CCI Toolbox

Carsten Brockmann, Norman Fomferra & CCI Toolbox Team

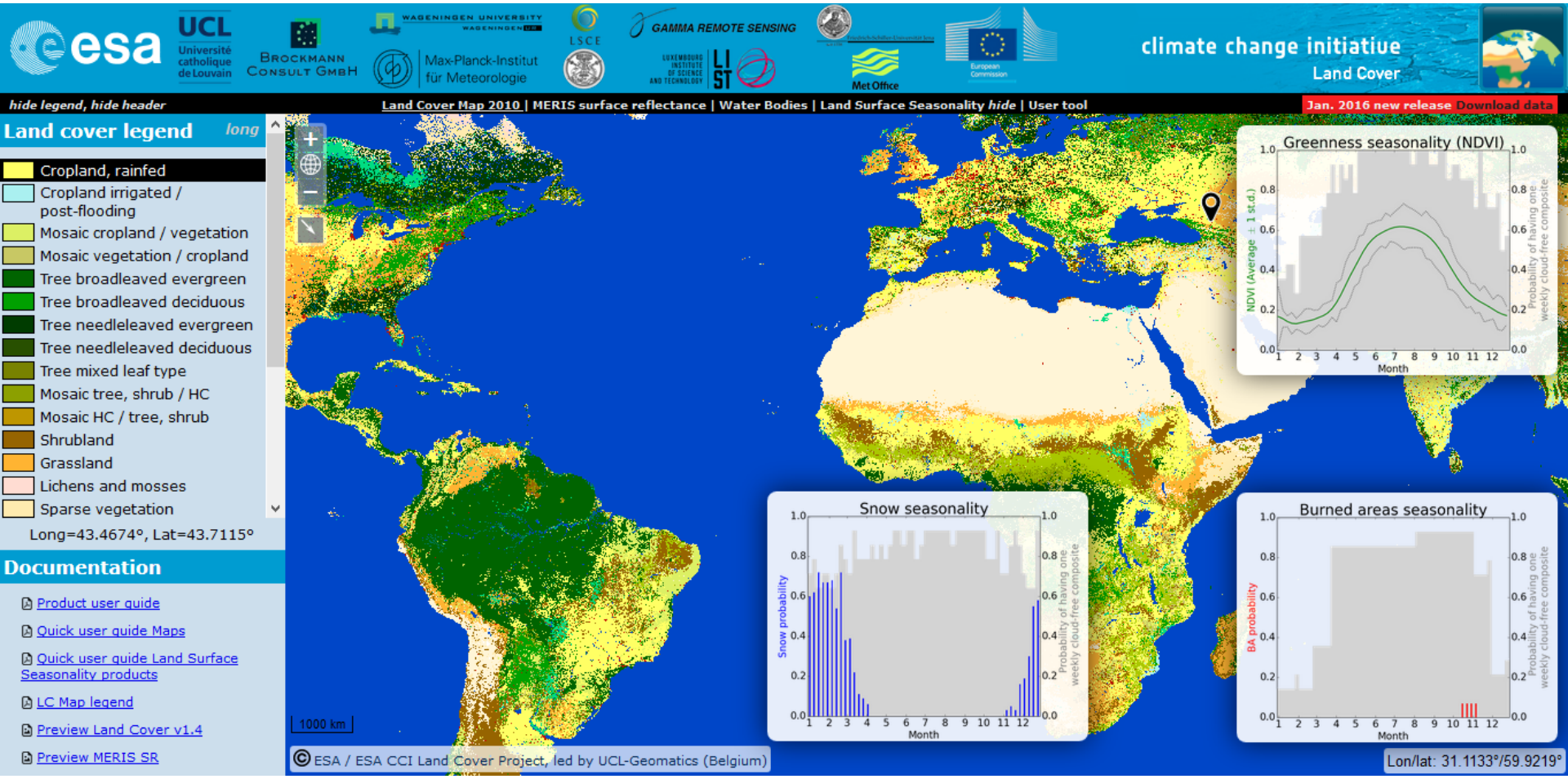
CCI Colocation, ESA ESRIN, October 2016

Presentation Overview

1. Motivation
2. Development Approach
3. Concepts & Implementation
4. User Support & Interactions
5. What's Next

6. Discussion

CCI User Tools

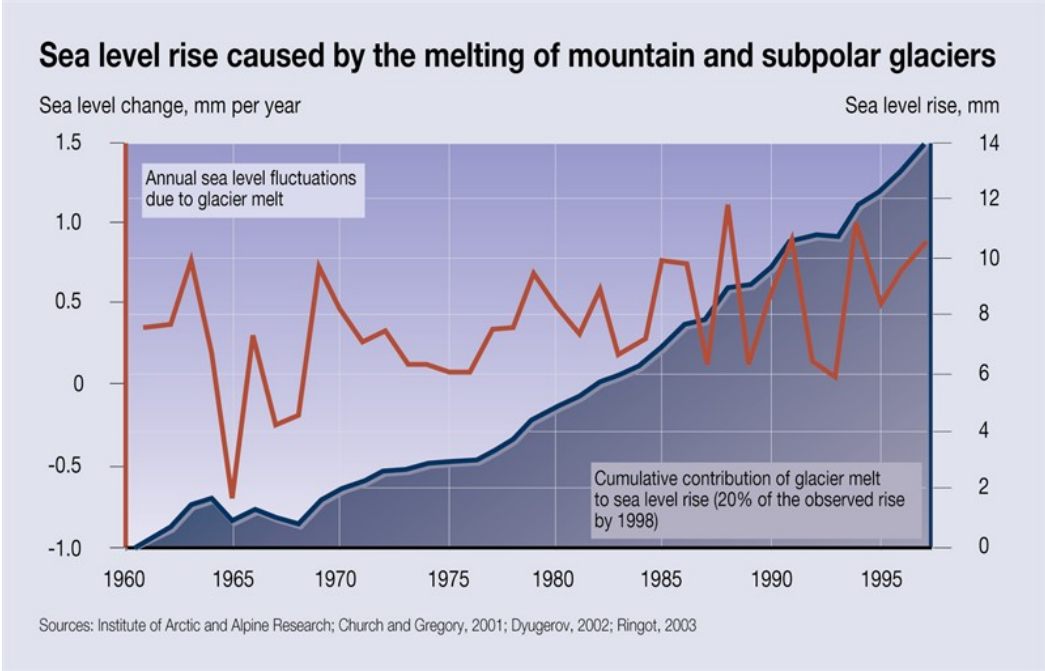


Cross-CCI Application

"What is the contribution of all glaciers to global sea level rise over a given time period in the future"

Requirements:

- ESA CCI Glacier and Sea Level + in-situ data
- Spatial and temporal aggregation, regridding, gap filling
- Topographic data from DEMs
- Spatial resampling & co-registration



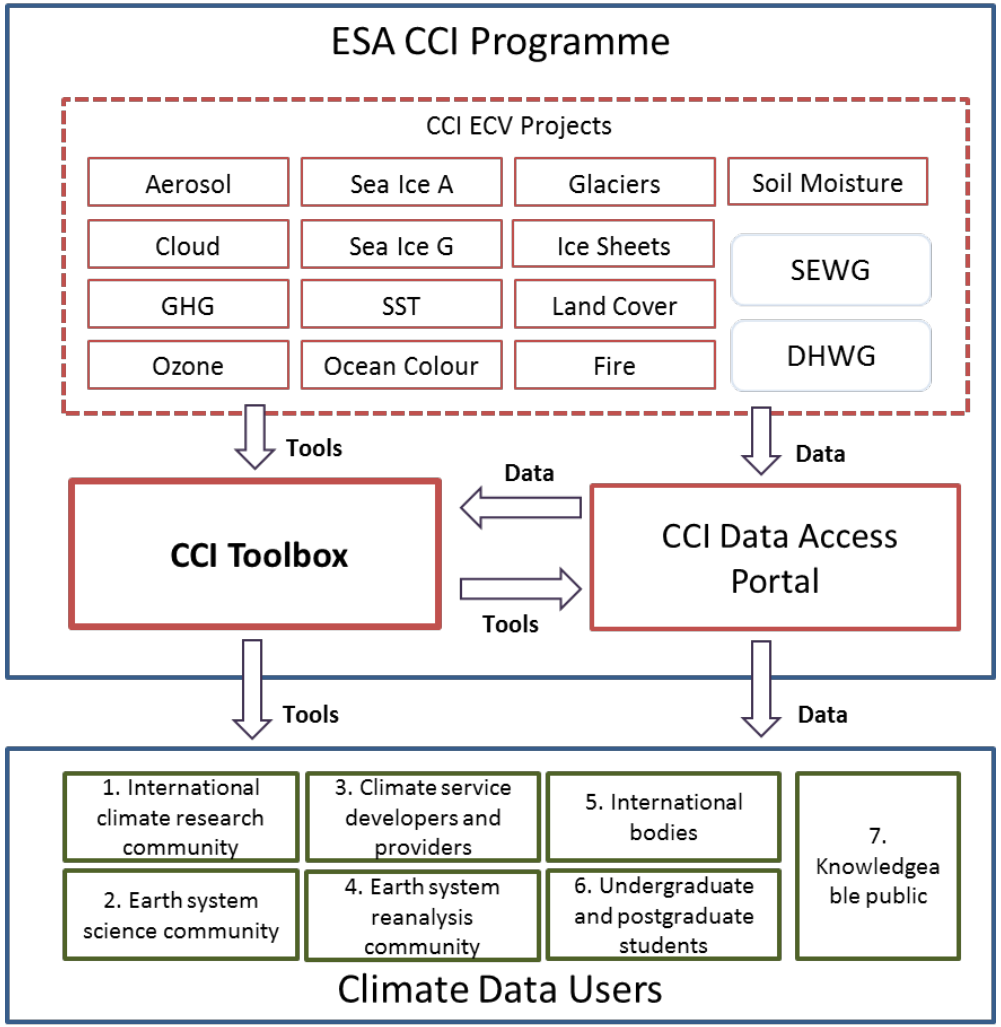
CCI Toolbox Motivation

Challenges addressed by the CCI TBX:

1. Limited means to ingest ECV data spanning different ECV types into a common data model
2. Limited means to apply algorithms homogenously across data associated with different ECV types
3. Limited means to conveniently analyse and visualise data drawn from 1 and 2 above

The main objective of the project is to equip climate users with the means to operate on CCI ECV data, overcoming these three challenges.

