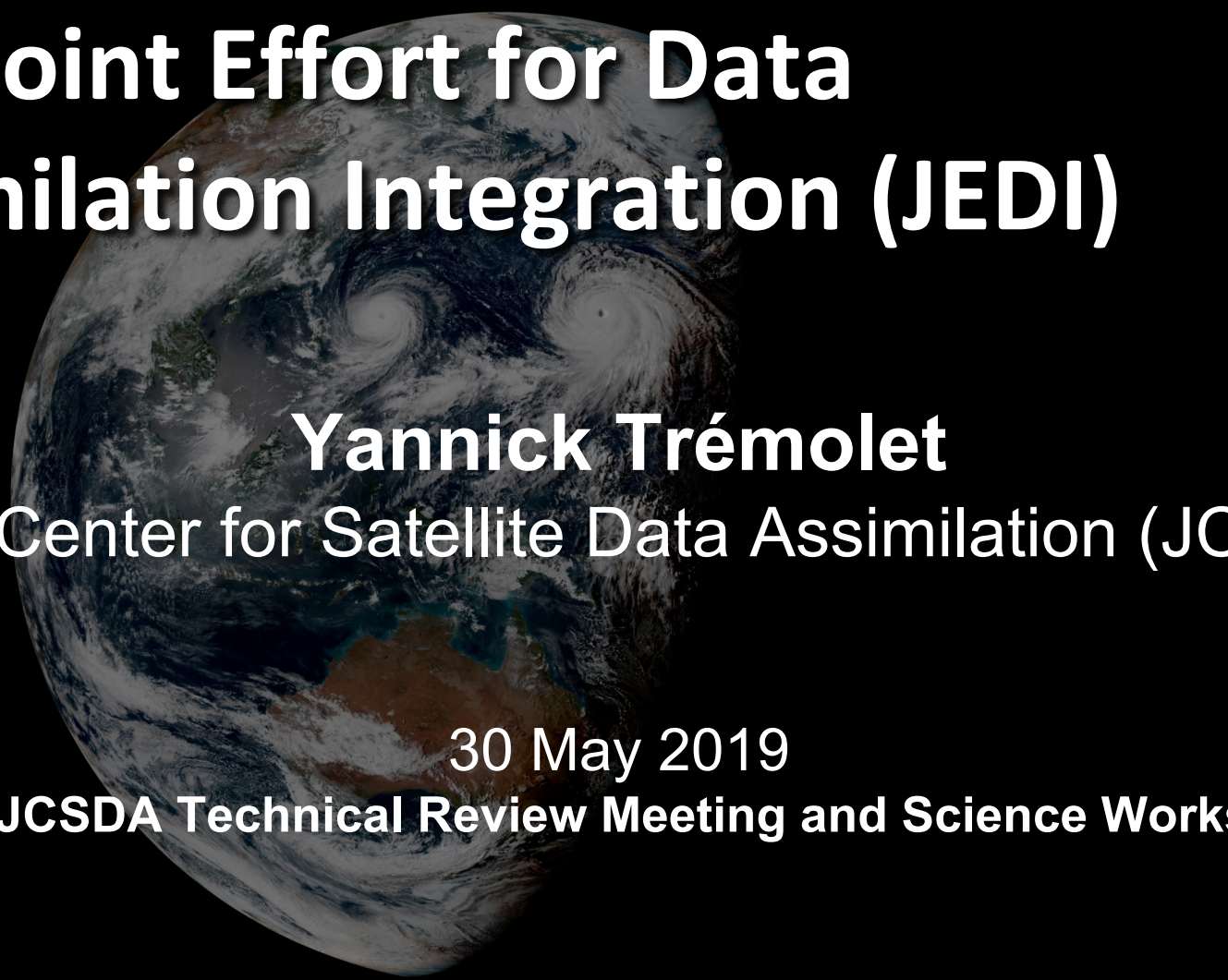
The Joint Effort for Data assimilation Integration (JEDI)



Yannick Trémolet
Joint Center for Satellite Data Assimilation (JCSDA)

30 May 2019

17th JCSDA Technical Review Meeting and Science Workshop

JEDI is a Joint Effort

JEDI core-team: Anna Shlyaeva, Benjamin Ménétrier, Clémentine Gas, Dan Holdaway, Mark Miesch, Mark Olah, Maryam Abdi-Oskouei, Steve Herbener, Xin Zhang, Yannick Trémolet

