GeoHealthCheck

QoS Monitor for (OGC OWS) Geospatial Web Services

Just van den Broecke

Tom Kralidis

Hannes I. Reuter

geohealthcheck.org

Credits

This presentation has been created with
Reveal.js by Hakim El Hattab
Create beautiful interactive slide decks using HTML.

Use left/right arrow keys. ESC for slides overview

PDF Print then File|Print... (Chrome only)


ABOUT JUST
Open Geo-ICT Professional
justobjects.nl
Just Objects
Member OpenGeoGroep (NL)
www.opengeogroep.nl
OPEN GEO GROEP
Secretary Board
OSGeo.nl
OSGeo.nl
Nederlandstalige afdeling
1. Intro

Tom Kralidis 

- Founder of pycsw
- Founder of GeoHealthCheck
- OWSLib
- pygeometa
- GeoNode
- QGIS (MetaSearch)
- PyWPS
- MapServer

Hannes Isaak Reuter 

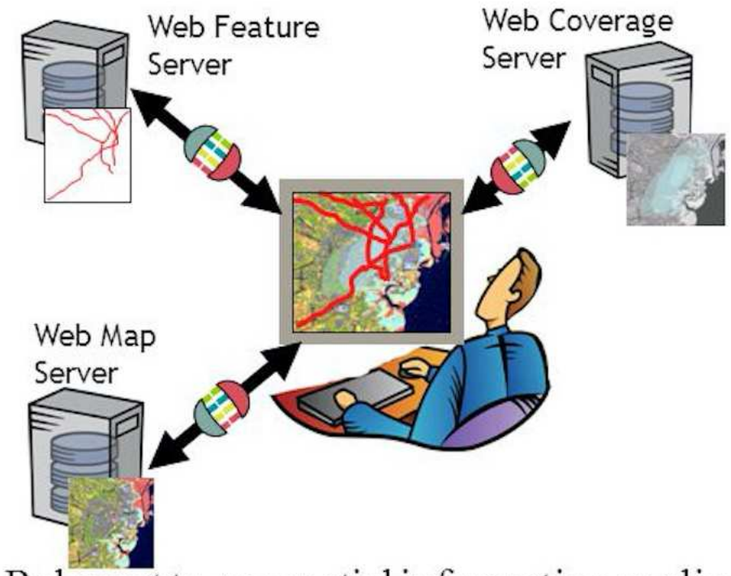
- Geoecologist (GIS, Soil) and DEM Scientist
- Contributor to GHC and other OSS
- Long Term user of ArcInfo, QGIS, GDAL, Python, pyWPS
- Working in the GISCO Team

OGC ? OWS?

- OGC: **Open Geospatial Consortium**
- OGC defines OWS standards
- OWS: **OGC Web Services**
- OWS standards include:
  - WMS, WFS, WCS, CSW, SOS, WPS, WMTS

OGC Web Services (OWS)

Just as http:// is the dial tone of the World Wide Web, and html / xml are the standard encodings, the **geospatial web** is enabled by OGC standards:



- Web Map Service (WMS)
- Web Feature Service (WFS)
- Web Coverage Service (WCS)
- Catalogue (CSW)
- Geography Markup Language (GML)
- OGC KML
- Others…

Relevant to geospatial information applications: Critical Infrastructure, Emergency Management, Weather, Climate, Homeland Security, Defense & Intelligence, Oceans Science, others

