Spice Me Up: The Extended OGC Coverage Implementation Schema

Peter Baumann
Jacobs University | rasdaman GmbH
Features & Coverages

- The basis of all: geographic feature
- Special kind of feature: coverage
  - aka space-time varying phenomenon
  - regular & irregular grids, point clouds, meshes
- Usually, coverages are Big Geo Data
Coverages in INSPIRE

- **All raster data are coverages:**
  Orthoimagery, elevation, thematic raster layers, weather, ...
  - More generally: sensor, image [timeseries], simulation, statistics data
  - Can have space and/or time coordinates

- **Used in many INSPIRE themes:** AC-MF, OF, ER, EL, NZ, OI, LU, LC, GE, SO
  - Data shall be provided using (mainly) the raster data spatial representation type – coverages
  - INSPIRE: „WCS is the natural way to provide coverages“
Big Datacube Standards

- **Open Geospatial Consortium (OGC):**
  - WCS „Big Geo Data“ standards suite
  - rasdaman Reference Implementation

- **ISO:**
  - TC211: coverages & WCS
    - ISO 19123 → 19123-1
    - OGC CIS → 19123-2
    - OGC WCS → ISO WCS
  - SC32: SQL/MDA („Multi-Dimensional Arrays“)

- **INSPIRE:**
  - coverages & WCS
  - rasdaman Reference Implementation
Coverage Types

- abstract: OGC Abstract Topic 6 = ISO 19123
- concrete, interoperable: Coverage Implementation Schema (CIS) \(\rightarrow\) ISO 19123-2
  - aka GMLCOV
  - CIS = CIS 1.0 + CIS 1.1

- MultiSolid Coverage
- MultiSurface Coverage
- MultiCurve Coverage
- MultiPoint Coverage

«FeatureType» Abstract Coverage

Rectified Grid Coverage
General Grid Coverage
Referenceable Grid Coverage

“datacubes”
Coverage Definition

Coverage Implementation Schema

«Feature Type»
GML::Feature

«Feature Type»
Coverage

«Data Type»
DomainSet

«Data Type»
RangeSet

rangeType

rangeSet

domainSet

0..1 metadata

«Data Type»
Metadata

+ any : any [0..*]

[OGC 09-146r6]
<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
      &amp;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>
    </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

```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
      },{]
    }
  },
  "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
Encoding Coverages

- **Single file encoding:**
  - Informationally complete: GML, JSON, RDF, …
  - Further formats: GeoTIFF, NetCDF, JPEG2000, GRIB, …

- **Multipart: container( “header” + file1 + file2 + … )**
  - Multipart/MIME, zip, GMLJP2, SAFE, GeoPackage, …
  - Collections, tiling, streaming
**Classical Approach vs Datacubes**

**WaterML 2.0: time slices**
- Good performance in x/y
- Disastrous performance on timeseries analysis
- same with TimeseriesML

**OGC Coverages:**
Implementation can choose **efficient layout**
- Same performance in all directions, space & time
OGC Web Coverage Service (WCS)

- **WCS Core**: access to spatio-temporal coverages & subsets
  - Encoding on the fly
  - subset = trim | slice

- **WCS Extensions**: optional functionality facets
  - from extraction up to flexible analytics

- **WCS Application Profiles**: domain-oriented bundling

Large, growing implementation basis: rasdaman, GDAL, QGIS, OpenLayers, OPeNDAP, MapServer, GeoServer, GMU, NASA WorldWind, EOx-Server; Pyxis, ERDAS, ArcGIS, ...
**WCS Core GetCoverage**

- Download a coverage (or a subset thereof), values **guaranteed unchanged**

- **Ex: „download coverage c001“**
  
  ![Image](http://www.acme.com/wcs?SERVICE=WCS&VERSION=2.0&REQUEST=GetCoverage&COVERAGEID=c001)

- **Ex: „coverage c001, lat/long cutout, time slice t=2009-11-06T23:20:52“**
  
  ![Image](http://www.acme.com/wcs?SERVICE=WCS&VERSION=2.0&REQUEST=GetCoverage&COVERAGEID=c001&SUBSET=Long(100,120)&SUBSET=Lat(50,60)&SUBSET=time("2009-11-06T23:20:52")

- **Ex: „coverage c001, in GeoTIFF“**
  
  ![Image](http://www.acme.com/wcs?SERVICE=WCS&VERSION=2.0&REQUEST=GetCoverage&COVERAGEID=c001&FORMAT="image/tiff")
WCS Transaction Extension [OGC 13-057]

- **WCS-T**: InsertCoverage + DeleteCoverage + UpdateCoverage

- **Ex:**
  
  ```
  http://www.acme.com/wcs
  ? SERVICE=WCS & VERSION = 2.0
  & REQUEST=InsertCoverage
  & COVERAGEREF=http://bcme.com/archive/hurricane.nc
  & USEID=new
  ```

- In rasdaman, enhanced with JSON recipe & ingredients layer
  - Data sources, defaults, missing data, continuous ingest, ...

