

Spatio-Temporal Datacubes: The OGC/ISO Coverage Standards

Peter Baumann

Motivation

- Transparent WCPS query generation from python code
 - Object-oriented, Lazy evaluation, operator overloading
- Benefits
 - Reduces gap between python and server-side datacube processing (aka queries)
 - Convenience for python coders
- Practical use
 - See projects at <https://l-sis.org> & live demo

rasdaman

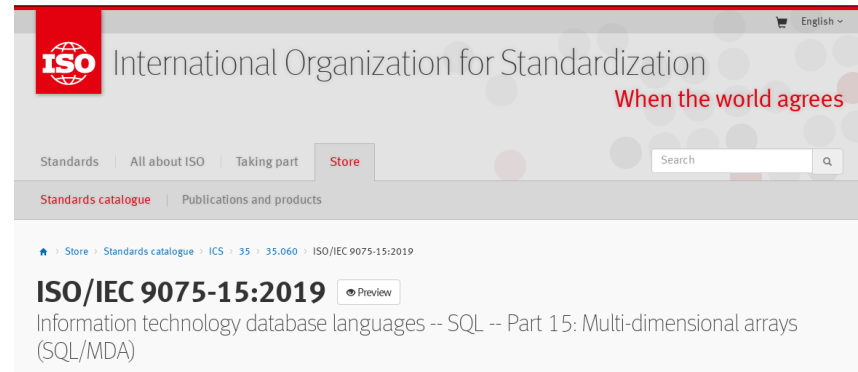
= „raster data manager“: n-D actionable datacubes

- pioneered datacubes, cf. publications & patents

■ massively scalable

Big Datacube Management & Analytics

- full-stack implementation
- Multi-PB; 1000x parallelization; federation; security; GreenCubes®



■ **Interoperable**: Reference Implementation, blueprint for standards

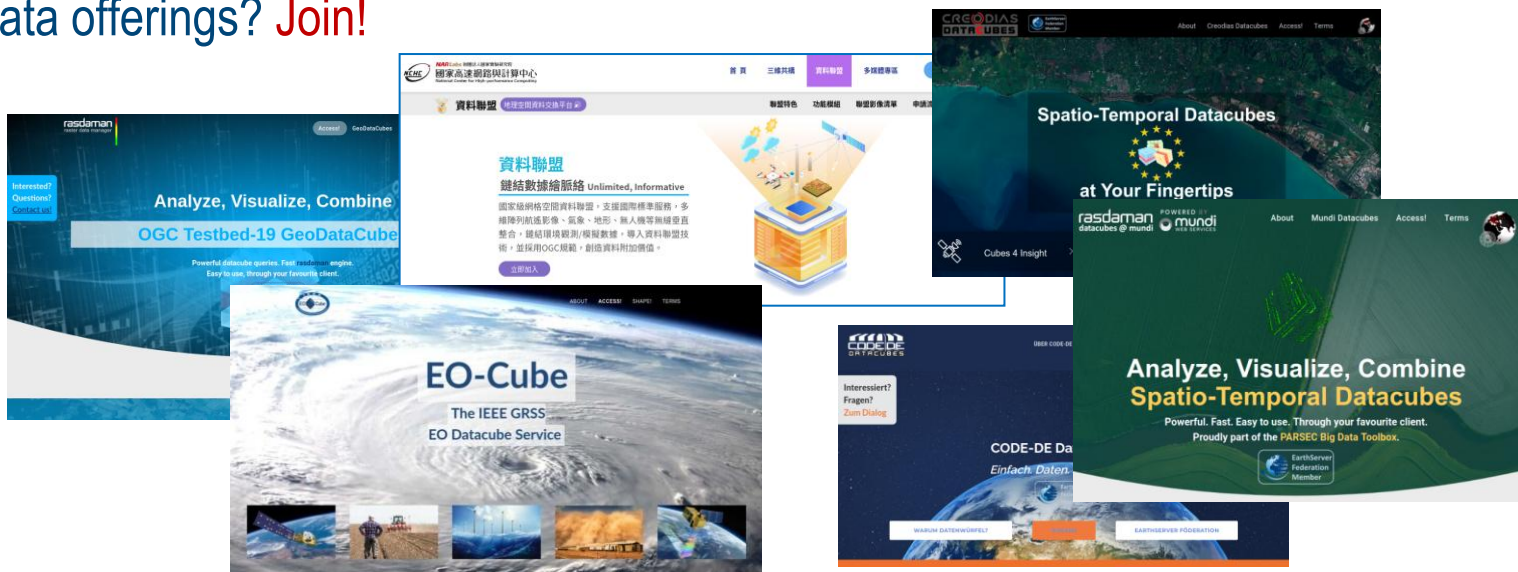




EarthServer

- datacube provider federation
 - Multi-PB location-transparent data space
 - Open standards, zero-coding