OGC

Helping the World to Communicate Geographically

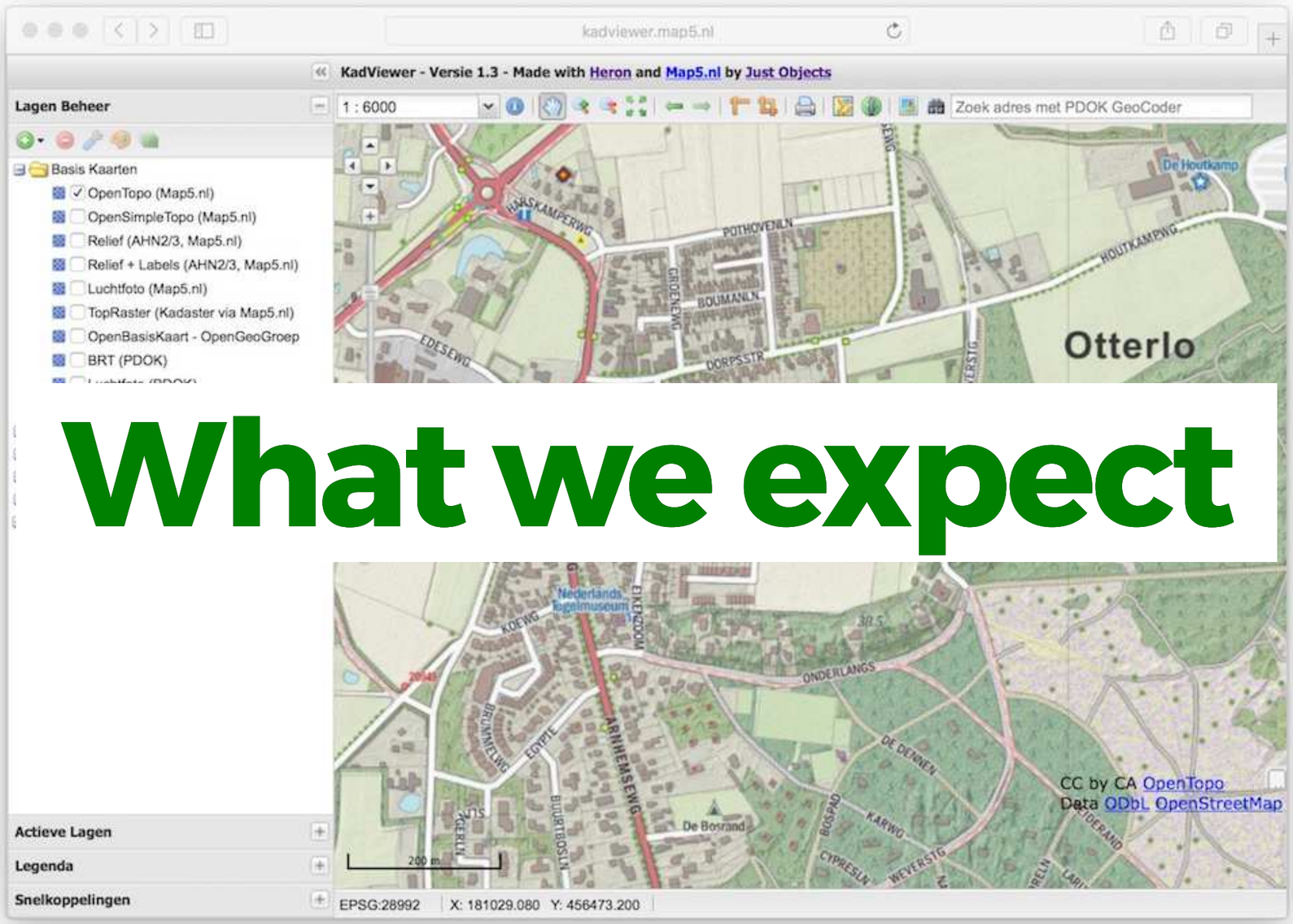
Contents

- OWS Monitoring Challenges
- GHC Walk-through
- GHC Setup
- GHC Architecture
- GHC Project