Context and Core Objectives



- Ingest different ECV products into a common data model
- Compute algorithms homogeneously across a common data model
- Support ECV analysis and visualisation
- Engage with the user community

Unified ECV Data Access

- All ESA CCI Open Data Portal datasets
- Local file system containing CCI ECV datasets
- Obs4MIPs & Earth System Grid Federation (ESGF) datasets
- CMIP5 and CMIP6 datasets
- Generic data access
  - FTP and HTTP standard file-based access
  - OPeNDAP Service
  - Web Map Service (WMS)
  - Web Coverage Service (WCS)
  - Web Feature Service (WFS)

Unified ECV Data Access

open data portal cci

WCRP obs4MIPs
World Climate Research Programme

CMIP5 and CMIP6 datasets

CMIP
The World Climate Research Programme's Coupled Model Intercomparison Project

Local file system containing ECV data

- Generic data access
  - FTP and HTTP standard file-based access
  - OPeNDAP Service
  - Web Map Service (WMS)
  - Web Coverage Service (WCS)
  - Web Feature Service (WFS)

Cross-ECV Operations (1/2)

- **ECV filtering**. Filtering the Common Data Model based on ECV type, in the case where an instance of the Common Data Model is holding data from multiple ECVs.
- **Simple transformations** including spatial and temporal aggregation, temporal concatenation, sub-sampling, re-projection and interpolation.
- **Geospatial polygon** filtering / masking based on parameter input of polygon, cookie cutting.
- **Geospatial point filtering**. Given a geospatial point the processor returns all temporal data for that point across time.
- **Temporal filtering**. For temporal range. Needs assumptions on if ECV data does not fit within time period, etc.
- **Quality parameter filtering**. Filtering of the common object model based on a given quality value per ECV, or a single quality value in the case where all ECV data processed shares the same quality flag in the Common Data Model.

Cross-ECV Operations (2/2)

- **Analysis, filtering**, exploration and propagation of uncertainties that accompany ECV data.
- **Calculating anomaly** information, comparing two ECVs by means and variance.
- **ECV parameter filtering**. Filtering on any ECV data feature.
- **Gap filling**. Suitable algorithms to fill geospatial and temporal gaps in observational data.
- **Ensemble**s. Calculation of statistics across ensembles of data sets.
- **Time series analyses**. Conducting time-series analysis (including Fourier transformations) from a specified subset of Common Data Model.
- **ECV comparison**. Given that different ECVs can be represented by different instances of the Common Data Model.
- **GeoTIFF layering**. Including layering of geospatial information displaying social and economic information.
- **Delta information** between two data sets / instances of the Common Data Model

Cross-ECV Operations

Multiple User Types

1. International climate research community – CMUG, CRG, …
   - *needs tools which are applicable and performant enough to investigate simultaneously multiple complete time series of CCI data sets*
2. Earth system science community – IGBP, CMIP, C4MIP, …
   - *needs tools which are applicable and performant enough to investigate multiple complete time series of CCI data sets*
3. Climate service developers and providers
   - *require repeatable exercises to be performed on CCI data in regular intervals and embed the CCI tools into their own operational procedures*
4. Earth system reanalysis community
   - *requires tools to inter-compare and make use of the available error information*
5. International bodies – UNFCCC, CEOS, IPCC, COP, …
   - *require quick, user-friendly and easy and condensed access to CCI data*
6. Undergraduate and postgraduate students
   - *perform exercises and application on representative and illustrative CCI data subsets*
7. Knowledgeable public
   - *do not have any experience in using satellite data and are not familiar with the terminology*

Existing Tools

- ECV-specific tools developed by CCI projects

- Giovanni, CEOS COVE, GrADS, UV-CDAT, Panoply
- OpenLayers, ncBrowse, Ferret, Dchart, ADAGUC, Godiva
- Apache Open Climate Workbench, Rasdaman, NASA EOSDIS Worldview, MPI's Climate Data Operators (CDO)
- ESA Earth Observation toolboxes such as Sentinel Toolboxes, SNAP

Presentation Overview

1. Motivation
2. **Development Approach**
3. Concepts & Implementation
4. User Support & Interactions
5. What's Next

6. Discussion

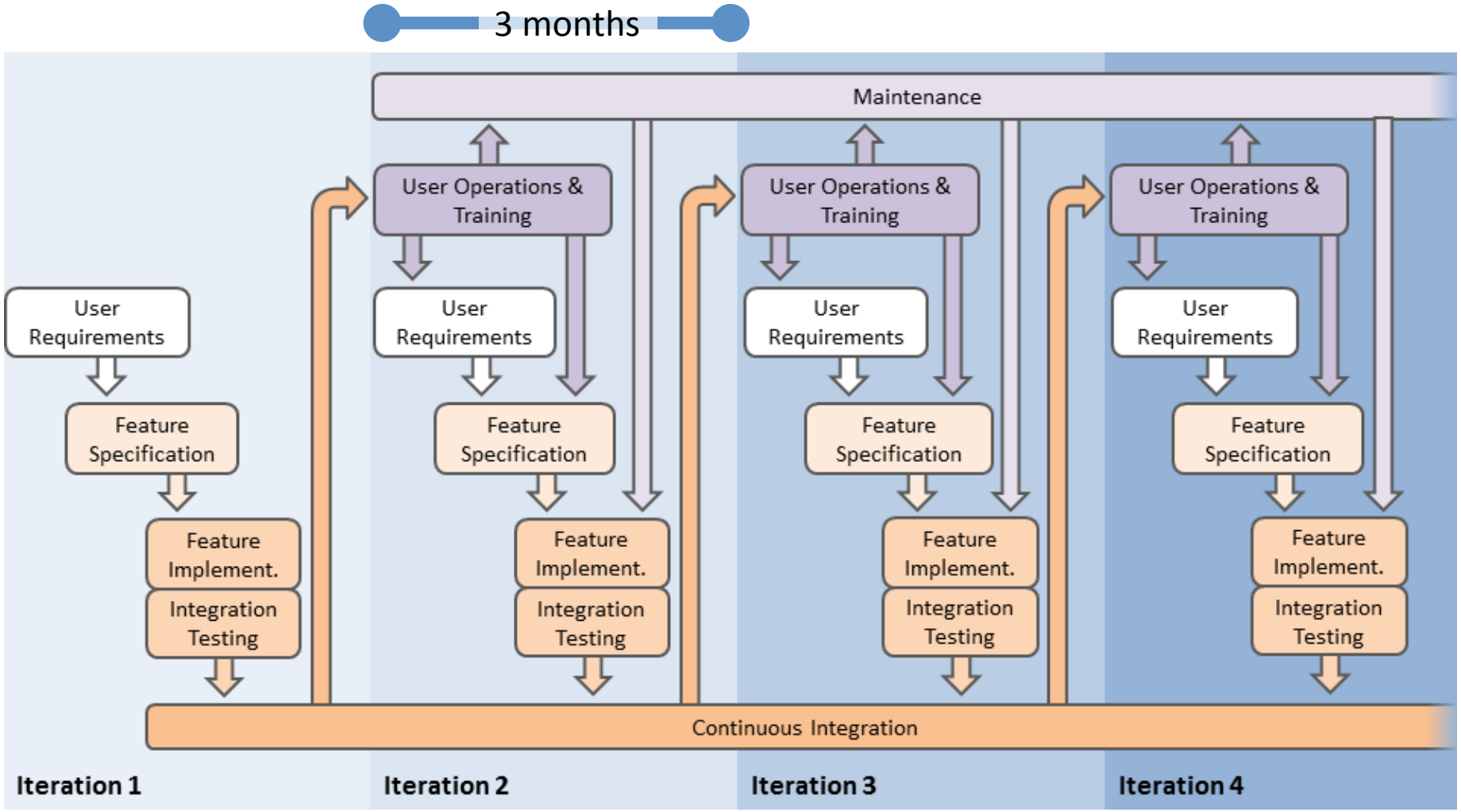
Champion Users

- Represent the 7 user types
- Develop use cases
- Be first users of every release
- Provide feedback, improve workflows
- Make sure toolbox is fit for purpose

