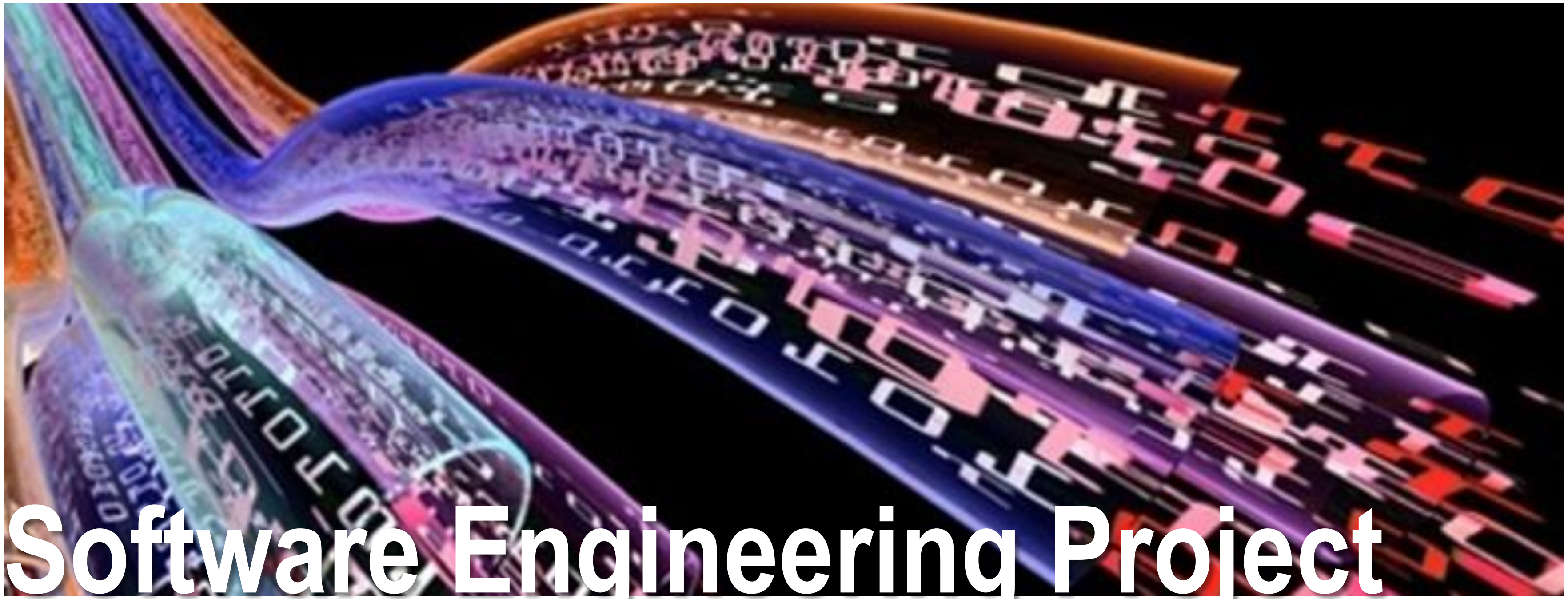


Software Engineering Project

[BAE Systems]

Instructor: Peter Baumann

email: p.baumann@jacobs-university.de

tel: -3178

office: Research 1, room 88

File not found.
Should I fake it? (Y/N)

Project Logistics

- Semester project
  - specify + implement + test + integrate + document
- 2-week code sprints:
  - teams of 2, changing teams + code base
  - advancing last-stage code base (no drop & recreate from scratch)
- grades per team
  - but reserve to deviate in exceptional cases
- Start by end of drop/add

Sprint Grading: Overview

- Automatic pull from repository shortly after submission deadline
- Quality checking; see course on quality measures, criteria include:
  - Code compiles, links, runs?
  - Code quality, with criteria such as (!) meaningful class structuring, exception handling, correct output formats, comments, proper formatting, meaningful variable & function naming (1...2-char vars!), ...
  - Amount and quality of documentation
  - Amount & quality of test cases
  - Amount of progress overall, based on (machine-readable) documentation file
  - Plagiarism check

Sprint Grading: Detail

- Core categories of evaluation:
  - Features added / improved
  - Tests added / improved
  - Documentation added /improved
  - Overall visual code appearance, i.e.: code quality
- In your README file, always describe your progress in these categories
  - To guarantee recognition of your work - if not in README, might not get found