- Open, free, transparent, democratic
 - Open & private; free & commercial
 - Have data offerings? **Join!**



OGC/ISO Coverage Implementation Schema

the rasdaman team

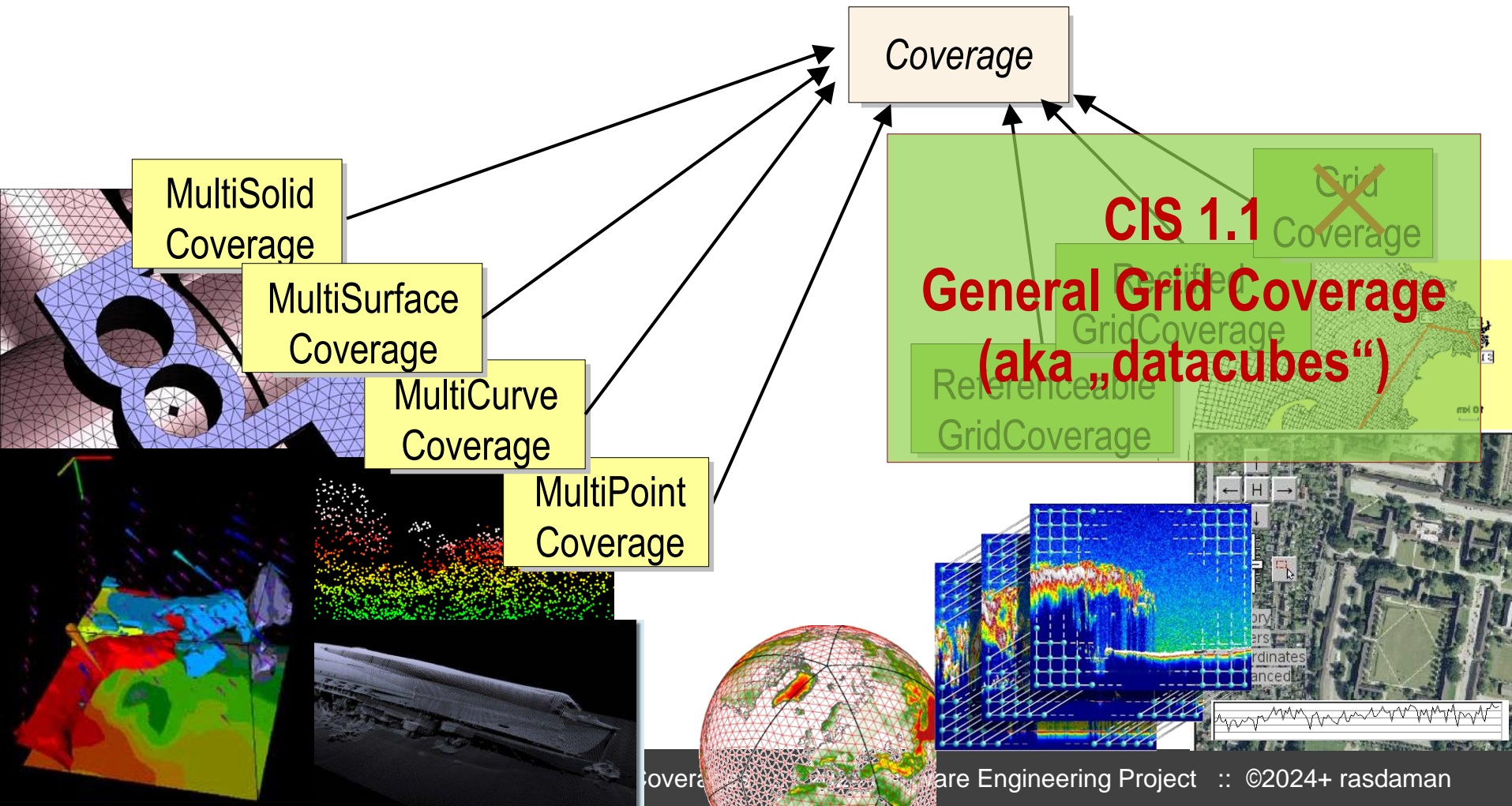
Constructor University | rasdaman GmbH