```json
{
    "config": {
        "service_url": "http://localhost:8080/rasdaman/ows",
        "tmp_directory": "/tmp/",
        "crs_resolver": "http://localhost:8080/def/",
        "default_crs": "http://localhost:8080/def/crs/OGC/0/Index2D",
        "mock": false,
        "automated": false,
        "track_files": false
    },
    "input": {
        "coverage_id": "Chlorophyll",
        "paths": [
            "SCALE*.TIFF"
        ],
    },
    "recipe": {
        "name": "time_series_irregular",
        "options": {
            "time_parameter": {
                "filename": {
                    "regex": "\(.\)\.(\.*)\.(\.*)",
                    "group": "2"
                },
                "datetime_format": "YYYY-MM"
            },
            "time_crs": "http://localhost:8080/def/crs/OGC/0/ AnsiDate",
            "tiling": "ALIGNED [0:3600, 0:1800, 0:0] TILE SIZE 19443000"
        }
    }
}
```
WCPS: Datacube Analytics

- Web Coverage Processing Service
  = spatio-temporal datacube analytics language

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

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

- s/t regular & irregular grids
...But That's Not What You Want to See

- Let users remain in comfort zone of well-known tools
  - Map navigation: OpenLayers, Leaflet, ...
  - Virtual globe: NASA WorldWind, Cesium, ...
  - Web GIS: MapServer, GeoServer, QGIS, ArcGIS, ...
  - Analysis: GDAL, R, python (OWSLIB, Jupyter notebooks), ...

- ...via WCS / WCPS / WMS
  as standard client/server APIs

[screenshots: diverse clients accessing rasdaman]
**WCS Testing: Down to Single Pixel Level**

**TEAM Engine**
(Test, Evaluation, And Measurement Engine)

Select test suite:

- **Organization**
  - OGC

**Select Profile**

- **rasdam 9.0**
  - WCS 2.0 Interface Standard- Core: Corrigendum 2.0.1
  - Official OGC® Reference implementation | Live Test | License | Source Code

- **rasdam 9.0**
  - Web Coverage Service 2.0 Interface Standard - XML/POST Protocol Binding Extension 1.0.0
  - Official OGC® Reference implementation | Live Test | License | Source Code

- **rasdam 9.0**
  - Web Coverage Service Interface Standard - Interpolation Extension 1.0
  - Official OGC® Reference implementation | Live Test | License | Source Code

- **rasdam 9.0**
  - Web Coverage Service Interface Standard - Scaling Extension 1.0
  - Official OGC® Reference implementation | Live Test | License | Source Code

- **rasdam 9.0**
  - Web Coverage Service Interface Standard - Range Subsetting Extension 1.0
  - Official OGC® Reference implementation | Live Test | License | Source Code

- **rasdam 9.0**
  - Web Coverage Service WCS Interface Standard - Processing Extension 2.0
  - Official OGC® Reference implementation | Live Test | License | Source Code

Credits: Jinsongdi Yu, Stephan Meissl
rasdaman: Agile Array Analytics

= "raster data manager": SQL + n-D arrays
  - Scalable parallel “tile streaming” architecture
  - Spatio-temporal regular & irregular grids

- Mature, in operational use
  - blueprint for Big Datacube standards: ISO, OGC, INSPIRE
  - OGC & INSPIRE WCS Reference Implementation
Architecture

Web clients (m2m, browser)

Internet

rasdaman

geo services

rasserver

file system

database

external archives

optional compression

distributed query processing
No single point of failure

alternative storage

[SSTD 2013]
Parallel, Distributed Processing

select
  max((A.nir - A.red) / (A.nir + A.red))
  - max((B.nir - B.red) / (B.nir + B.red))
  - max((C.nir - C.red) / (C.nir + C.red))
  - max((D.nir - D.red) / (D.nir + D.red))
from A, B, C, D

1 query → 1,000+ cloud nodes
[ACM SIGMOD DanaC 2014]
[VLDB BOSS 2016]
EarthServer: Datacubes At Your Fingertips

- Agile Analytics on x/y/t + x/y/z/t Earth & Planetary datacubes
  - Rigorously standards: OGC WMS + WCS + WCPS
  - EU rasdaman + US NASA WorldWind
  - 700+ TB → 1+ PB

- Intercontinental initiative,
  3+3 years: EU + US + AUS

- www.earthserver.eu,
  www.planetserver.eu
