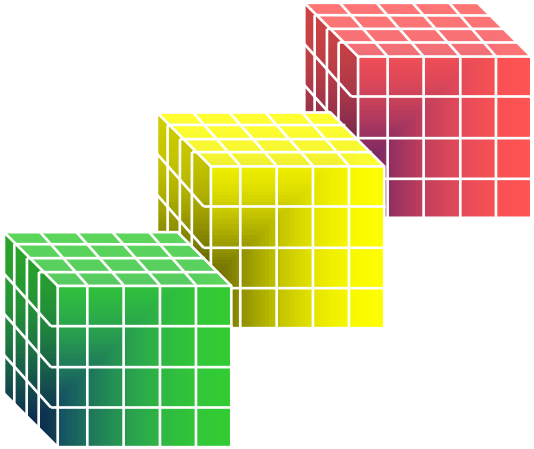
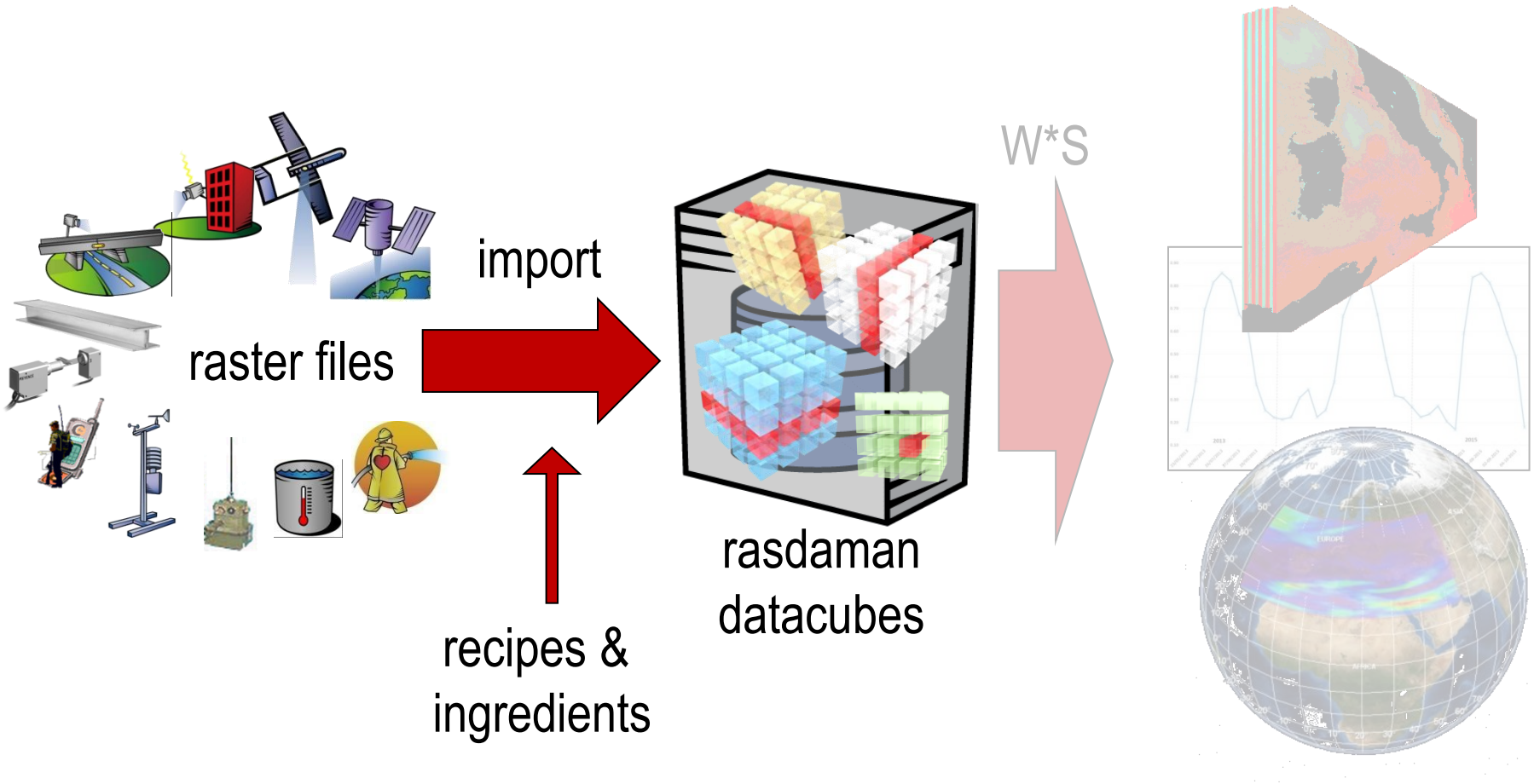