www.jacobs-university.de/isis | www.rasdaman.com

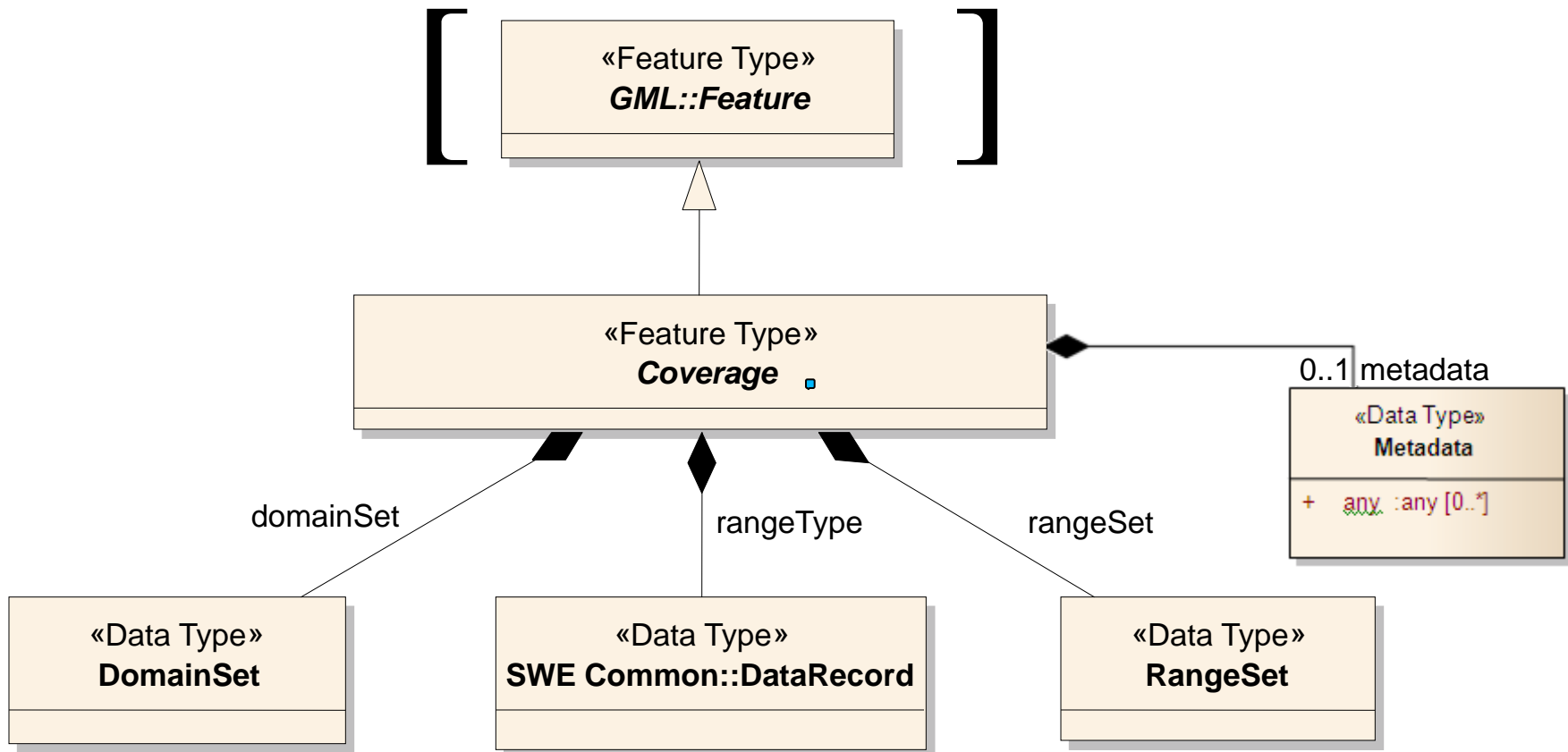


Coverages = Grids + Point Clouds + Meshes

- **abstract**: OGC Abstract Topic 6, ISO 19123-1
- **concrete**, interoperable: ISO/OGC Coverage Implementation Schema (CIS, aka GMLCOV)