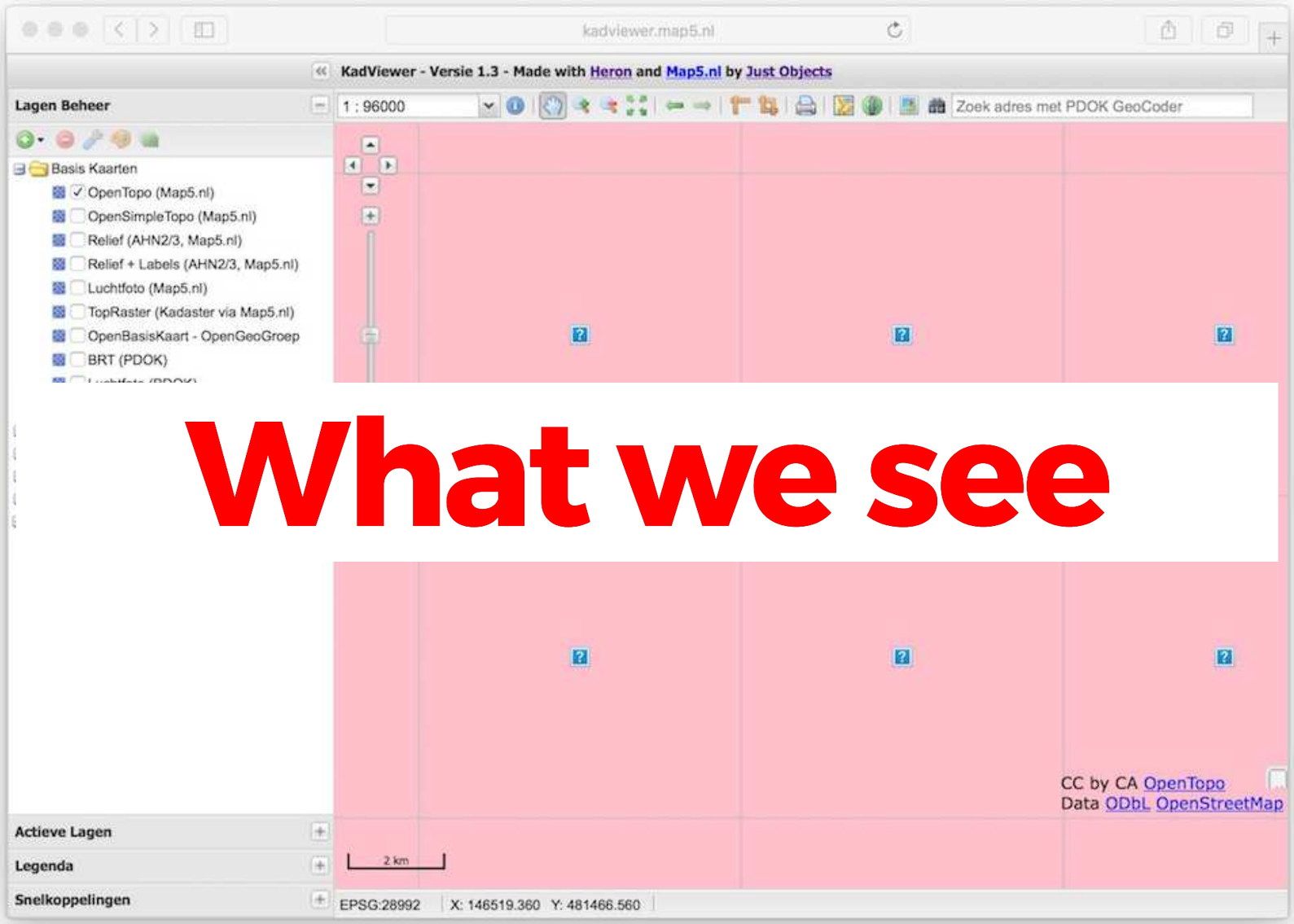
GHC=GeoHealthCheck

OWS Monitoring Challenges

“I see pink tiles!”



What we expect


kadviewer.map5.nl
KadViewer - Versie 1.3 - Made with Heron and Map5.nl by Just Objects
Lagen Beheer
1 : 96000
Zoek adres met PDOK GeoCoder
Basis Kaarten
OpenTopo (Map5.nl)
OpenSimpleTopo (Map5.nl)
Relief (AHN2/3, Map5.nl)
Relief + Labels (AHN2/3, Map5.nl)
Luchtfoto (Map5.nl)
TopRaster (Kadaster via Map5.nl)
OpenBasisKaart - OpenGeoGroep
BRT (PDOK)
CC by CA OpenTopo
Data ODbL OpenStreetMap
Actieve Lagen
Legenda
2 km
Snelkoppelingen
EPSG:28992
X: 146519.360 Y: 481466.560