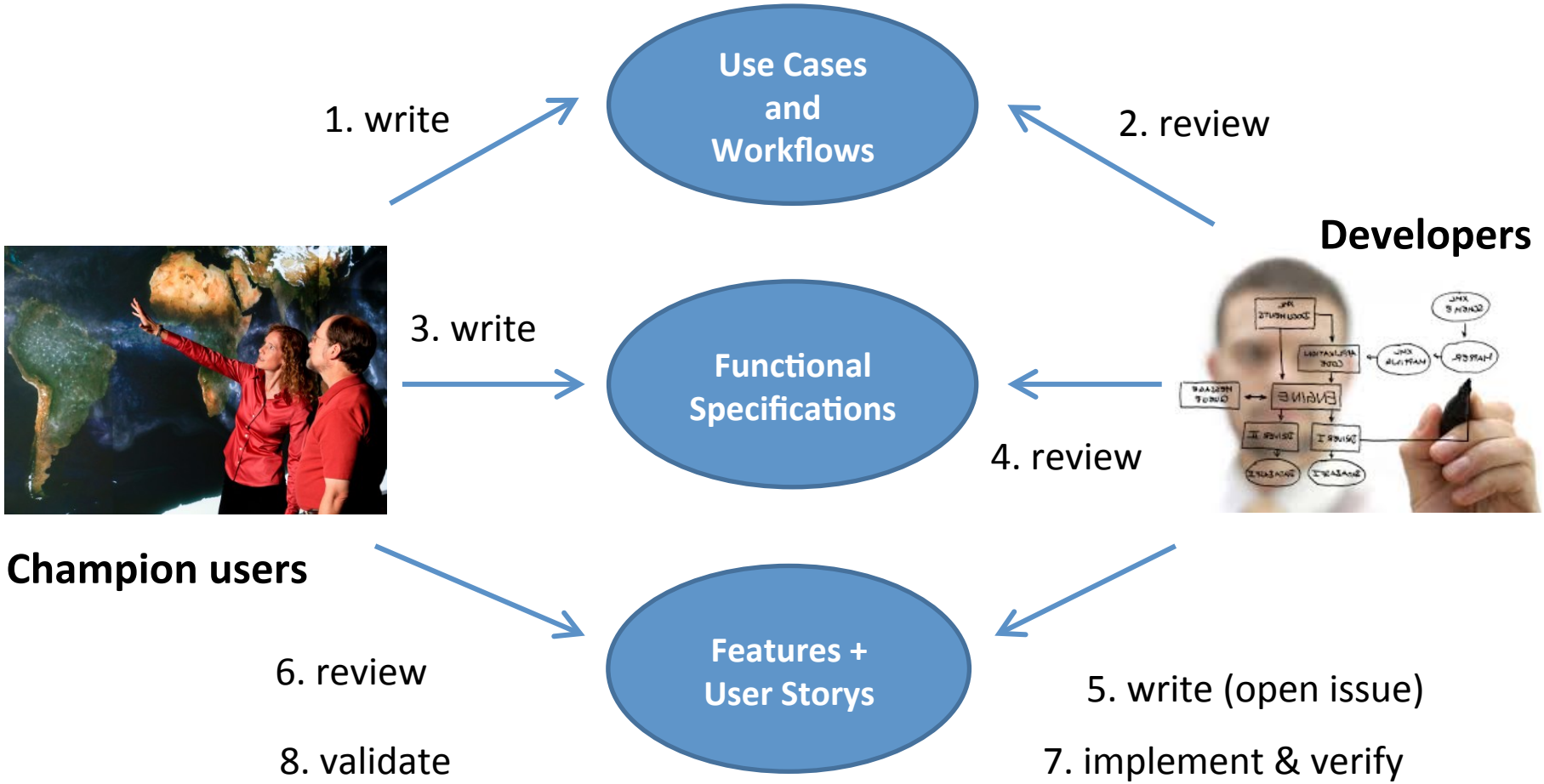
- Deutscher Wetterdienst (DWD)
- University of Reading (UoR)
- University of Zürich (UZH)

Type of application area	Champion User		
	DWD	UZH	UoR
1 - International climate research community	X	X	
2 - Earth system science community	(X)	X	
3 - Climate service developers and providers	X		
4 - Earth system reanalysis community	X		
5 - International bodies	X	X	
6 - Undergraduate and postgraduate students		X	X
7 - Knowledgeable public	X	X	

Iterative Development



Users and Developers



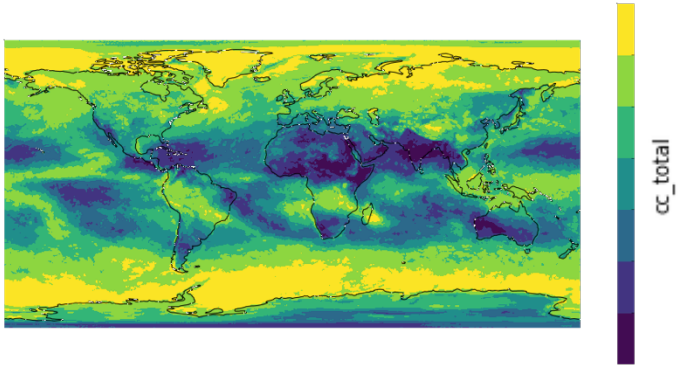
Driven by Use Cases

- We have developed **22 real-life use cases** during requirements analysis
- Involving all **7 user types**
- Every **software release addresses new use cases** and revisits older ones
- We plan to **release every 3 months**

- For the first release, we picked a very simple one. However, it addresses:
  - Loading CCI ECV data from ESA CCI ODP
  - Common pre-processing operations
  - Basic analysis

Use Case #9

Relationships between Aerosol and Cloud ECV

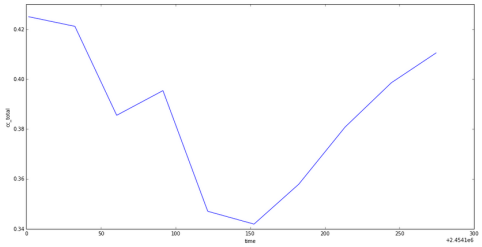


User Types:

Earth system science community

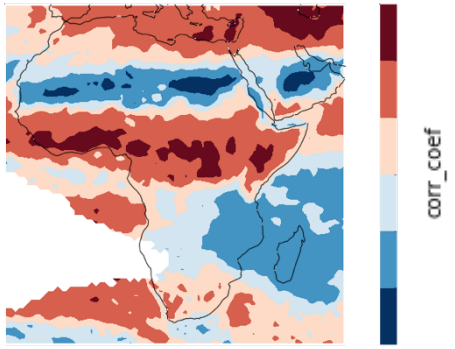
Problem Definition:

A climate scientist wishes to analyse potential correlations between Aerosol and Cloud ECVs.



Required Toolbox Features:

- Access to and ingestion of ESA CCI Aerosol and Cloud data
- Geometric adjustments
- Spatial (point, polygon) and temporal subsetting
- Visualisation of both times series at the same time: e.g. time series plot, time series animation
- Correlation analysis, scatter-plot of correlation statistics
- Saving of image and correlation statistics on disk (format options)



Presentation Overview

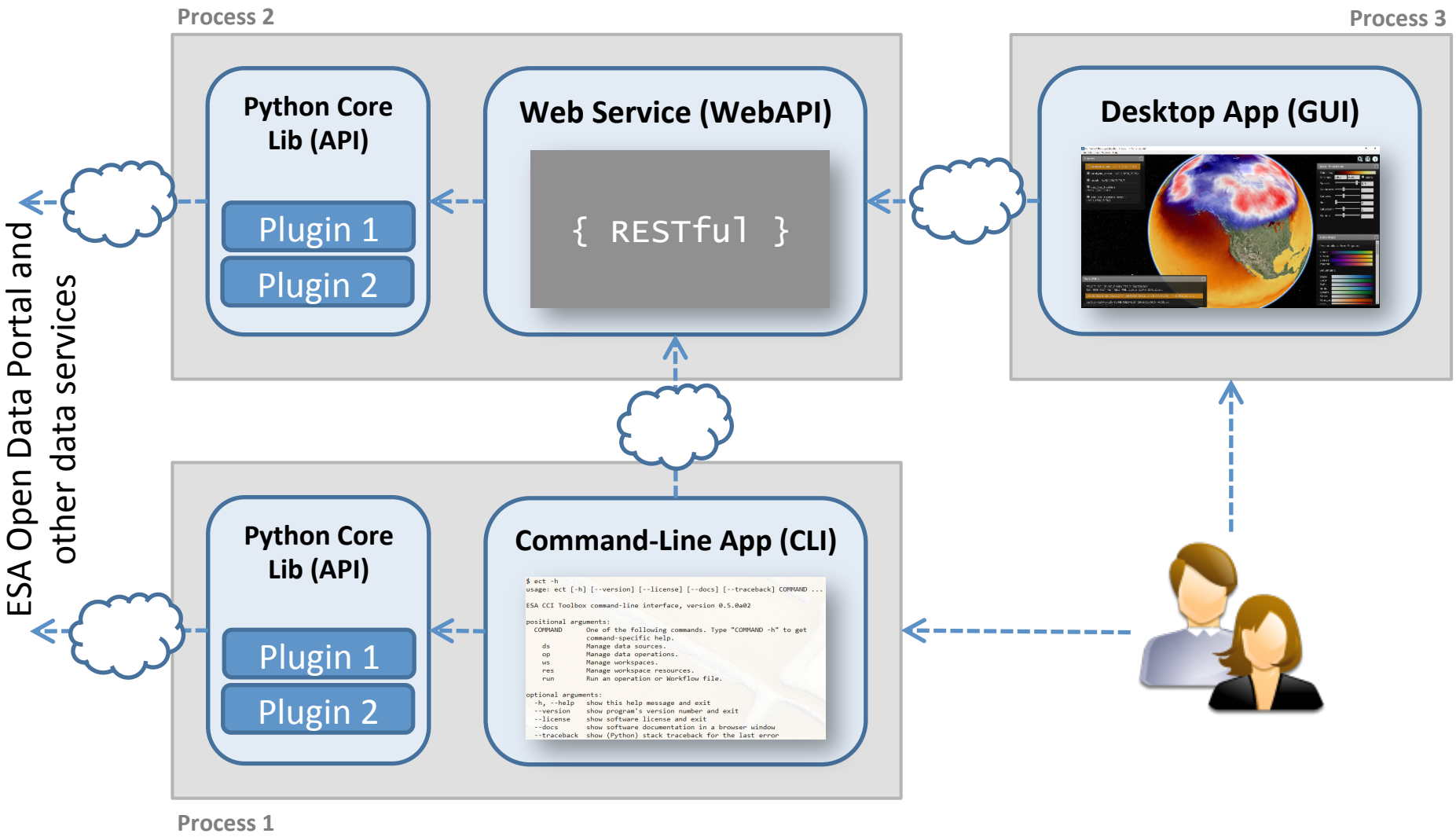
1. Motivation
2. Development Approach
3. **Concepts & Implementation**
4. User Support & Interactions
5. What's Next