Coverage Definition



A Simple Coverage, in GML

```

<generalGridCoverage ... gml:id="CIS_001">
  <domainSet>
    <generalGrid srsName="http://www.opengis.net/def/crs-compound?
      1=http://www.opengis.net/def/crs/EPSG/0/4979
      &2=http://www.opengis.net/def/crs/OGC/0/AnsiDate"
      axisLabels="Lat Long h date">
      <regularAxis axisLabel="Lat" uomLabel="deg" lowerBound="40" upperBound="60" resolution="10"/>
      <regularAxis axisLabel="Long" uomLabel="deg" lowerBound="-10" upperBound="10" resolution="10"/>
      <irregularAxis axisLabel="h" uomLabel="m">
        <c> 0</c>
        <c>100</c>
      </irregularAxis>
      <irregularAxis axisLabel="date" uomLabel="d">
        <c>2015-12-01</c>
        <c>2015-12-02</c>
      </irregularAxis>
      <gridLimits srsName="http://www.opengis.net/def/crs/OGC/0/Index4D" axisLabels="i j k l">
        <indexAxis axisLabel="i" lowerBound="0" upperBound="2"/>
        <indexAxis axisLabel="j" lowerBound="0" upperBound="2"/>
        <indexAxis axisLabel="k" lowerBound="0" upperBound="1"/>
        <indexAxis axisLabel="l" lowerBound="0" upperBound="1"/>
      </gridLimits>
    </generalGrid>
  </domainSet>

  <rangeSet>
    <dataBlock>
      <v>01</v> <v>02</v> <v>03</v> <v>04</v> <v>05</v> <v>06</v> <v>07</v> <v>08</v> <v>09</v>
      <v>01</v> <v>02</v> <v>03</v> <v>04</v> <v>05</v> <v>06</v> <v>07</v> <v>08</v> <v>09</v>
      <v>01</v> <v>02</v> <v>03</v> <v>04</v> <v>05</v> <v>06</v> <v>07</v> <v>08</v> <v>09</v>
      <v>01</v> <v>02</v> <v>03</v> <v>04</v> <v>05</v> <v>06</v> <v>07</v> <v>08</v> <v>09</v>
    </dataBlock>
  </rangeSet>

  <rangeType>
    <swe:DataRecord>
      <swe:field name="panchromatic">
        <swe:Quantity definition="http://opengis.net/def/property/OGC/0/Radiance">
          <swe:uom code="W.m-2.sr-1.nm-1"/>
        </swe:Quantity>
      </swe:field>
    </swe:DataRecord>
  </rangeType>
</generalGridCoverage>

```


A Simple Coverage, in JSON

```
{ "type": "CoverageByDomainAndRangeType",
  "domainSet":{
    "type": "DomainSetType",
    "generalGrid":{
      "type": "GeneralGridCoverageType",
      "srsName": "http://www.opengis.net/def/crs/OGC/0/Index2D",
      "axisLabels": ["i", "j"],
      "axis": [{ "type": "IndexAxisType", "axisLabel": "i", "lowerBound": 0, "upperBound": 2
        },{ "type": "IndexAxisType", "axisLabel": "j", "lowerBound": 0, "upperBound": 2 }
    ]
  }
},
  "rangeSet": { "type": "RangeSetType",
    "dataBlock": { "type": "VDataBlockType", "values": [1,2,3,4,5,6,7,8,9] } },
  "rangeType": { "type": "DataRecordType",
    "field":{ "type": "QuantityType",
      "definition": "ogcType:unsignedInt",
      "uom": { "type": "UnitReference", "code": "10^0" } }
  }
}
```