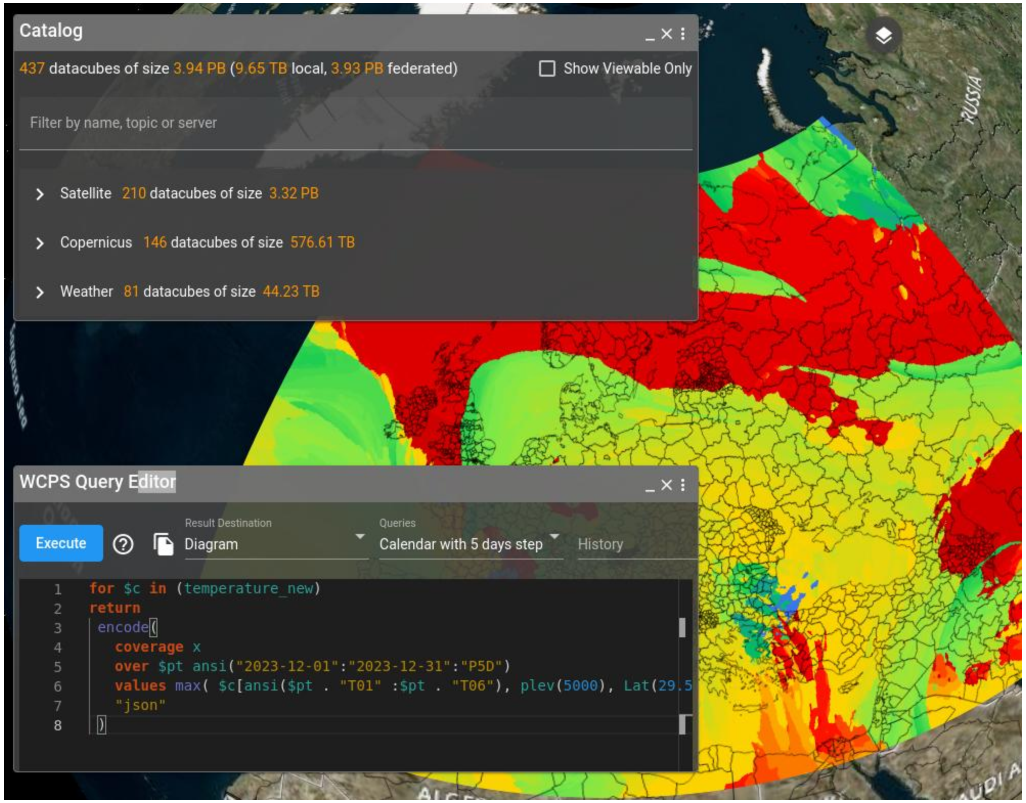
Background: Why?

- Goal: as close as possible to industry situation
- Aligned Learning Outcomes:
  - Collaboration & teamwork
    - *Organise yourself: collaborative editing, meetings, issue list, ...*
    - *Good: team develops jointly – Bad: distribute work, don‘t care about teammate*
  - Informed, planned decisions about where & what to contribute
  - Full software life cycle: → spec → code → test → handover
  - From hack-fix-repeat to engineering
    - *test code equally important*
  - How to work incrementally on preexisting code
  - Etc, see class discussion

> „The way to your goal starts the day you take over 100% responsibility for your actions.“
> – Dante Alighieri

This Year‘s Project

- Transparent WCPS query generation from python code
- WCPS (Web Coverage Processing Service)
  - Geo datacube query language
  - Standardized by Open Geospatial Consortium (OGC) & ISO
  - Used by many sites, such EarthServer federation



Catalog

437 datacubes of size 3.94 PB (9.65 TB local, 3.93 PB federated)

□ Show Viewable Only

Filter by name, topic or server

- Satellite 210 datacubes of size 3.32 PB
- Copernicus 146 datacubes of size 576.61 TB
- Weather 81 datacubes of size 44.23 TB

WCPS Query Editor

Execute

Result Destination: Diagram

Queries: Calendar with 5 days step

History

```
for $c in (temperature_new)
return
 encode(
   coverage x
   over $pt ansi("2023-12-01":"2023-12-31":"P5D")
   values max( $c[ansi($pt . "T01" :$pt . "T06"), plev(5000), Lat(29.5
   "json"
 )
```

Project: Task

- Normally: WCPS query string → datacube server → response
- From python:

```
response = requests.post(service, data = {'query': query}, verify=False)
Image(data=response.content)
```

- Some functionality available in both python and WCPS
  - Typically, better in WCPS: processing close to data, reduced response volumes
  - But not all python power available in WCPS
- Task: when operating with datacubes in python,
  silently push python ops to server via WCPS
  - Python reflection & overloading capabilities
  - Implement in python, using object-oriented approach

Project: WCPS

- Declarative query language – like SQL, but on datacubes rather than tables
- Datacube = „coverage“ in stds terminology
  - Technically, datacube = array + metadata
  - Syntax close to FLOWR expressions, cf XML & JSON
- Operations: access, subsetting, processing, aggregation, fusion, encoding
- Ex:

```
for $c in (S2_L2A_32631_B01_60m)
return
  encode(
    $c[ time( "2021-04-09" ), E( 669960:729960 ), N( 4990200:5015220 )  ],
    "image/jpeg“
  )
```

Project: Implementation Hints

- Database connection object, *dbc*
- Datacube object, *dco*: python „twin“ of database object
  - connected to server via *dbc*
- Identification via datacube („coverage“) name lookup
  - OGC WCS DescribeCoverage request
- Lazy evaluation:
  - *dco* collects operations during python execution, rather than executing the code
  - Once non-transformable operation is encountered, generate & execute query
  - Continue python execution
  - Utilizes python‘s reflection capabilities

Project: Deliverables & Material

- Deliverables
  - Python library: wdc (WPCS Datacube)
    - *Correct, clean, elegant, inline-documented code*
  - Implementation documentation, including UML class & swimlane diagrams
  - User training material: Jupyter notebook
  - Sufficient test cases, collected in regression test suite
  - ...all in git repo
- Material
  - WCPS introduction, with many further links
  - Datacube server to be used