6. Discussion

Basic Toolbox Design

- Desktop Application (GUI)
  - Manage data stores and call all operations from a graphical user interface
  - Visualise datasets in 2D and 3D, display layers of ECV data on an animated 3D globe
  - Uses **web technology stack** (HTML5/JS) from a desktop app (Electron)
- Web Service (WebAPI)
  - Provides all CCI Toolbox functionality through a web service
  - Holds **state information**, i.e. **opened datasets**, intermediate **operation results**, and other **resource**
- Command-Line Application (CLI)
  - Manage data stores and call all operations from a Unix shell or Windows console
- Python Core Library (API)
  - Defines API and plugin extension points for (but not limited to) **data stores** and **operations**
  - Provides numerous **data store** and **operation** implementations
  - Depends on and reuses a well-known **Python technology stack**

Hi-Level Architecture

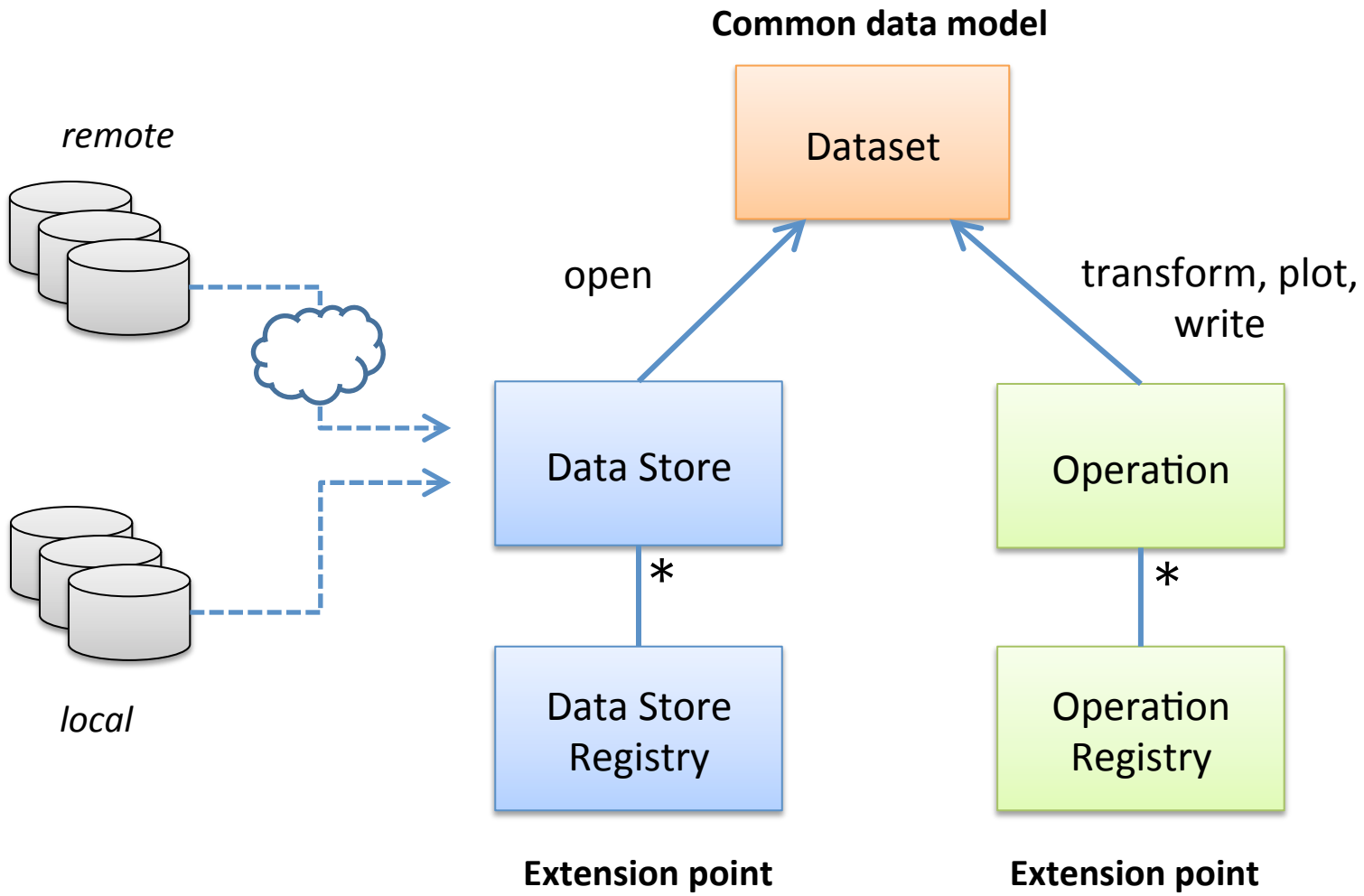


Python Technology Stack

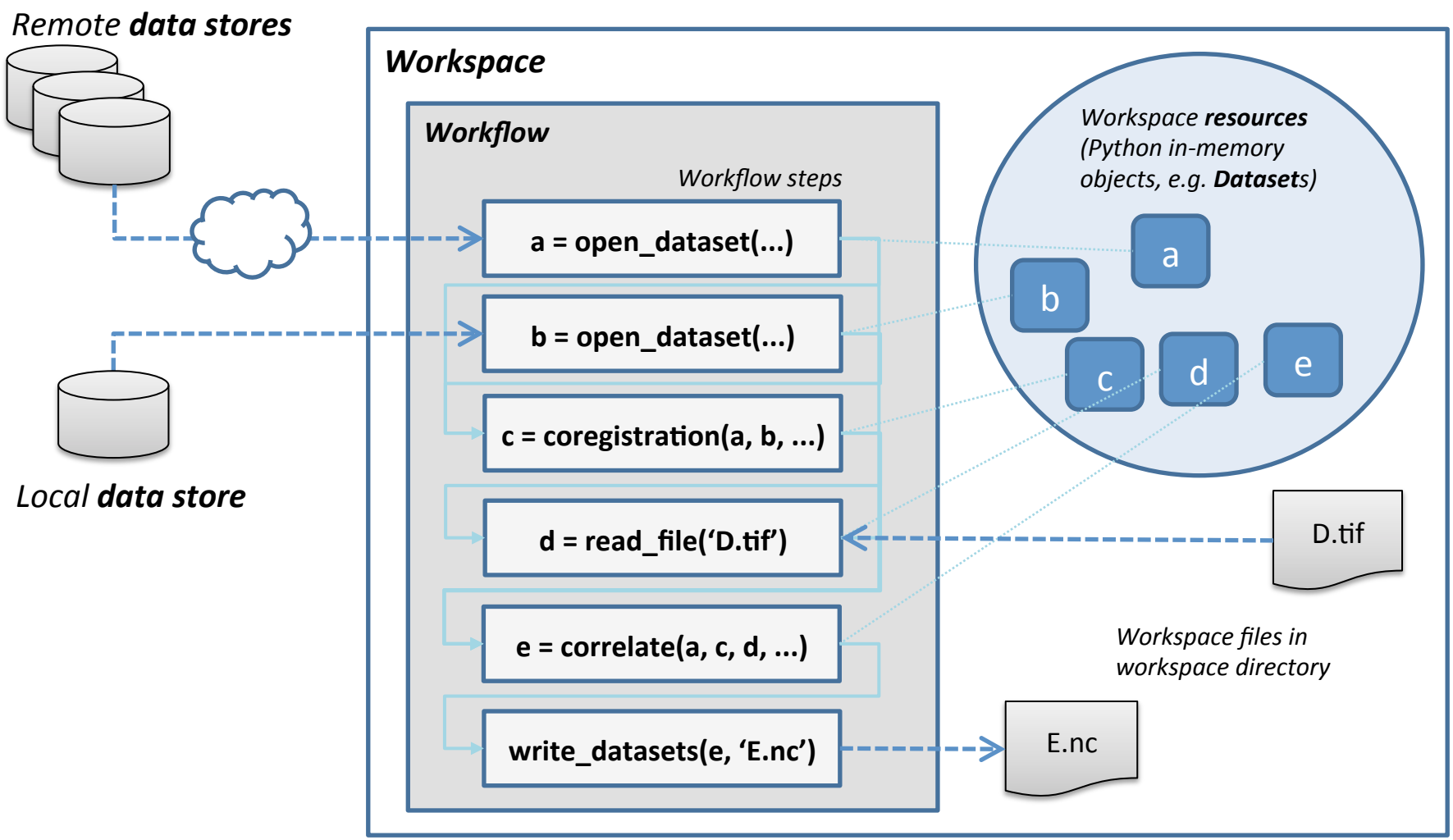
- **Python 3.5**, Miniconda distribution