rasdaman: Geo Data Import

the rasdaman team

Jacobs University | rasdaman GmbH

Data Import: Schematic Procedure



WCST Import: The rasdaman ETL Tool

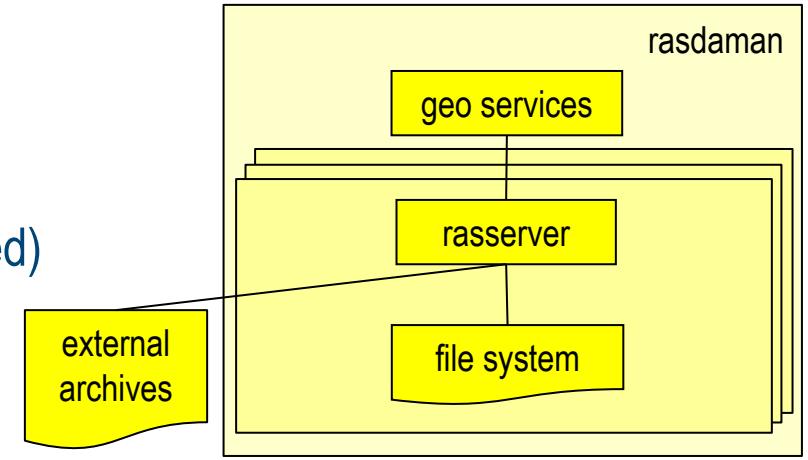
- rasdaman import tool: *WCST Import*
  - ETL: extract, transfer, load
  - Automates data import
  - Recognizes geo metadata from files
  - Organizes files into coverages based on user input
  - Maintains coverages via web requests
- 1 invocation, 1 parameter:

```
$ wcst_import.sh /path/to/myIngredient.json
```

- Config file specifies all relevant information
  - Recipes: shape of final datacube -- *normally not changed*
  - Ingredients: parameters for recipe & import process -- *customized for each cube*

Import = Ingest | Registration

- Ingestion
  - input files copied into rasdaman database
  - afterwards input files ignored (can be deleted)
  - particularly efficient storage structures
- Registration = „in-situ“
  - File metadata stored in rasdaman database (no copying)
  - Files assumed to remain available (until explicit deregistration)
  - Few restrictions (ex: cannot delete files, just unregister)
- Query speed: registered data just a little slower
  - with optimal archive organization



Recipes & Ingredients

- Goal: make data import as simple & automatic as ever possible
  - Automatic recognition
  - Where not possible: Precanned configuration
  - Where not possible: Admin configuration, only absolute minimum
- Approach: short, concise datacube configuration files
  - Recipe files: define common situations - SAR, optical, regular, irregular, ...
    - *share/rasdaman/wcst_import/*
    - *No editing required*
  - Ingredients files: parameters particular for a datacube
    - *Clone for new datacube*