JEDI contributors: Andrew Collard, Ben Johnson, BJ Jung, Chris Harrop, Clara Draper, Cory Martin, David Davies, David Rundle, Dom Heinzeller, Emily Liu, François Vandenberghe, Guillaume Vernières, Hailing Zhang, Hui Shao, Jonathan Guerrette, Junmei Ban, Marcin Chrust, Marek Wlasak, Mariusz Pagowski, Ming Hu, Rahul Mahajan, Ricardo Todling, Sarah King, Scott Gregory, Sergey Frolov, Steve Sandbach, Steve Vahl, Travis Sluka, Yali Wu, Yanqiu Zhu…

JEDI collaborators: Anthony Weaver, Chris Snyder, Dale Barker, Daryl Kleist, Jeff Whitaker, Nancy Baker, Ron Gelaro, Tom Auligné

Representing: JCSDA, NOAA/EMC, NOAA/ESRL, NASA/GMAO, NRL, USAF, NCAR, UKMO, ECMWF, CERFACS

And about 80 padawans who attended two JEDI Academies

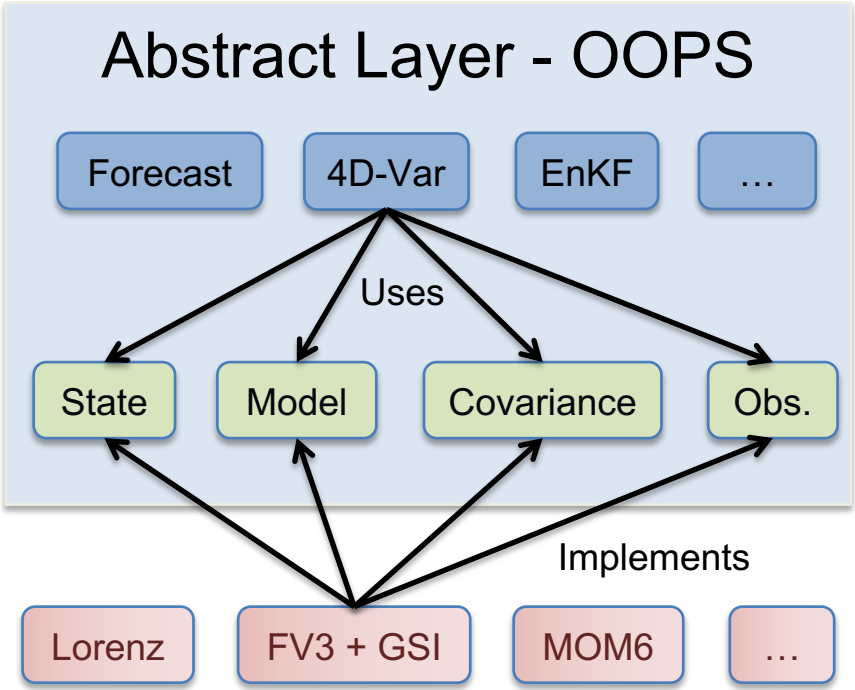
Object Oriented Prediction System (OOPS)

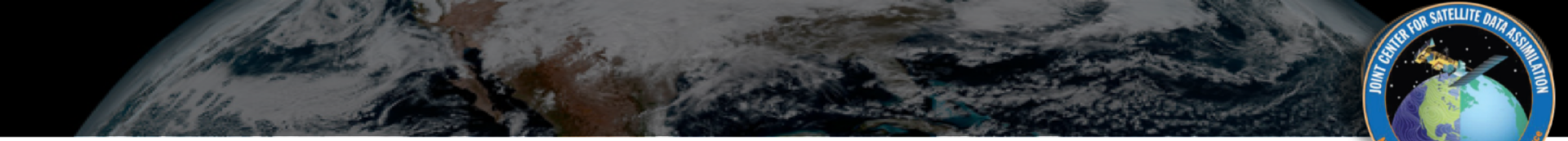
Generic, portable, model-agnostic DA system