- **xarray**: common data model and compute API for netCDF/CF and other gridded data sources
- **geopandas**: common data model and compute API for Shapefile-like data sources (not yet in use)
- **dask**: out of core array computations for data subsetting and aggregation
- **numba**: just-in-time machine code compiler for spatial resampling implementation
- **tornado**: for the RESTful web service
- **numpy**, **scipy**, **pandas**, **pillow**, **matplotlib**, **cartopy**, **basemap**, **netcdf4**: scientific Python programmers know them very well

Basic Concepts



Basic Concepts



Data stores, Datasets

- Data store:
  - Local or remote end point providing multiple *datasets*
  - Data stores are an import **extension point** of the CCI Toolbox
  - Example: **esacci** in Earth System Grid Federation (ESGF) at CEDA
- Dataset:
  - Entirety of contributing data files (usually time series)
  - ECV, aggregation time, product type, sources, version
  - Example: **esacci.SOILMOISTURE.day.L3S.SSMV.multi-sensor.multi-platform.COMBINED.02-1.r1**
- Opened Dataset object:
  - Gridded array data: `xarray.Dataset` instances
  - Vector (shapefile) data: `geopandas.GeoDataFrame` instances
  - Example: A dataset object opened from all data files of year 2007 of data source **esacci.SOILMOISTURE.day.L3S.SSMV.multi-sensor.multi-platform.COMBINED.02-1.r1**

CLI: List data sources

```
$ ect ds list
61 data sources found
   1: esacci.CLOUD.day.L3U.CLD_PRODUCTS.AVHRR.NOAA-15.AVHRR_NOAA.1-0.r1
   2: esacci.CLOUD.day.L3U.CLD_PRODUCTS.AVHRR.NOAA-16.AVHRR_NOAA.1-0.r1
   3: esacci.CLOUD.day.L3U.CLD_PRODUCTS.AVHRR.NOAA-17.AVHRR_NOAA.1-0.r1
   4: esacci.CLOUD.day.L3U.CLD_PRODUCTS.AVHRR.NOAA-18.AVHRR_NOAA.1-0.r1
   5: esacci.CLOUD.day.L3U.CLD_PRODUCTS.MODIS.Aqua.MODIS_AQUA.1-0.r1
   6: esacci.CLOUD.day.L3U.CLD_PRODUCTS.MODIS.Terra.MODIS_TERRA.1-0.r1
   7: esacci.CLOUD.mon.L3C.CLD_PRODUCTS.AVHRR.NOAA-15.AVHRR_NOAA.1-0.r1
   8: esacci.CLOUD.mon.L3C.CLD_PRODUCTS.AVHRR.NOAA-16.AVHRR_NOAA.1-0.r1
   9: esacci.CLOUD.mon.L3C.CLD_PRODUCTS.AVHRR.NOAA-17.AVHRR_NOAA.1-0.r1
  10: esacci.CLOUD.mon.L3C.CLD_PRODUCTS.AVHRR.NOAA-18.AVHRR_NOAA.1-0.r1
  11: esacci.CLOUD.mon.L3C.CLD_PRODUCTS.MODIS.Aqua.MODIS_AQUA.1-0.r1
  12: esacci.CLOUD.mon.L3C.CLD_PRODUCTS.MODIS.Terra.MODIS_TERRA.1-0.r1
  13: esacci.CLOUD.mon.L3S.CLD_PRODUCTS.AVHRR.multi-platform.AVHRR_MERGED.1-0.r1
  14: esacci.CLOUD.mon.L3S.CLD_PRODUCTS.MODIS.multi-platform.MODIS_MERGED.1-0.r1
  15: esacci.FIRE.day.L4.BA.multi-sensor.multi-platform.MERIS.v4-1.r1
  16: esacci.GHG.day.L2.CH4.TANSO-FTS.GOSAT.GOSAT.v2-3-6.r1
  17: esacci.GHG.day.L2.CH4.TANSO-FTS.GOSAT.GOSAT.v2-3-6.r2
  18: esacci.GHG.day.L2.CH4.TANSO-FTS.GOSAT.GOSAT.v2-3-7.r1
  19: esacci.GHG.day.L2.CH4.TANSO-FTS.GOSAT.GOSAT.v2-3-7.r2
```

CLI: Show data source info

```
$ ect ds info esacci.CLOUD.day.L3U.CLD_PRODUCTS.AVHRR.NOAA-15.AVHRR_NOAA.1-0.r1

Data source esacci.CLOUD.day.L3U.CLD_PRODUCTS.AVHRR.NOAA-15.AVHRR_NOAA.1-0.r1
==============================================================================

cci_project:             CLOUD
data_type:               CLD_PRODUCTS
number_of_aggregations:  0
number_of_files:         1000
platform_id:             NOAA-15
processing_level:        L3U
product_string:          AVHRR_NOAA
product_version:         1-0
project:                 esacci
realization:             r1
sensor_id:               AVHRR
size:                    881315464000
time_frequency:          day
version:                 20160704

Temporal coverage: 2007-01-01 to 2009-11-05
```

Operations

- **Functions** that take one or more **dataset** or other objects as input and produce a new **dataset** or other object
- In the narrow sense:
  - Any Python function
  - Plus additional input/output meta-information
  - Registered in the CCI Toolbox' operation registry
- Goal is to visualise, analyse, process datasets only by means of registered operations → **equivalence of GUI, CLI, and API**
- Operations are the most important extension point of the CCI Toolbox
- A special operation type is the *Workflow*

CLI: Show operations

```
$ ect op list
21 operations found
   1: coregister
   2: harmonize
   3: open_dataset
   4: pearson_correlation
   5: plot_map
   6: read_json
   7: read_netcdf
   8: read_object
   9: read_text
  10: save_dataset
  11: select_var
  12: subset_spatial
  13: subset_temporal
  14: subset_temporal_index
  15: tseries_mean
  16: tseries_point
  17: write_json
  18: write_netcdf3
  19: write_netcdf4
  20: write_object
  21: write_text
```

Options

- Query by name
- Query by tag

CLI: Show operation info

```
$ ect op info  pearson_correlation

Operation ect.ops.correlation.pearson_correlation
=================================================

Do product moment Pearson's correlation analysis. See

http://www.statsoft.com/Textbook/Statistics-Glossary/P/button/p#Pearson%20Correlation
http://support.minitab.com/en-us/minitab-express/1/help-and-how-to/modeling-statistics/regre
erpret-the-results/

Inputs:
  ds_y (Dataset)
      The 'dependent' dataset
  ds_x (Dataset)
      The 'variable' dataset
  var_y (str)
      Dataset variable to use for correlation analysis in the 'dependent' dataset
  var_x (str)
      Dataset variable to use for correlation analysis in the 'variable' dataset
  file (str)
      Filepath variable. If given, this is where the results will be saved in a text file.
      default value: None

Output:
  return (Dataset)
```

API: Register Operation

```python
import xarray as xr
from ect.core.op import op
from scipy.stats import pearsonr


@op(tags=['correlation'])
def pearson_correlation(ds_y: xr.Dataset, ds_x: xr.Dataset,
                        var_y: str, var_x: str, file: str = None) -> xr.Dataset:
    """
    Do product moment Pearson's correlation analysis. See

    http://www.statsoft.com/Textbook/Statistics-Glossary/P/button/p#Pearson%20Correlation
    http://support.minitab.com/en-us/minitab-express/1/help-and-how-to/modeling-statistics

    :param ds_y: The 'dependent' dataset
    :param ds_x: The 'variable' dataset
    :param var_y: Dataset variable to use for correlation analysis in the
    'dependent' dataset
    :param var_x: Dataset variable to us
    'variable' dataset
    :param file: If given, this is the file path where the results will
    be saved in a text file.
    """
    if len(ds_y[var_y].dims) != 1 or len(ds_x[var_x].dims) != 1:
        raise ValueError('Person correlation for multi-dimensional variables is not yet im
```