MEA: Sample Datacubes Coverages
MEA: Daily Hydro Estimator
ECMWF: River Discharge
MEA: Land Surface Temperature, Cloudfree
**Server-Side Processing: Federation**

SELECT ENCODE(CASE
WHEN (CONDENSE + over i, j in [42364:42368] using
d[0:3600, 0:1800, i, j[0]] / 1423 + 1.47) > ((CONDENSE +
over i, j in [42364:42368] using (c) [**:;**, i, j[0]])*1000))
THEN ((255) + {1c, 0c, 0c, 0c} + (255) * {0c, 1c, 0c, 0c} + (255)
* {0c, 0c, 1c, 0c} + (0) * {0c, 0c, 0c, 1c})

WHEN (CONDENSE + over i, j in [42364:42368] using
d[0:3600, 0:1800, i, j[0]] / 1423 + 4) > ((CONDENSE + over
i, j in [42364:42368] using (c) [**:;**, i, j[0]])*1000))
THEN ((0) + {1c, 0c, 0c, 0c} + (128) * {0c, 1c, 0c, 0c})

**Query:**

Heavy rainfall risk areas

**Server:**

ECMWF

Run Query
German recommendations for Roadmap

EO Big Data

- Establish „European Datacube Federation“ as a strategic goal
  - Advancing user service quality, specifically: timeseries analytics
  - Enable flexible framework for value-adding services, through open standards
  - Maintain European technology lead
  - EarthServer as reference example to be enlarged

- Foster and Organize joint activities with platform developers and operators in member states
  - Analyse existing EO Exploitation Platforms with respect to common functions and interoperable interfaces
  - Define federation requirements including IT security aspects on appropriate levels
  - Support harmonization and standardization on interfaces of federated platforms
  - Develop components and workflows supporting harmonized scenarios and interfaces
Conclusion

- **Coverage Suite**: data & service model for spatio-temporal data
  - CIS + WCS
  - All raster data: regular & irregular sensor, image (timeseries), simulation, statistics data
  - from simple access (WCS Core) to datacube analytics (WCPS)
  - Conformance tests down to pixel level, interoperable

- **Convergence**: stds bodies, open-source & proprietary implementers
  - OGC, ISO, INSPIRE
  - Growing number of implementations

- **rasdaman** reference implementation
  - Proven, scalable, operational

---

visit the rasdaman booth!
Resources

- OGC authoritative standards page
  - http://www.opengeospatial.org/standards/wcs
  - http://www.opengeospatial.org/standards/wcps

- OGC Coverages.DWG wiki (background info, stds pre-releases)
  - http://external.opengeospatial.org/twiki_public/CoveragesDWG


- Training material
  - Webinars: www.earthserver.eu/webinars
  - Online interactive demos & sandbox: http://standards.rasdaman.com/
  - Wikipedia: coverages, WCS, WCPS
  - Jupyter notebooks: http://nbviewer.jupyter.org/github/earthserver-eu/

- rasdaman: www.rasdaman.org
Backup
Facing the Coverage Tsunami

sensor feeds [OGC SWE]

coverage server

simulation data
Taming the Coverage Tsunami

sensor feeds [OGC SWE]

coverage server

simulation data
Serving Coverages

- **SWE SOS:** upstream data capturing
- **CIS & WCS:** downstream access & analytics
Inset: INSPIRE -- Summary of Issues

- Recombining, mixing with new constituents → new “coverage” types
  - not semantically interoperable with OGC coverages

- Modeling of interleaved data inadequate on conceptual, rather than encoding level
  - different classes → impact on other, unrelated capabilities of the data type chosen
  - addresses only very specific case (time), not general interleaving (any axis, subsets)
  - pattern may not solve streaming

- timeseries handling is unnecessarily complicated

- unclear: multi-dimensional CRSs; bindings to non-GML data formats; arbitrary user-defined metadata
Inset: INSPIRE Time Handling

- OGC Coverages: time just another axis
- INSPIRE (WaterML): timseries = time slices
  - WaterML extended: scalars → images
Jupyter Notebooks

Planetserver WCPS python notebook for the INSPIRE conference 2016

- Please, run the next cell to see a RBG combination retrieved using WCPS

To run the code below:
1. Click on the cell to select it.
2. Press SHIFT+ENTER on your keyboard.

**RGB false color combination**