A Simple Coverage, in RDF

```

<http://www.opengis.net/cis/1.1/examples/CIS_05_2D>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
  <http://www.opengis.net/cis/1.1/CoverageByDomainAndRangeType> .

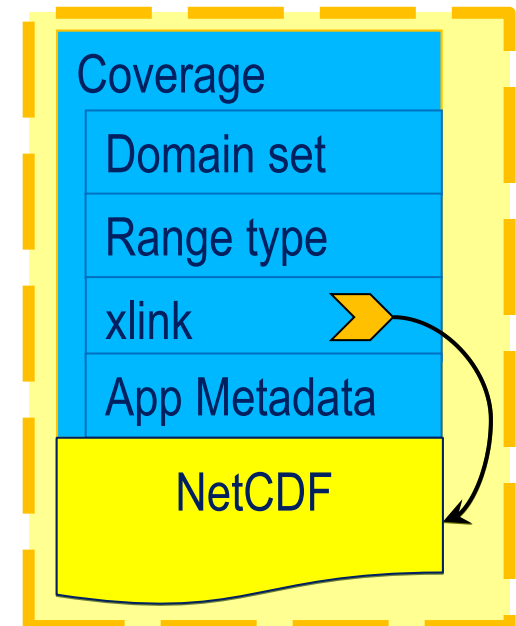
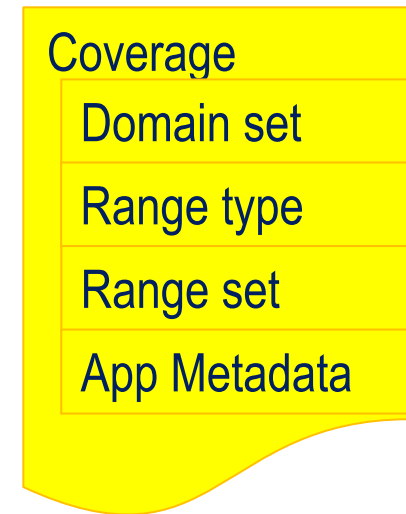
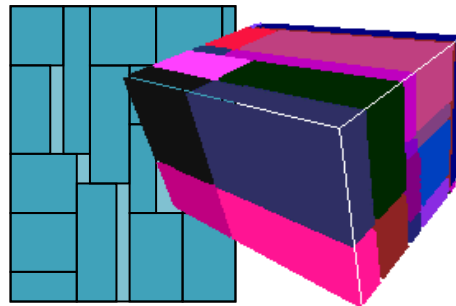
<http://www.opengis.net/cis/1.1/examples/CIS_05_2D>
  <http://www.opengis.net/cis/1.1/domainSet>
  <http://www.opengis.net/cis/1.1/examples/CIS_DS_05_2D> .
<http://www.opengis.net/cis/1.1/examples/CIS_DS_05_2D>
  <http://www.opengis.net/cis/1.1/generalGrid>
  <http://www.opengis.net/cis/1.1/examples/CIS_DS_GG_05_2D> .
<http://www.opengis.net/cis/1.1/examples/CIS_DS_05_2D>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
  <http://www.opengis.net/cis/1.1/DomainSetType> .
<http://www.opengis.net/cis/1.1/examples/CIS_DS_GG_05_2D>
  <http://www.opengis.net/cis/1.1/axis>
  <http://www.opengis.net/cis/1.1/examples/CIS_DS_GG_I_05_2D> .
<http://www.opengis.net/cis/1.1/examples/CIS_DS_GG_05_2D>
  <http://www.opengis.net/cis/1.1/axis>
  <http://www.opengis.net/cis/1.1/examples/CIS_DS_GG_J_05_2D> .
<http://www.opengis.net/cis/1.1/examples/CIS_DS_GG_05_2D>
  <http://www.opengis.net/cis/1.1/axisLabels>
  <http://www.opengis.net/cis/1.1/axisLabels0> .
<http://www.opengis.net/cis/1.1/axisLabels0> <http://www.w3.org/1999/02/22-rdf-syntax-ns#first> "i" .
<http://www.opengis.net/cis/1.1/axisLabels0> <http://www.w3.org/1999/02/22-rdf-syntax-ns#rest> <http://www.opengis.net/cis/1.1/axisLabels1> .
<http://www.opengis.net/cis/1.1/axisLabels1> <http://www.w3.org/1999/02/22-rdf-syntax-ns#first> "j" .
<http://www.opengis.net/cis/1.1/axisLabels1> <http://www.w3.org/1999/02/22-rdf-syntax-ns#rest> <http://www.w3.org/1999/02/22-rdf-syntax-ns#rest> .

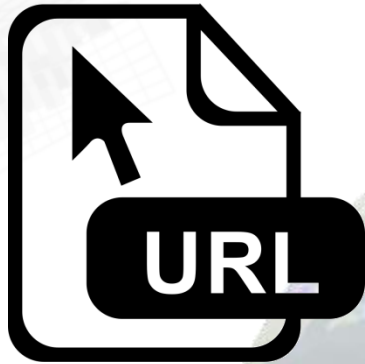
```