What we see

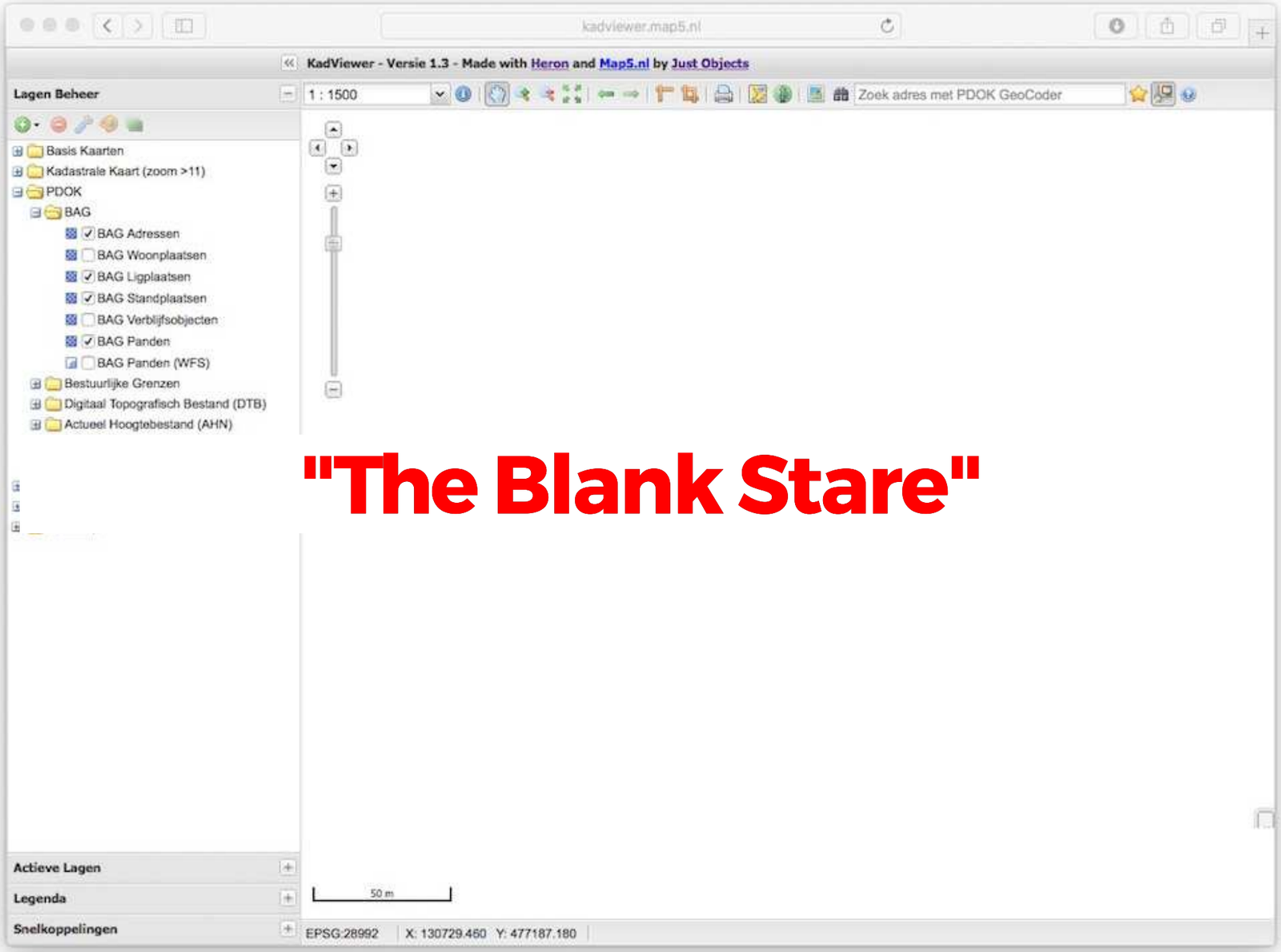
```
nl version="1.0" encoding="UTF-8" standalone="no"
OCTYPE ServiceExceptionReport
TEM "http://kademo.nl/gs2/schemas/wms/1.1.1/
_exception_1_1_1.dtd">
rviceExceptionReport version="1.1.1" >
<Se
   featureType: nlextract:pand does not have a
   properly configured datastore
</ServiceException>
erviceExceptionReport>
```

What we received

kademo.nl/gs2/nlextract/wms?LAYERS=pand&STYLES=&EXCEPTIONS=INIMAGE&FORMAT=image/png&SERVICE=WMS&VERSION

Internal error
featureType: nlextract:pand does not have a properly configured datastore