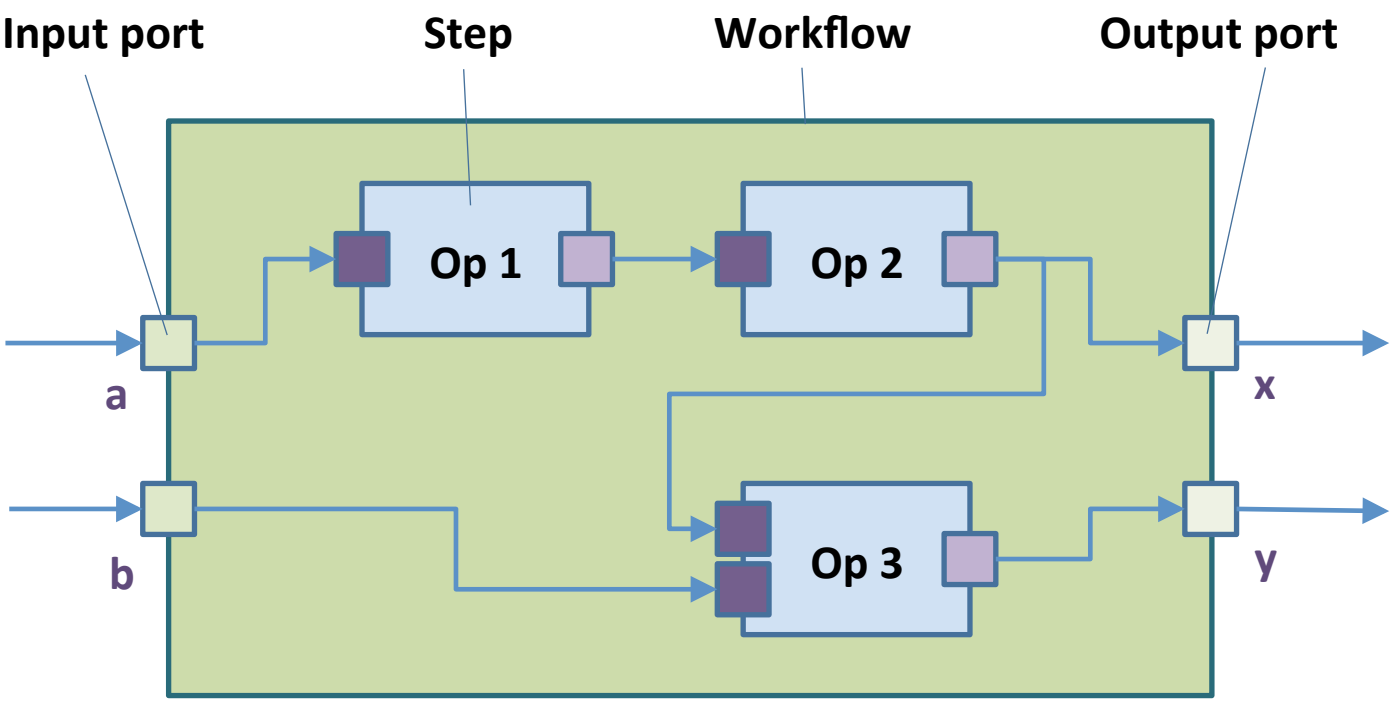
Import xarray, CCI Toolbox package(s), and a science lib

Write Python function, add @op() decorator

Operation meta-info retrieved from Python type annotations

Operation meta-info retrieved Python docstring

Workflows



- Processing graph
- Contains workflow steps
- Different step types
- Has named input/output ports

- Monitoring
- Acts like operation → is an operation
- Building block for higher level operations

Workflow Step Types

- Operation steps
  - reference any *registered* Python function
- Expression steps
  - evaluate Python expressions (e.g. subtract two dataset variables)
- Script steps
  - execute Python scripts
- Sub-process steps
  - execute any external programs (e.g. call ECV-specific tool, CDO, SNAP Operators (SNAPPy) …)
- Workflow steps
  - reference another workflow (in a JSON file)

Command-Line Application (CLI)

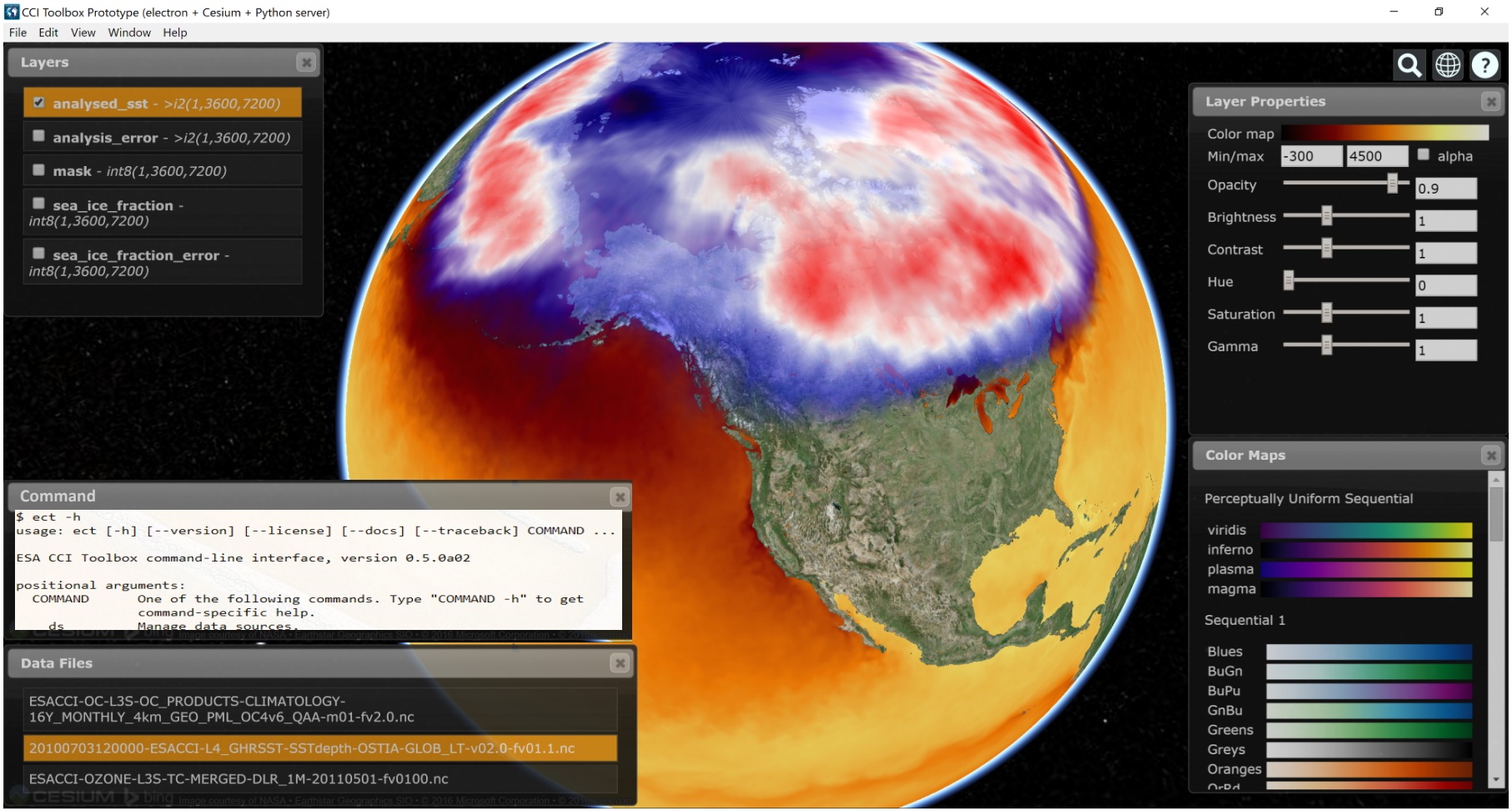
```
$ ect -h
usage: ect [-h] [--version] [--license] [--docs] [--traceback] COMMAND ...

ESA CCI Toolbox command-line interface, version 0.5.0a02

positional arguments:
  COMMAND       One of the following commands. Type "COMMAND -h" to get
                command-specific help.
    ds          Manage data sources.
    op          Manage data operations.
    ws          Manage workspaces.
    res         Manage workspace resources.
    run         Run an operation or Workflow file.

optional arguments:
  -h, --help    show this help message and exit
  --version     show program's version number and exit
  --license     show software license and exit
  --docs        show software documentation in a browser window
  --traceback   show (Python) stack traceback for the last error
```

Desktop Application (GUI)



Demo code available here: https://github.com/CCI-Tools/demo

Workspaces and Resources

- Workspaces
  - combine **datasets and operations** used within **analyses** performed by users
  - support **interactive mode** in **CLI and GUI**
  - **store operation results** and make them available to other operations as named *workspace resources*
- Workspace Resources
  - Output of any CCI Toolbox operation
  - Represented as **workflow steps**
- Workspace commands
  - New, open, save, close, status
- Physically, workspaces are just directories
  - hidden directory (`.ect-workspace`) for internal state info
  - hidden workflow file (`.ect-workspace/workflow.json`) stores resources
  - user files in the workspace directory are given with relative paths
  - ZIP it, share it

CLI: Workspace Management

```
$ ect ws -h
usage: ect ws [-h] COMMAND ...

Manage workspaces.

positional arguments:
  COMMAND     One of the following commands. Type "COMMAND -h" for help.
    init      Initialize workspace.
    new       Create new in-memory workspace.
    open      Open workspace.
    close     Close workspace.
    save      Save workspace.
    run       Run operation.
    del       Delete workspace.
    clean     Clean workspace (removes all resources).
    status    Print workspace information.
    list      List all opened workspaces.
    exit      Exit interactive mode. Closes all open workspaces.
```

CLI: Resource Management

```
$ ect res -h
usage: ect res [-h] COMMAND ...

Manage workspace resources.

positional arguments:
  COMMAND     One of the following commands. Type "COMMAND -h" for help.
    open      Open a dataset from a data source and set a resource.
    read      Read an object from a file and set a resource.
    write     Write a resource to a file.
    set       Set a resource from the result of an operation.
    del       Delete a resource.
    print     If EXPR is omitted, print value of all current
              resources.Otherwise, if EXPR identifies a resource, print its
              value.Else print the value of a (Python) expression evaluated in
              the context of the current workspace.
    plot      Plot a resource or the value of a (Python) expression evaluated
              in the context of the current workspace.
```

CLI: A simple session

- ect ws new
- ect res read X precip_and_temp.nc
- ect res set Y tseries_point ds=X lat=0 lon=0
- ect res plot Y -v temperature
- ect ws save
- ect ws close

Use case #9 Demo

- **IPython notebook** demo on the correlation analysis using the API can be found on GitHub: https://github.com/CCI-Tools/ect-core/blob/master/notebooks/ect-uc9.ipynb

- Demo on the correlation analysis **using the CLI** can be found on GitHub: https://github.com/CCI-Tools/ect-core/wiki/UC9-CLI-Demo

Cloud Computing

- Cloud readiness: CLI (and API) allow for CCI Toolbox operation in “headless” servers, cloud computing environments **out of the box**.