Encoding Coverages

- **Single file** encoding:
 - Informationally complete: GML, JSON, RDF, ...
 - *Caveat: GeoJSON, CovJSON does **not** work*
 - Further formats: GeoTIFF, NetCDF, JPEG2000, GRIB, ...

- **Multipart**: container(“header” + file1 + file2 + ...)
 - Multipart/MIME, zip, GMLJP2, SAFE, GeoPackage, ...
 - Built-in collections / tiling





<https://ows.rasdaman.org/rasdaman/ows>

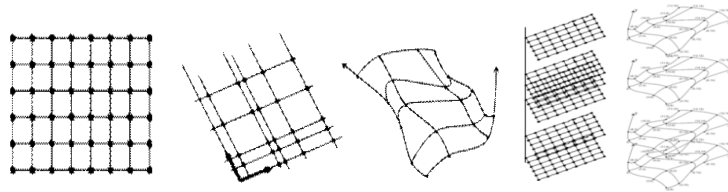


**BlueMarbleCov,
AvgTemperatureColorScaled**

Coverage Data Model: Summary

datacubes

- **Coverage** = regular & irregular **grids**, point clouds, meshes
 - See <http://schemas.opengis.net/cis/1.1/> for schemas + examples



- **Coverage Implementation Schema: OGC CIS = ISO 19123-2**
 - **1.0** = RectifiedGridCoverage, ReferenceableGridCoverage
 - **1.1** = GeneraldGridCoverage
- **Caveat: Hijacking Attempts, eg: CoverageJSON – incompatible**

Web Coverage Processing Service (WCPS)

the rasdaman team

Constructor University | rasdaman GmbH

L-sis.org | rasdaman.com



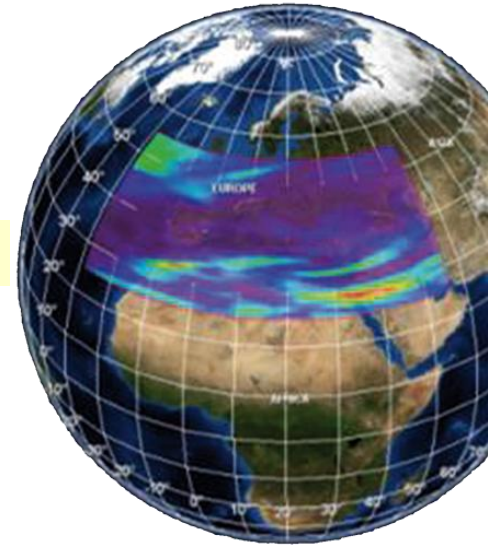
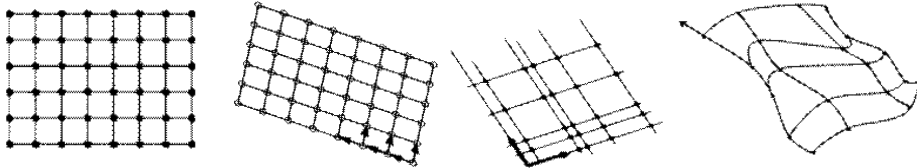
OGC WCPS: Space/Time Datacubes

- **Web Coverage Processing Service (WCPS)**

- spatio-temporal datacube analytics language

```
A[ Lat(10.2) , Long(8.4) , date("2017-12-04") ]
```

- space & time, regular & irregular grids



- "From MODIS scenes M1, M2, M3: **difference red & nir**, as TIFF"
 - "...but only those where nir exceeds 127 somewhere"

```
for $c in ( M1, M2, M3 )
where some( $c.nir > 127 )
return encode( $c.red - $c.nir, "image/tiff" )
```