In Image Error Message


kadviewer.map5.nl
KadViewer - Versie 1.3 - Made with Heron and Map5.nl by Just Objects
Lagen Beheer
1 : 1500
Zoek adres met PDOK GeoCoder
Basis Kaarten
Kadastrale Kaart (zoom >11)
PDOK
BAG
BAG Adressen
BAG Woonplaatsen
BAG Ligplaatsen
BAG Standplaatsen
BAG Verblijfsobjecten
BAG Panden
BAG Panden (WFS)
Bestuurlijke Grenzen
Digitaal Topografisch Bestand (DTB)
Actueel Hoogtebestand (AHN)
"The Blank Stare"
Actieve Lagen
Legenda
Snelkoppelingen
50 m
EPSG:28992
X: 130729.460 Y: 477187.180

But our HTTP Monitor said:

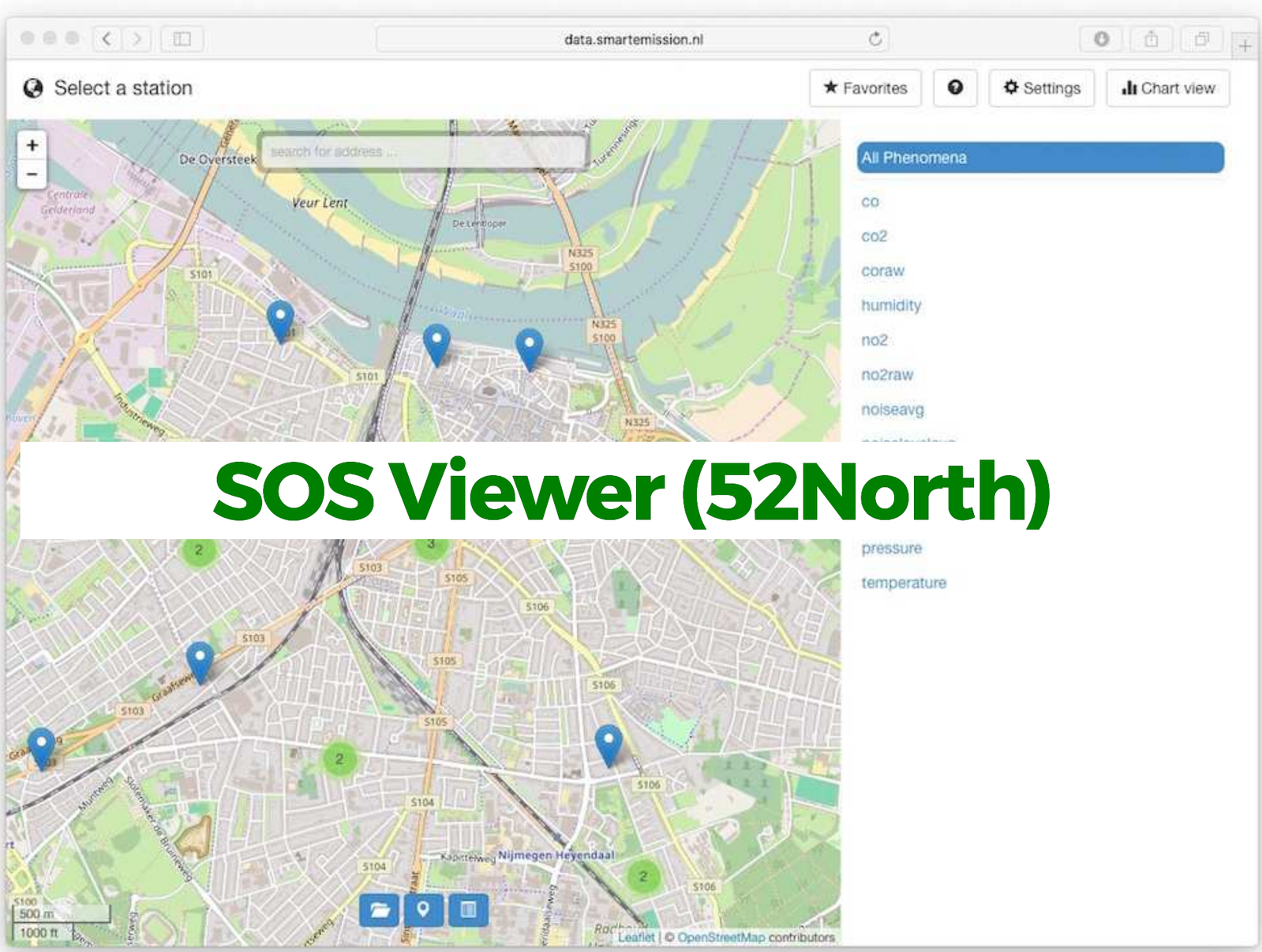
200 OK...