- Possible future development:
  - The web service that serves the CLI and GUI is currently run in the background on the user’s computer (at address “localhost”)
  - Design of the CCI Toolbox allows using this web service at any other address in the internet. In principle.
    - not dealing with security/authorisation/permissions/quota/scheduling yet
  - This would allow for remote processing
    - with local data access to specific datasets
    - serving specific operations due to available soft- and hardware
  - However, our web service is currently designed for optimal performance for CLI/GUI user interactions, as a detail of its architecture. Not necessarily for machine-machine interoperability.
    - Comply with OGC Web Processing Service (WPS)

Presentation Overview

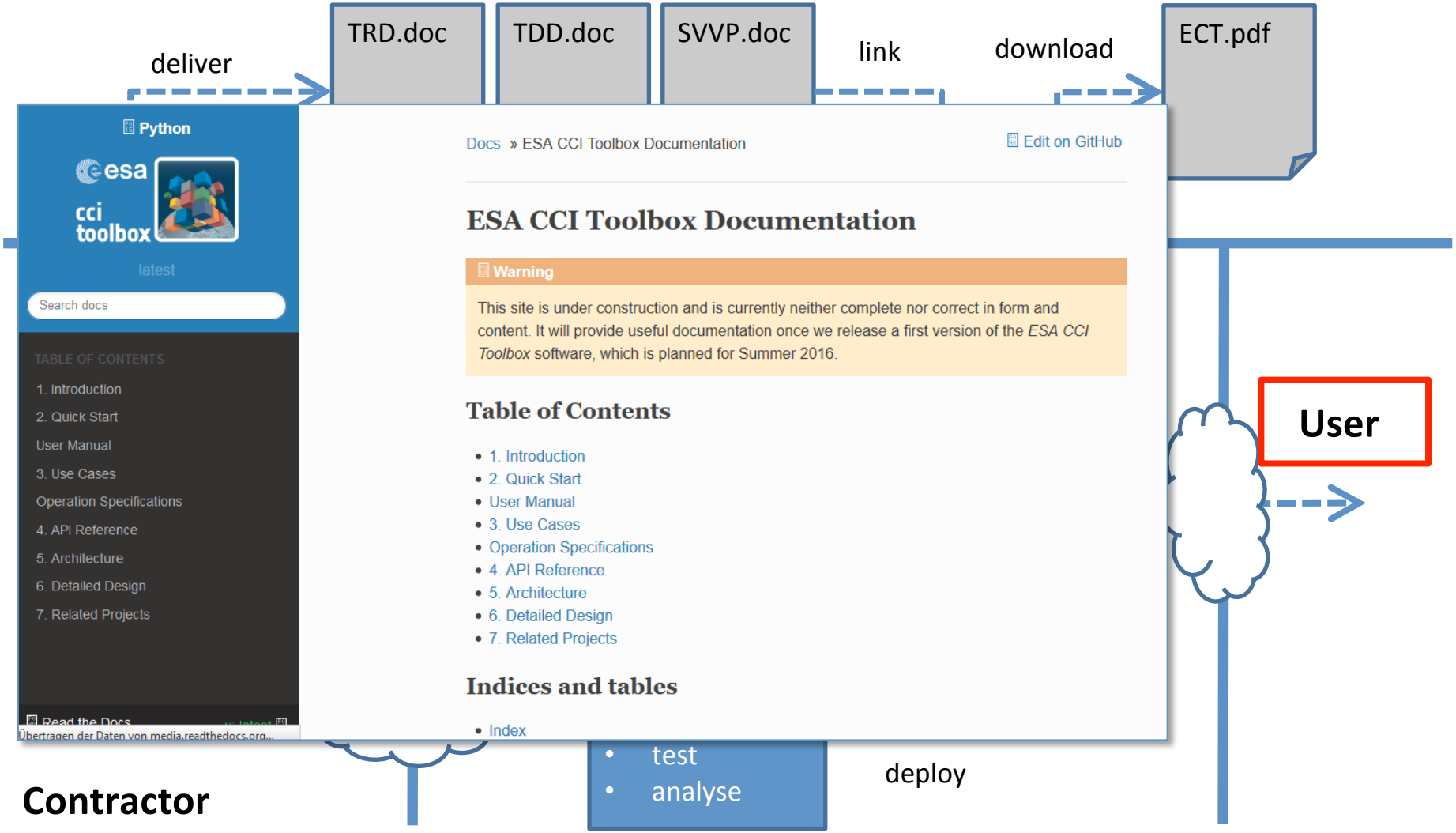
1. Motivation
2. Development Approach
3. Concepts & Implementation
4. **User Support & Interactions**
5. What's Next

6. Discussion

Documentation Approach

- Most of the technical documentation content will be
  - in the code: „doc-strings“ in source code
  - with the code: doc/ folder containing RST files
- Documentation will be generated from RST and doc-strings (Sphinx tool)
  - HTML, LaTeX, ePub, PDF, ...
- HTML version published to ReadTheDocs
  - Download as PDF
  - Always related to a specific software release version (tag)

Single Source Documentation



Development Environment

ESA CCI Toolbox

Cross-ECV tools supporting the ESA Climate Change Initiative (CCI)

ESA ECSAT, Harwell, UK http://cci.esa.int/

Repositories | People 5 | Teams 1 | Settings

Filters Find a repository... New repository

ect-core Python ★ 1 ⑂ 0

The Python core of the ESA CCI Toolbox (ECT)

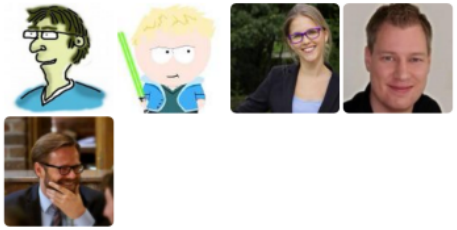
Updated 12 hours ago

ect-conda Shell ★ 0 ⑂ 0

Installer and Conda package for ESA CCI Toolbox

Updated 3 days ago

People 5 >



Invite someone

Visit us at
https://github.com/CCI-Tools

GitHub Repository

CCI-Tools / ect-core

Unwatch 5 | Star 1 | Fork 0

Code | Issues 27 | Pull requests 0 | Projects 0 | Wiki | Pulse | Graphs | Settings

The Python core of the ESA CCI Toolbox (ECT) — Edit

880 commits | 2 branches | 1 release | 4 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

forman fixed broken test — Latest commit 06aca50 23 hours ago

doc	updated CLI doc	6 days ago
ect	cleaned up cli	23 hours ago
notebooks	Coregistration with resampling routines works	6 days ago
test	fixed broken test	23 hours ago
.gitignore	Exclude the ECT workspace from gitignore	12 days ago
.travis.yml	require pyqt explicitly (so that DLLs are correctly linked on Windows)	2 days ago
LICENSE.md	switched to MIT license	19 days ago
README.md	updated packages	4 days ago

Alpha Release Available

Pre-release

v0.5.0a01

59e8d59

Alpha release 1

mzuehlke released this 2 days ago · **3 commits** to master since this release

This test release of the CCI Toolbox contains a first version of the command-line interface (CLI) and Python API. Its functional range comprises the features required to perform our simple use case #9 using the CLI:

- Read (netCDF) datasets from ESA CCI Open Data Portal
- Spatial and temporal subsetting
- Spatial resampling (up- and downsampling)
- Spatial coregistration
- Basic correlation analysis
- Basic plotting
- Writing to netCDF3 and -4

Downloads

ect-0.5.0a01-Linux-x86_64.sh	276 MB
ect-0.5.0a01-Windows-x86_64.exe	339 MB
Source code (zip)	
Source code (tar.gz)	

Software Installers

Issue Tracking

CCI-Tools / ect-core

Unwatch 5 | Star 1 | Fork 0

Code | Issues 27 | Pull requests 0 | Projects 0 | Wiki | Pulse | Graphs | Settings

Filters | is:issue is:open | Labels | Milestones | New issue

27 Open | 30 Closed | Author | Labels | Milestones | Assignee | Sort

- [SSL: CERTIFICATE_VERIFY_FAILED] when accessing ESA ODP via HTTPS api bug help wanted — 1
  #64 opened 23 hours ago by forman
- Temporal subset should be able to recognize time in Julian days api uc09 — 1
  #60 opened 2 days ago by JanisGailis
- CLI (and WebAPI) take too long to launch cli enhancement webapi
  #58 opened 2 days ago by forman v2.0
- Pearson correlation time by time uc09
  #57 opened 3 days ago by JanisGailis v1.0
- Pearson correlation pixel by pixel correlation uc09
  #56 opened 3 days ago by JanisGailis v1.0

Contributing

- Improve software
  - Download and install toolbox
  - Try it
  - Report suggestions and bugs in the **very simple issue tracker**