Use **object-oriented** and **generic** programing

Each model implements pre-defined abstract interfaces

Separation of concerns



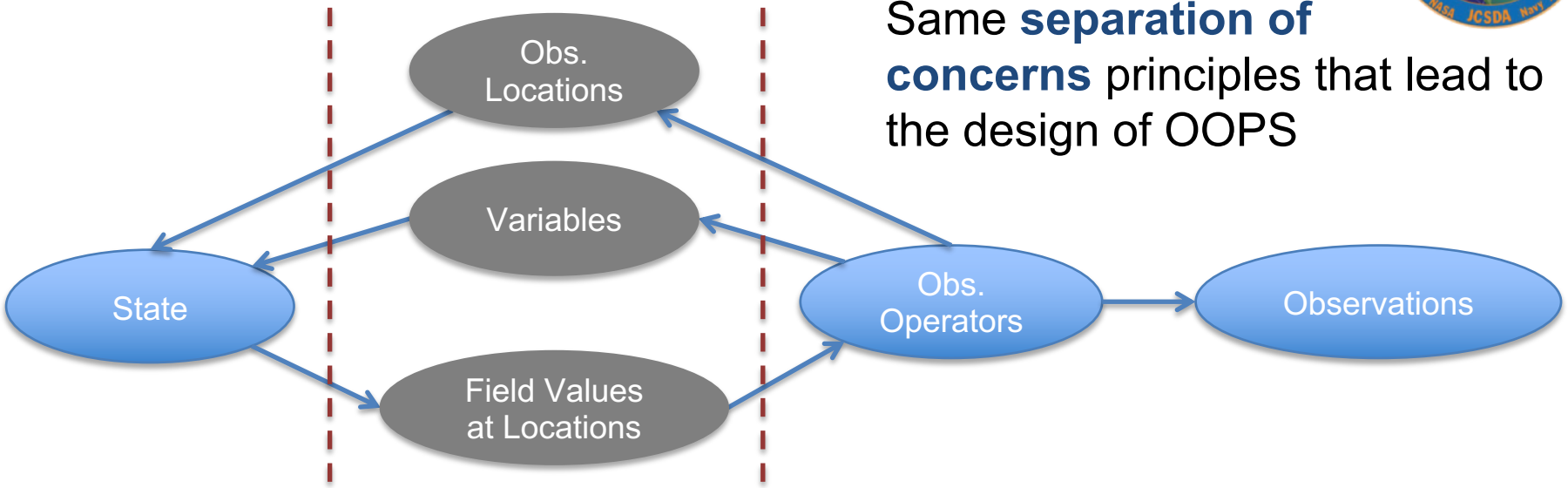


JEDI Observations Interfaces

Unified Forward Operator (UFO)

- Share observation operators between JCSDA partners and reduce duplication of work
  - Taking the model agnostic approach one level down into the observation operators
- Faster use of new observing platforms
  - Regular satellite missions are expensive
  - Cube-sat have short expected life time
  - Include users and instruments science teams
- Unified Forward Operator (UFO)
  - Build a community *app-store* for observation operators

Unified Forward Operator (UFO)



Same **separation of concerns** principles that lead to the design of OOPS

JEDI/UFO introduces standard interfaces between models and observations

Observation operators are independent of the model and can easily be shared, exchanged, compared

UFO Observation “filters”

- Abstract “observation filters” are called before and after the obs. operator
- Observation filters are generic and have access to
  - Observation values and metadata
  - Model values at observations locations (GeoVaLs)
  - Simulated observation value (for post-filter)
  - Their own private data
- Filters are written once and used with many observation types
- Generic filters already exist for:
  - Gross error check, background check, blacklisting, thinning
  - Entirely controlled from yaml configuration file(s)
- More filters will be developed as needed
  - Generic filters should cover most needs

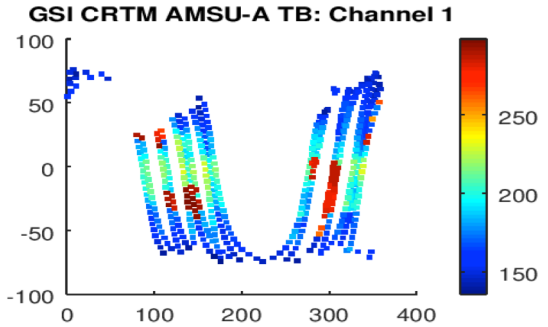
Observation Operators Status

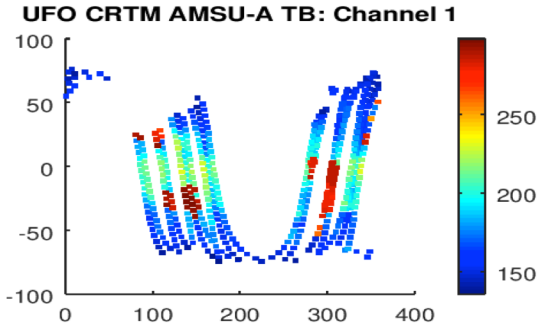
Interpolation from model native grids to observation locations