GetCapabilities response OK, but

- Capabilities doc may be static file
- No guarantee specific services will work:
  WMS GetMap, WFS GetFeature, ...

Time-based OGC Services

- SensorWeb Enablement (SWE)
  Internet of Things (IoT)
- Sensor Observation Service (SOS)
- SensorThings API (STA)



SOS Viewer (52North)

Gaps in SOS-Data

Public "Uptime" services

Generic HTTP checking (keywords)

But: most critical OGC-services run internally on intranets

Conclusion

Need (FOSS) OGC-Service (OWS)

QoS Checking

with History Capture

GeoHealthCheck Walk-through

demo.geohealthcheck.org

GeoHealthCheck Setup

GHC Parts

- Python Webapp (Dashboard)
- HealthCheck Runner (Plugins!)
- Database

Python Webapp

- WSGI - standard Python
- Flask - web framework
- Run Standalone, or
- In http-server e.g Nginx or Apache2

HealthCheck Runner

- Via cron-job
- Frequency and history retain configured
- Result reports
- Email Notification trigger (optional)

Plugins - Probes and Checks

- Standard (included in GHC)
- Custom (include your own!)
- Configurable via Web UI
- More later on...

Database

- Entities:

  Users, Resources, Runs,

  Tags, ProbeVars, CheckVars

- Maintains history (Runs)

- Multiple backends

  via SQLAlchemy: default SQLite

Tags

- For grouping Resources
- Provide in UI
- More later on...

Installation

- Standard Python setup - Instructions
- Paver for setup and management tasks
- Alembic with Flask-Migrate
  for DB upgrades

Use Docker!

- Versioned GHC Images on DockerHub
- Docker Compose support:
  complete stack: Webapp, DB and Jobs
- Documentation

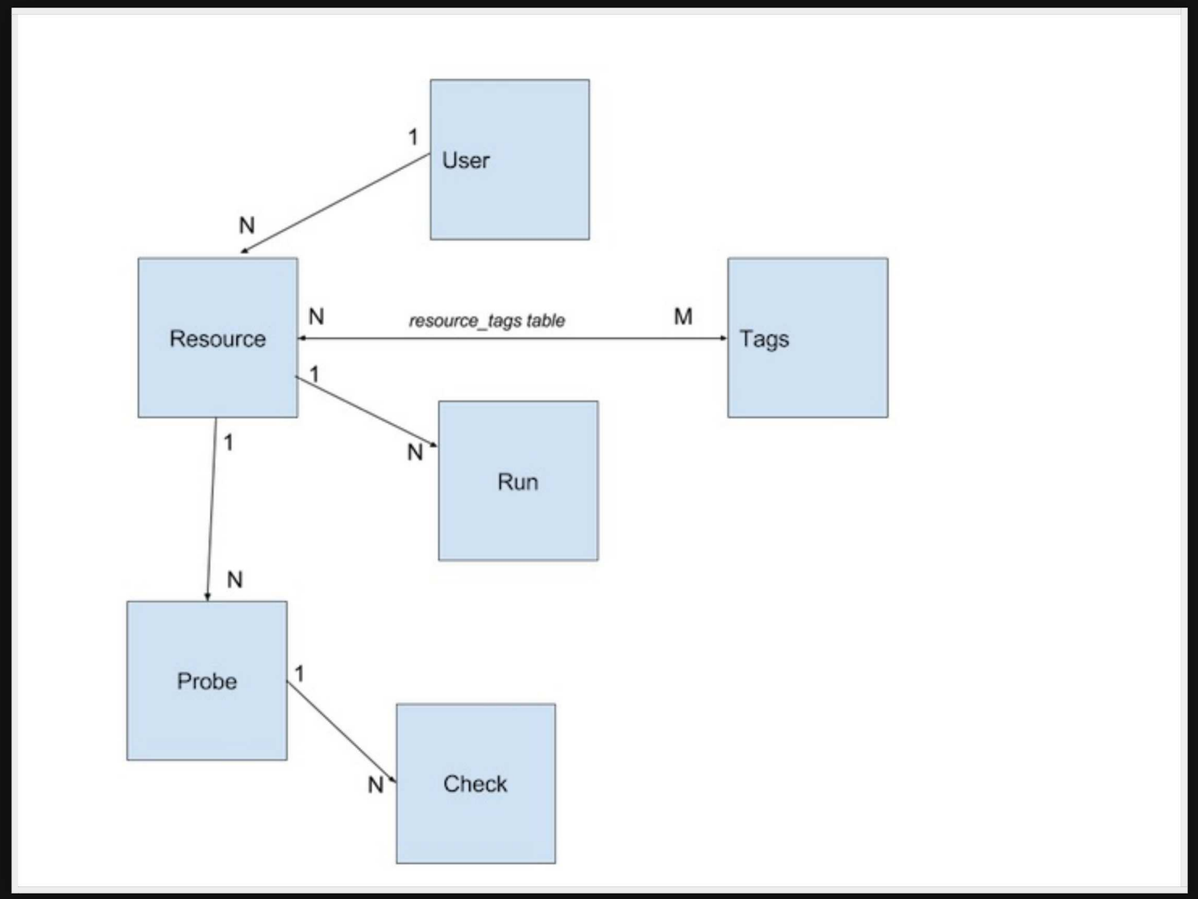
GHC Up & Running in minutes!!