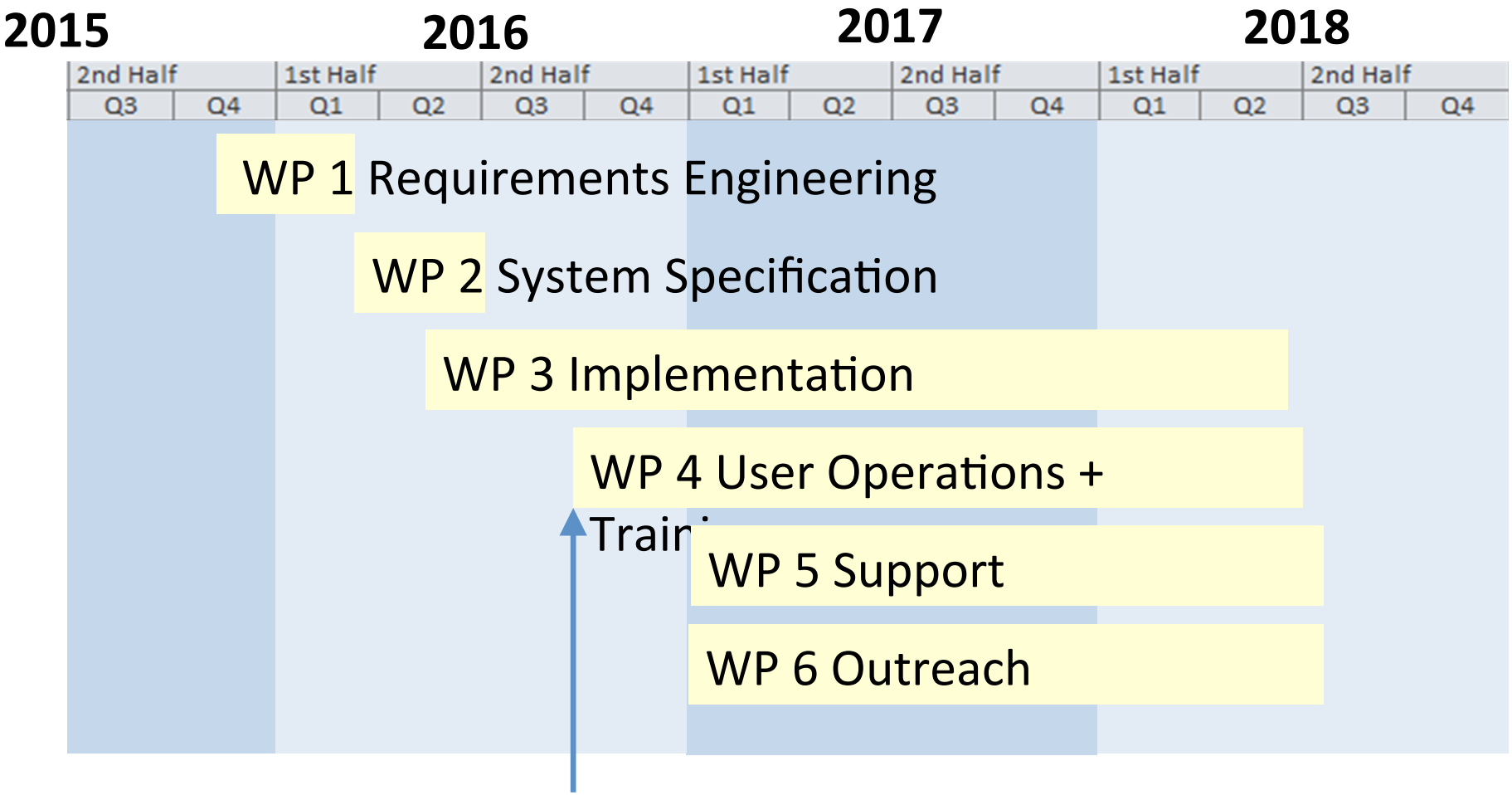
- Collaborative development
  - **Fork code repository** on GitHub, it's easy
  - **Clone** forked repository on **your computer**
  - **Change code**, e.g. add operation, fix bug
  - **Push changed code** to your forked repository
  - **Send** us a **pull request**

Presentation Overview

1. Motivation
2. Development Approach
3. Concepts & Implementation
4. User Support & Interactions
5. **What's Next**

6. Discussion

Project Schedule

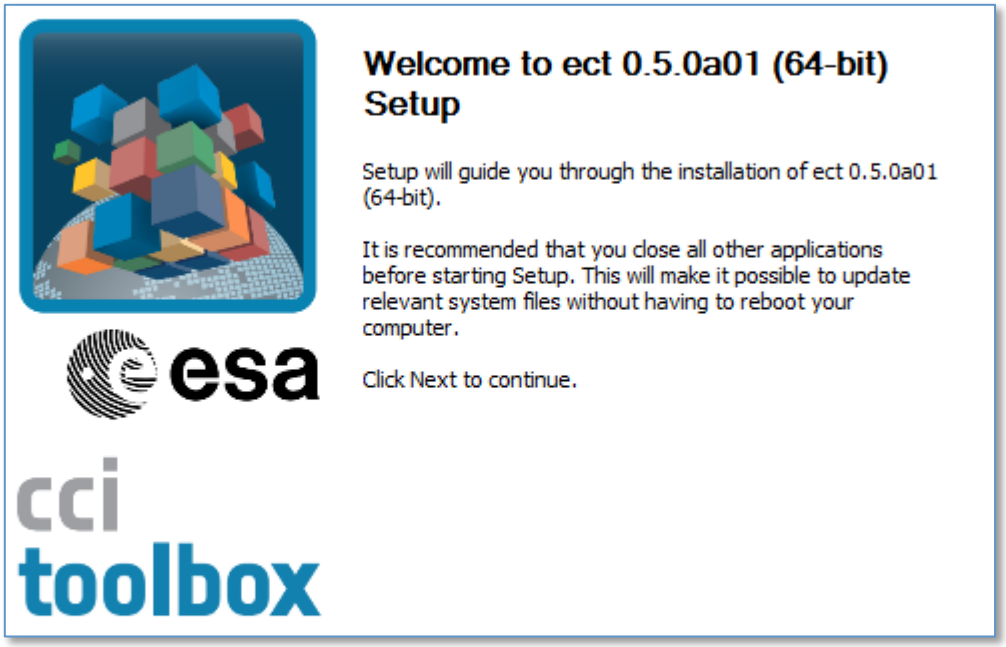


Next Steps

- Test and publish final version 1
  - Release date 14.10.2016

- Version 2 alpha planned for mid January 2017
- Plan Iteration 2
  - Use case including different data types
  - Use case requiring visualisation
- Main Features of v2
  - Desktop GUI
  - OPeNDAP
  - Shapefiles (Ice-Sheets and Glaciers CCI)

Summary



Adressing 7 different user communities

Champion users & use cases driving the development

Open, transparent, agile development process, iterating through champion users

- Features
  - Easy, transparent access to ESA CCI ECV datasets
  - Workflow-based approach
  - Use any dataset/operation via the API, CLI, and GUI in a consistent way
  - Easily write new operations and publish them the API, CLI, and GUI
  - Out-of-core computations
  - Well-acknowledged, scientific Python technology stack
  - Interfacing existing tools

Presentation Overview

1. Motivation
2. Development Approach
3. Concepts & Implementation
4. User Support & Interactions
5. What's Next

6. Discussion

Discussion Points

- We want the Open Data Portal and theToolbox to be a success story. User engagement is critical for this.

  Are our plans and the means for user engagement appropriate? Recommendations to change something? How do we deal with the transition from CCI to CCI+ (gap? different/new actors)

- DATA STORES and OPERATIONS are important concepts.

  Today we consider ESGF (CCI, Obs4MIPS, CMIP) + local, as (almost) sufficient stores. Would you agree? What about the data available at individual CCIs? What about different versions of products?

  Which OPERATIONS are mandatory?

  Which OPERATIONS could be problematic (e.g. gap filling)? How can offer them (to be attractive) but assure best scientific quality and soundness? Engangement of CCI teams in OPERATION definition?

- Link between CCI Open Data Portal and CCI Toolbox

  What are do DOs and DON'Ts?

SPARE

CCI Data Standards

- Gridded NetCDF/CF datasets
  - CF can be applied in multiple ways
  - Within CCI, CF *is* applied in multiple ways and with various extends
    - lat vs. latitude
    - varying geo-coding references, cell center vs. cell borders
    - various time units
    - different global attribute sets
- Recommendations from CCI Toolbox:
  - Define **common baseline** of CF conventions to be applied, **and how**. Examples:
    - must use latitude-longitude dim names and coordinate variables, not lat-lon
    - must provide cell boundary coordinate variables
    - must provide value range for variables
    - must use auxiliary variables, if any
  - Define the set of **common global attributes**
    - on specific data transformation
      - identify the ones that remain
      - identify the ones that change, define in which way
  - Each product must **pass CF compliancy checker**

CCI Data Standards

- For each dataset:
  - Identify variables, colobars and value ranges for “quicklook” generation

Hint System

- Ideas
  - Give users a hint what they can/should do with a given ECV dataset/variable
  - Suggest, allow, or disable operations for selected dataset/ variable
  - Provide operation parameter defaults
  - Suggest possible following operations

- Solution
  - Extendible hint database
  - Mapping between ECV dataset/variables and operation information

  - A lot of work to setup and maintain, not an easy task, many opinions