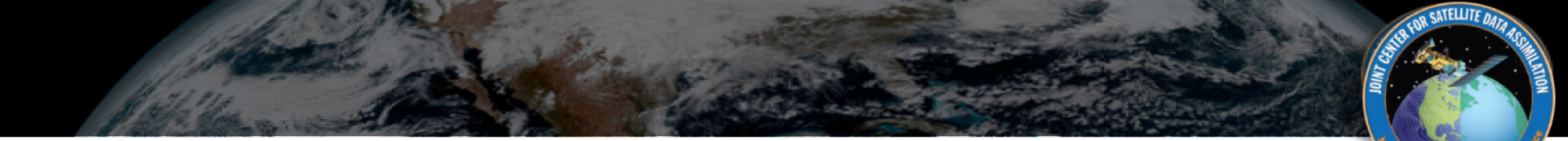
Interface classes (Locations, Variables, GeoVaLs)

Growing subset of instruments

	Nonlinear	Linear (tangent)	Adjoint
Radiosonde	✓	✓	✓
Aircraft	✓	✓	✓
AMVs	✓		
Satellite Radiances	✓	✓	✓
GNSSRO Refractivity	✓	✓	✓
GNSSRO Bending Angle	✓	✓	✓
Aerosol Optical Depth	✓	✓	✓