WCPS in Python

```

from IPython.display import Image
import requests
query = '''
for $c in (S2_L2A_32631_B12_20m),
    $d in (S2_L2A_32631_B08_10m),
    $e in (S2_L2A_32631_B03_10m)
let $cutOut := [ ansi( "2021-04-09" ), E( 670000:730000 ), N( 4990220:5015220 ) ]
return
    encode( { red: $c[ $cutOut ]; green: $d[ $cutOut ]; blue: $e[ $cutOut ] } / 15.0,
        "image/png"
    )'''
response = requests.post( service_endpoint, data = {'query': query}, verify=False )
Image( data=response.content )

```

- More examples: https://standards.rasdaman.com/demo_jupyter.html,
<https://nbviewer.org/github/earthserver-eu/INSPIRE-notebooks/blob/master/index.ipynb>

WCPS via Command Line

- Request type = `ProcessCoverages`, so URL schema is:

```
https://ows.rasdaman.org/rasdaman/ows?
SERVICE=WCS&
VERSION=2.0.1&
REQUEST=ProcessCoverages&
query={your-query-here-escaped}
```

- Use any URL resolution tool, such as `curl`

- Ex:

```
$ curl "http://ows.rasdaman.org/rasdaman/ows" \
--out test.png \
--data-urlencode \
'service=WCS&version=2.0.1&request=ProcessCoverages&query=
for c in (mean_summer_airtemp) return encode(c, "png")'
```

Sample Sandbox Tasks

- Extract **timeslice** from AvgLandTemp datacube:
 - for \$c in (AvgLandTemp)
return encode(\$c[ansi("2014-01")], "png")
- Extract **timeline** from AvgLandTemp datacube:
 - for \$c in (AvgLandTemp)
return encode(\$c[Lat(53.08), Long(8.80)], "csv")
- **Minimum** temperature in January 2014:
 - for \$c in (AvgLandTemp)
return min(\$c[ansi("2014-01")])

Sample Sandbox Tasks

- **Pixel-wise** operations (all expected unary & binary ops available):

- for \$c in (AvgLandTemp)
return encode(\$c[ansi("2014-01")] + 15, "png")

- **Masking:**

- for \$c in (AvgLandTemp)
return encode(\$c[ansi("2014-01")] > 100, "png")

- **Case distinction:**

- for \$c in (AvgLandTemp)
return encode(switch
 case \$c[ansi("2014-01")] > 100 return { red:0; green:0; blue:255 }
 default return { red:0; green:0; blue:0 }
end),
"png")

Sample Sandbox Tasks

- Mean summer temperature inside **polygon**:

```
- for $c in (mean_summer_airtemp)
  return
    encode(
      clip( $c,
        POLYGON( ( -12.3829 132.0117, -33.4314 120.4102, ... ,
          -36.3151 143.7891 ) )
      ),
      "image/png",
      "{\"nodata\": [0] }"
    )
```

- [WKT syntax documentation](#)

Sample Sandbox Tasks

- Query too big? Structure it with **variables**:

```
- for $c in (mean_summer_airtemp)
  let $polygon := POLYGON( ( -12.3829 132.0117, -33.4314 120.4102, ... ,
                             -36.3151 143.7891 ) )
return
  encode( clip( $c, $polygon ), "image/png" )
```

- Any number of variables

Wrap-Up

the rasdaman team

Constructor University | rasdaman GmbH

L-sis.org | rasdaman.com



„Ship Code to Data“ -- *What Code to Ship?*

- ISO SQL & OGC WCPS are **safe in evaluation**
- programming languages (like python) are **not**



COMMON SENSE

Just because you can, doesn't mean you should.

Relevant Links

- Earth Datacube Playground: <https://standards.rasdaman.com>
 - Several 1D..4D datacubes
- Endpoints:
 - Jupyter notebook: https://standards.rasdaman.com/demo_jupyter-python.html
 - WCPS: https://standards.rasdaman.com/demo_wcps.html
- Intro:
 - <https://earthserver.eu/wcs>
 - <https://ai-cu.be/chatcube>