Ingredients with *time_series_regular*

- stack of images → x/y/t cube
  - not necessarily contiguous
- Equal distance between time slices
- Recipe options:
  - time_start
  - time_format
  - time_crs
  - time_step

```
{
  "config": {
    "service_url": "http://localhost:8080/rasdaman/ows",
    "tmp_directory": "/tmp/",
    "default_crs": "http://localhost:8080/def/OGC/0/Index2D",
    "mock": true,
    "automated": false
  },
  "input": {
    "coverage_id": "MyCoverage",
    "paths": [
      "/var/data/*"
    ]
  },
  "recipe": {
    "name": "time_series_regular",
    "options": {
      "time_start": "2012-12-02T20:12:02",
      "time_format": "auto",
      "time_crs": "http://localhost:8080/def/crs/OGC/0/AnsiDate",
      "time_step": "2 days 10 minutes 3 seconds"
    }
  }
}
```

Ingredients with *time_series_irregular*

- stack + mosaic of images
  → x/y/t cube
  - not necessarily contiguous
- Variable distance between time slices
- Recipe options:
  - time_parameter
  - time_crs

```
{
  "config": {
    "service_url": "http://localhost:8080/rasdaman/ows",
    "tmp_directory": "/tmp/",
    "default_crs": "http://opengis.net/def/OGC/0/Index2D",
    "mock": true,
    "automated": false
  },
  "input": {
    "coverage_id": "MyCoverage",
    "paths": [
      "/var/data/*"
    ]
  },
  "recipe": {
    "name": "time_series_irregular",
    "options": {
      "time_parameter": {
        "metadata_tag": {
          "tag_name": "TIFFTAG_DATETIME"
        },
        "datetime_format": "YYYY:MM:DD HH:mm:ss"
      },
      "time_crs": "http://opengis.net/def/crs/OGC/0/AnsiDate"
    }
  }
}
```

Ingredients: Sample Time Origins

- Time in TIFF tag:

```
"recipe": {
  "name": "time_series_irregular",
  "options": {
    "time_parameter": {
      "metadata_tag": {
        "tag_name": "TIFFTAG_DATETIME"
      },
      "datetime_format": "YYYY:MM:DD HH:mm:ss"
    },
    "time_crs": "http://opengis.net/def/crs/OGC/0/AnsiDate"
  }
}
```

- Time in file name:

```
"recipe": {
  "name": "time_series_irregular",
  "options": {
    "time_parameter": {
      "filename": {
        "regex": "(.*)_(.*)_(.+?)_(.*)",
        "group": "2"
      },
      "time_crs": "http://opengis.net/def/crs/OGC/0/AnsiDate"
    }
  }
}
```

Extra Metadata

- Research on Analysis-Ready Data (ARD): need for more metadata
  - Beyond CIS 1.x schema
- Stored in reserved metadata slot, no collision with other metadata
  - Can be read, but subject to unannounced changes
  - Specific namespace xmlns:ras=„https://www.rasdaman.com“

```
<cis11:Metadata>
    <ras:covMetadata>
        <ras:axes>
            <ras:time>
                <ras:areasOfValidity>...</ras:areasOfValidity>
            </ras:time>
        </ras:axes>
        ...
    </ras:covMetadata>
    <other:anyOtherMetadata>...</other:anyOtherMetadata>
</cis11:Metadata>
```

Hook Syntax

```
"hooks": [
    {
      "description": "Remove file after import",
      "when": "after_ingestion",
      "cmd": "rm -f ${file:path}"
    }
]
```

- Concept
  - before hooks: execute action before importing data file
  - after hooks: execute action after importing data file
- Basics
  - description: human-readable description (mandatory)
  - when: when to invoke cmd
    - *one of: before_ingestion, after_ingestion (mandatory)*
  - cmd: command to be executed (mandatory)
  - replace_path: path to file imported, if different from original input file
    - *(optional, default: empty string = "no replacement")*

Import Process

- Run with:

```
$ wcst_import.sh myIngredient.json
```

  - validation: user asked for OK - confirmation starts import process
  - Optionally: import → demon process, automatic import of new files arriving

```
$ wcst_import.sh myIngredient.json --watch <interval>
```

- Sample ingredients for each recipe at:

```
/opt/rasdaman/share/rasdaman/wcst_import/ingredients
```

- Troubleshooting:
  - log file in import directory
  - https://doc.rasdaman.org/05_geo-services-guide.html#logging-and-error-handling