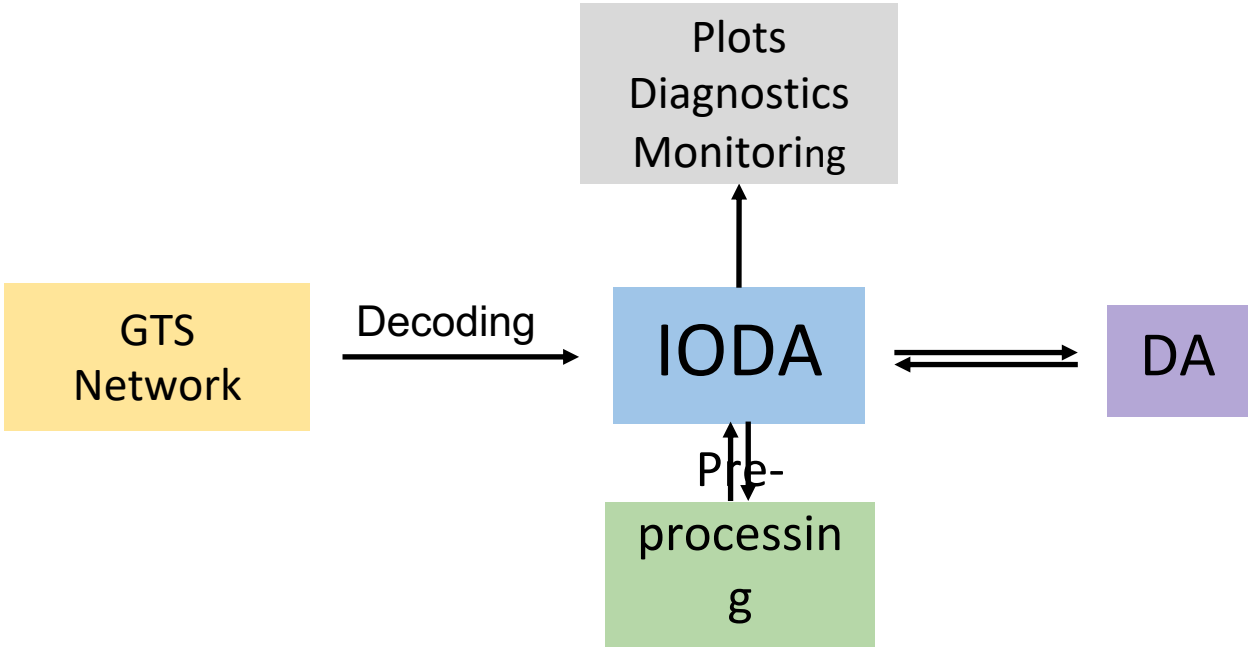






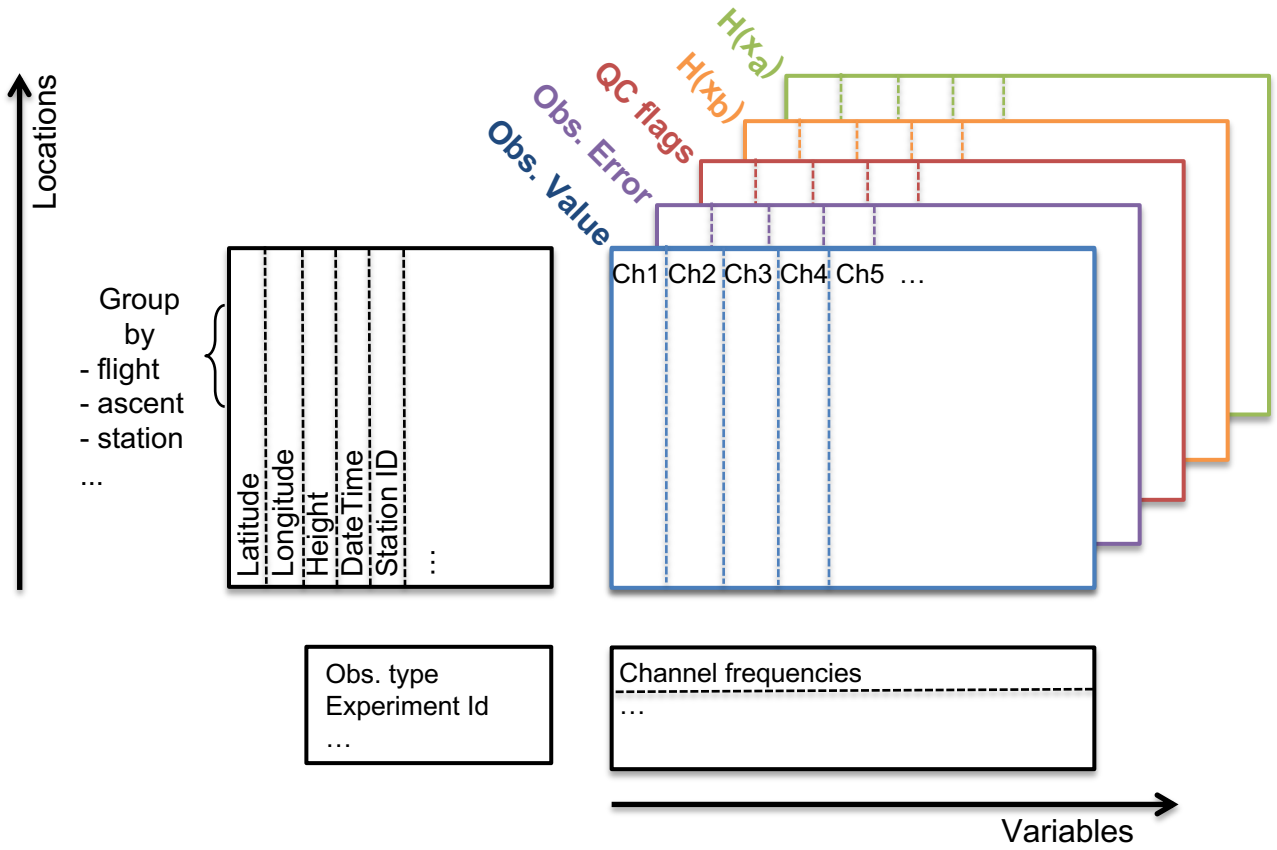
JEDI Observations Handling

Observation Data Flow with IODA



One observation data handling interface across the whole NWP chain, ideally across the entire memory hierarchy (memory, disk, archive)

