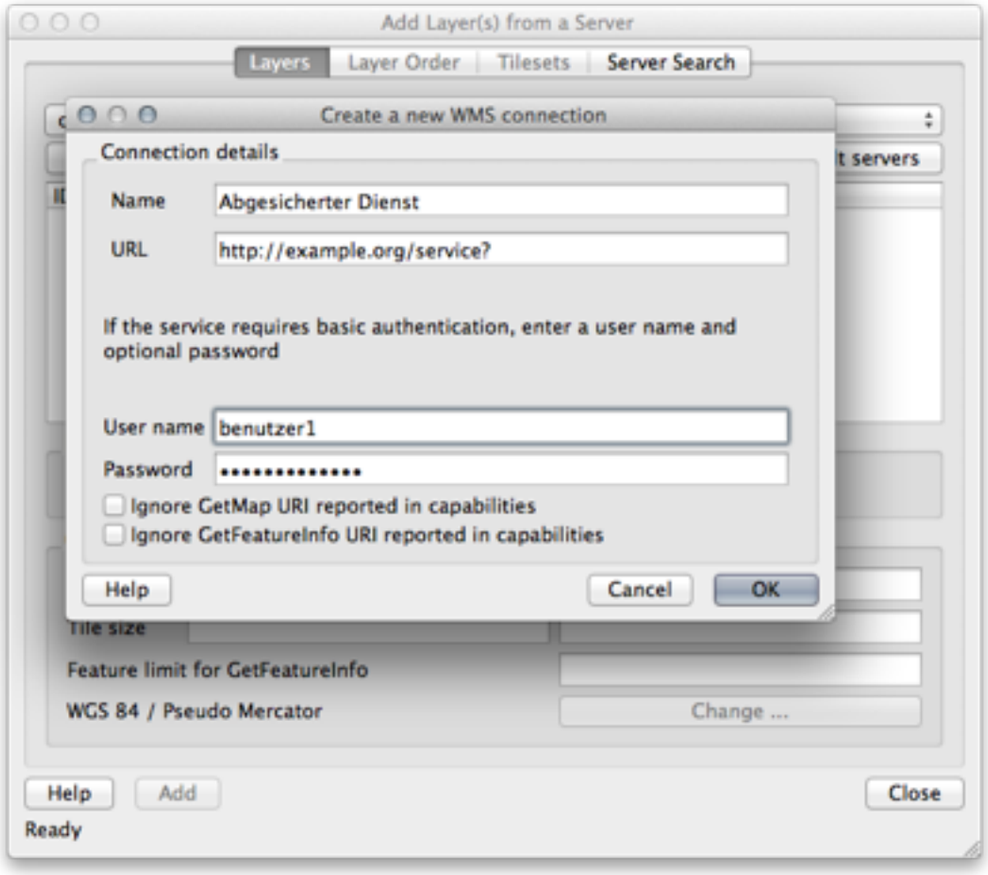
Hybrid-Authentifizierung

- Über Cookie, wenn gesetzt (Identification)
- Sonst HTTP-Basic Auth (Challenge)

Omniscale

Hybrid-Authentifizierung





Zusammenfassung

- Absicherung
  - Authentifizierung
  - Autorisierung
- WSGI Zwischenschicht
- Flexible Lösungen

Omniscale

Vielen Dank

Weiter Informationen: http://mapproxy.org

Oliver Tonnhofer
E-Mail: tonnhofer@omniscale.de