Settings

```
GHC_RETENTION_DAYS = 30
GHC_RUN_FREQUENCY = 'hourly'
GHC_SELF_REGISTER = False
GHC_NOTIFICATIONS = False
GHC_NOTIFICATIONS_VERBOSITY = True
GHC_ADMIN_EMAIL = 'you@example.com'
GHC_NOTIFICATIONS_EMAIL = ['you2@example.com', ..]
GHC_SITE_TITLE = 'GeoHealthCheck Demonstration'
GHC_SITE_URL = 'http://host'
GHC_SMTP = (email settings)
GHC_RELIABILITY_MATRIX = (when to show green/orange/red)
GHC_MAP = (map setup)
GHC_PLUGINS = (Built-in Probes and Checks)
GHC_USER_PLUGINS = (Probes and Checks, YOURS!)
GHC_PROBE_DEFAULTS = (Default Probe per Resource Type)
```

GeoHealthCheck Architecture

Data Model


User
1
N
Resource
N
resource_tags table
M
Tags
1
1
N
Run
N
Probe
1
N
Check

HealthCheck Model

- Resource has URL
- URL is usually OWS Endpoint
- Probes: fire request(s) on URL
- Resource has N Probes

HealthCheck Model

- Probe has N Checks (checklist)
- Each Check checks Probe result aspect
- Check gives aspect verdict (success/fail)
- All Checks: Probe Run Report (JSON)

Plugin Model

- Probes and Checks are Plugins
- Plugin class and/or modules in config
  * Built-in Plugins: GHC_PLUGINS=
  * Your Plugins: GHC_USER_PLUGINS=
- Must be in $PYTHONPATH

Plugin Model - Probe Types

- Template (OWS) Requests
- Free-form: Probe Anything!

Time for some code!

See also Plugin Docs

Simplest Probe Class

- an HTTP GET on a *Resource* URL
- checks if the HTTP Response is not errored, i.e. a 404 or 500 status
- optionally checks if the HTTP Response (not) contains expected strings

Below is the implementation of the class `GeoHealthCheck.plugins.probe.http.HttpGet`:

```
1  from GeoHealthCheck.probe import Probe
2
3  class HttpGet(Probe):
4      """
5      Do HTTP GET Request, to poll/ping any Resource bare url.
6      """
7
8      NAME = 'HTTP GET Resource URL'
9      DESCRIPTION = 'Simple HTTP GET on Resource URL'
10     RESOURCE_TYPE = '*:*'
11     REQUEST_METHOD = 'GET'
12
13     CHECKS_AVAIL = {
14         'GeoHealthCheck.plugins.check.checks.HttpStatusNoError': {
15             'default': True
16         },
17         'GeoHealthCheck.plugins.check.checks.ContainsStrings': {},
18         'GeoHealthCheck.plugins.check.checks.NotContainsStrings': {},
19     }
       """Checks avail"""
```