```python
import urllib2
import matplotlib.pyplot as plt

# create a file-like object from the url
f = urllib2.urlopen('http://access.planetserver.eu:8080/inspire/ows?service=WCB&version=2.0.1&request=ProcessCoverages&query=for%20data%20of%20the%20region
data

# read the image file in a numpy array
a = plt.imread(f)
plt.imshow(a)
plt.show()
```
Related Work: Hadoop – *one size does not fit all*

- “Since it was not originally designed to leverage the structure its performance is suboptimal” [Daniel Abadi]
- U Madison / GMU benchmark confirms [AGU 2015]

**COMMON SENSE**

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

Managing Complexity

- Separate conformance classes for core, gridded and discrete data, partitioning, encoding
Synopsis of Coverage Service Suite

- **WCS** -- spatio-temporal coverage access
- **WCPS** -- spatio-temporal geo analytics language
- **WPS** -- predefined server functionality
- **WCS-T** -- ingest & update
Better Than DescribeCoverage: OGC WQS

- Web Query Service [OGC 14-121]
  - Discussion Paper, adopted May 2016

- Can query any hierarchically organized server offering
  - XML, JSON, ...

- Query = XPath expression

- Ex:
  - http://www.acme.com/ows?
    SERVICE=WQS & VERSION=1.0 & REQUEST=Query &
    QUERY=/Capabilities/Contents/CoverageSummary/CoverageId &
    FORMAT=application/gml+xml

  - QUERY=/Capabilities/formatSupported

  - QUERY=/OfferedCoverage/RectifiedGridCoverage/domainSet/Grid/[@dimension="3"]
WCS Range Subsetting Extension [12-040]

- Select components from compound cells
  - "channel", "band" subsetting

- Ex:
  ...& RANGESUBSET=red &...
  ...& RANGESUBSET=nir,red,green &...
  ...& RANGESUBSET=green,red,blue &...
  ...& RANGESUBSET=nir:green &...
  ...& RANGESUBSET=band01,band03:band05,band19:band21 &...
OGC Web Processing Service (WPS)

- WCPS: semantics in language, ready → semantic interoperability (m2m!)
  
  ```
  for $c$ in (M1, M2, M3)
  return encode abs($c$\_red - $c$\_nir), "netcdf"
  ```

- WPS: semantics in human-readable text → syntactic interoperability