Check Class HTTP Status

Next look at the Checks, the class `GeoHealthCheck.plugins.check.checks.HttpStatusNoError`:

```
1  import sys
2  from owslib.etree import etree
3  from GeoHealthCheck.plugin import Plugin
4  from GeoHealthCheck.check import Check
5
6  """ Contains basic Check classes for a Probe object."""
7
8  class HttpStatusNoError(Check):
9      """
10     Checks if HTTP status code is not in the 400- or 500-range.
11     """
12
13     NAME = 'HTTP status should not be errored'
14     DESCRIPTION = 'Response should not contain a HTTP 400 or 500 range Error'
15
16     def __init__(self):
17         Check.__init__(self)
18
19     def perform(self):
20         """Default check: Resource should at least give no error"""
21         status = self.probe.response.status_code
22         overall_status = status / 100
23         if overall_status in [4, 5]:
24             self.set_result(False, 'HTTP Error status=%d' % status)
25
```

Base GetCapabilities Probe

```python
class OwsGetCaps(Probe):
    """
    Fetch OWS capabilities doc
    """

    AUTHOR = 'GHC Team'
    NAME = 'OWS GetCapabilities'
    DESCRIPTION = 'Perform GetCapabilities Operation and check validity'
    # Abstract Base Class for OGC OWS GetCaps Probes
    # Needs specification in subclasses
    # RESOURCE_TYPE = 'OGC:ABC'

    REQUEST_METHOD = 'GET'
    REQUEST_TEMPLATE = \
        '?SERVICE={service}&VERSION={version}&REQUEST=GetCapabilities'

    PARAM_DEFS = {
        'service': {
            'type': 'string',
            'description': 'The OWS service within resource endpoint',
            'default': None,
            'required': True
        },
        'version': {
            'type': 'string',
            'description': 'The OWS service version within resource endpoint',
            'default': None,
            'required': True,
            'range': None
        }
    }
```

WMS GetCapabilities Probe

```
48  class WmsGetCaps(OwsGetCaps):
49      """Fetch WMS capabilities doc"""
50
51      NAME = 'WMS GetCapabilities'
52      RESOURCE_TYPE = 'OGC:WMS'
53
54      PARAM_DEFS = Plugin.merge(OwsGetCaps.PARAM_DEFS, {
55
56          'service': {
57              'value': 'WMS'
58          },
59          'version': {
60              'default': '1.1.1',
61              'range': ['1.1.1', '1.3.0']
62          }
63      })
64      """Param defs"""

    class WfsGetCaps(OwsGetCaps):
        """WFS GetCapabilities Probe"""
```

GeoHealthCheck Project

Open Source (MIT) on GitHub

Founded by Tom Kralidis

- Started in the air, literally!

- In flight en route to FOSS4G 2014 (YYZ -> YYC -> PDX)

A geopython Project

geopython

geopython is a GitHub organization comprised of Python projects related to geospatial.

- OWSLib
- pycsw
- PyWPS
- MetaSearch
- GeoHealthCheck
- MapSlicer
- CadTools
- Stetl

Also join us in irc://freenode.net/#geopython or the mailing list.

For more geospatial projects, check out the Toblerity Project.

Current Status (June 23, 2017)

- Second alpha release: v0.2.0
- Tags and Plugins (Probes & Checks)
- Demo demo.geohealthcheck.org
- Dev dev.geohealthcheck.org
- Docker: Images on DockerHub

Under Development

- See Issue Tracker
- Documentation

Planned

- REST API architecture
- Monitoring tools integration
  (Icinga, Munin etc)

You Can Help!

- Coding (Plugins!)
- Testing
- Documentation
- User Stories
- Sponsored Development
- Translation

Thank You!

- Website: geohealthcheck.org
- Demo: demo.geohealthcheck.org
- Development: dev.geohealthcheck.org
- Sources: code.geohealthcheck.org
- Docs: docs.geohealthcheck.org
- Presentation: geohealthcheck.org/presentation