```xml
<ProcessDescriptions ...>
  <ProcessDescription processVersion="2" storeSupported="true" statusSupported="false">
    <ows:Identifier>Buffer</ows:Identifier>
    <ows:Title>Create a buffer around a polygon</ows:Title>
    <ows:Abstract>Create a buffer around a single polygon. Accepts the polygon as GML and provides GML output for the buffered feature.</ows:Abstract>
    <ows:Metadata xlink:title="spatial" />
    <ows:Metadata xlink:title="geometry" />
    <ows:Metadata xlink:title="buffer" />
    <ows:Metadata xlink:title="GML" />
    <DataInputs>
      <Input>
        <ows:Identifier>InputPolygon</ows:Identifier>
        <ows:Title>Polygon to be buffered</ows:Title>
        <ows:Abstract>URI to a set of GML that describes the polygon.</ows:Abstract>
        <ComplexData defaultFormat="text/XML" defaultEncoding="base64" defaultSchema="http://foo.bar/gml/3.1.0/polygon.xsd">
          <SupportedComplexData>
```
W...whatever...S?

- OGC standards cover full range from data-intensive to processing-intensive „Big Data“ coverage services
Coverages in ISO

- Historically, ISO 19123 defines abstract coverage model
  - Identical to OGC Abstract Topic 6
  - Abstract = not interoperable
    – many diverging implementations can (and are known to) exist

- Currently (2016):
  - 19123 → 19123-1
  - OGC CIS 1.1 → 19123-2
  - Later: OGC WCS 2.1 → ISO WCS
create table LandsatScenes(
  id: integer not null, acquired: date,
  scene: row( band1: integer, ..., band7: integer ) mdarray [0:4999,0:4999]
)

select id, encode(scene.band1-scene.band2)/(scene.nband1+scene.band2), "image/tiff"
from LandsatScenes
where acquired between "1990-06-01" and "1990-06-30" and
  avg( scene.band3-scene.band4)/(scene.band3+scene.band4) > 0
Cloud Demo
Ortho Image Timeseries

[Diedrich et al 2002, using rasdaman]
Elevation

for $s$ in (SatImage), $d$ in (DEM) where $s$/metadata/@region = "Glasgow"
return encode(
    struct {
        red: (char) $s.b7[x0:x1,x0:x1],
        green: (char) $s.b5[x0:x1,x0:x1],
        blue: (char) $s.b0[x0:x1,x0:x1],
        alpha: (char) scale( $d, 20 )
    },
    "image/png"
)
WMS via WCPS

for $p$ in (OrthoPhoto),
   $wl$ in (WaterLines), $wa$ in (WaterAreas),
   $d$ in (DEM)
return
   encode( (unsigned char) (  
      $p$ * { 1, 1, 1 }  
      overlay  
      $wl$ * { 0, 128, 255 }  
      overlay  
      $wa$ * { 191, 255, 255 }  
      overlay  
      switch $d$
         case $d > 260$ return { red:255, green:0, blue:0 }  
         case $d > 262$ return { red:0, green:255, blue:0 }  
         case $d > 264$ return { red:0, green:0, blue:255 }  
         default return { red:0, green:0, blue:0 }  
      end
   ),
   "image/png" )
Application 2: BigPicture Project

- Diagnosis in the field
  - Big-Data-based determination of causes for satellite image derived and site-specific variations
  - Goal: Recommendations for targeted measures
    - fertilizer placement, application of plant protection products, choice of species to grow, etc.
  - Ground truthing: 500 farmers

- rasdaman via OGC WCS & WCPS

- Supported by German Federal Ministry of Food and Agriculture
Application 1: EOfarm Startup

- **Big Data Analytics for farmers**
  - rasdaman via OGC WCS & WCPS
  - similar framework deployed for water quality monitoring

- **Data: Landsat8, Sentinels, RapidEye**

- **Functionality:**
  - Color Composites, Band Ratios and Indices
  - Vegetation Detection
  - Canopy Greenness Estimation
  - Land Surface Temperature
  - Time series over AOI
Analysis-Ready Data

- **Before**: $10^x$ EO „flat“ scenes
- **Users** responsible for „Big Picture“

- **Now**: few space/time datacubes
- **Data providers** responsible for harmonization to „Big Picture“

[EOX] [EarthServer]