Access to Observations

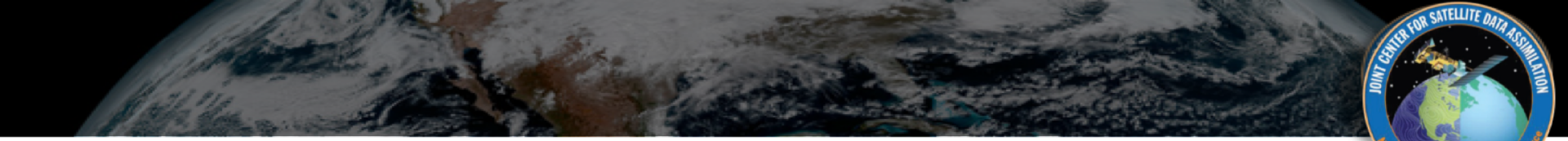


Uniform view and interface for observation data access

Abstract representation, not necessarily actual storage structure

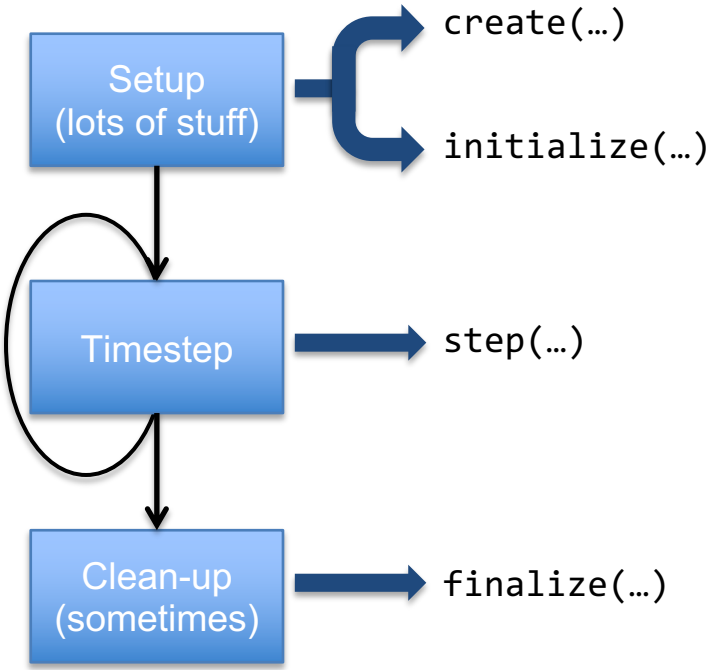
Generalize to all levels of memory hierarchy?

Many (!) years in DA without knowledge of file format/structure



JEDI Model Interfaces

Model design



Between model “steps” OOPS calls post-processors

- OOPS manages when post-processors are called
- Post-processing removed from model code (**separation of concerns**)

Post-processors isolate data assimilation from the model (**separation of concerns**)

- Computing simulated observations H(x)
- Jc-DFI, …

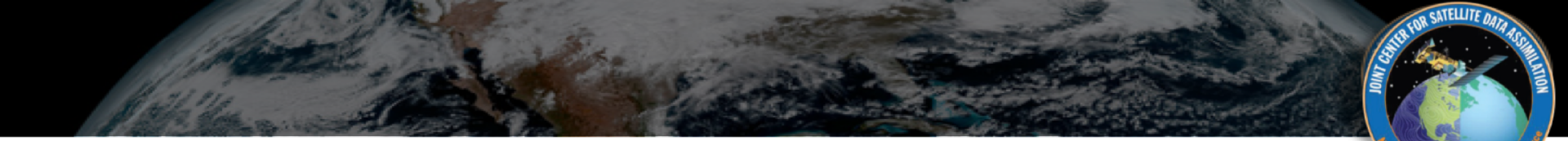
Post-processors do not modify the State

Models Interfacing Status

	State	3D H(x)	M(x)	4D H(x)	3D-Var	TL/AD	4D-Var
FV3-GFS (NOAA)						(dry)	
FV3-GEOS (NASA)							
MPAS (NCAR)						N/A	
WRF (NOAA/NCAR)							
LFRic (UKMO)							
NAVGEM (NRL)							
NEPTUNE (NRL)							
CICE5 (JCSDA/NOAA)						N/A	
MOM6 (JCSDA/NOAA)						N/A	

= technically working = in progress

Coding started in August 2017



JEDI Working Practices

Many people, many organizations, many models

How is fast progress possible?

Infrastructure, working practices

Project methodology inspired by Agile/SCRUM

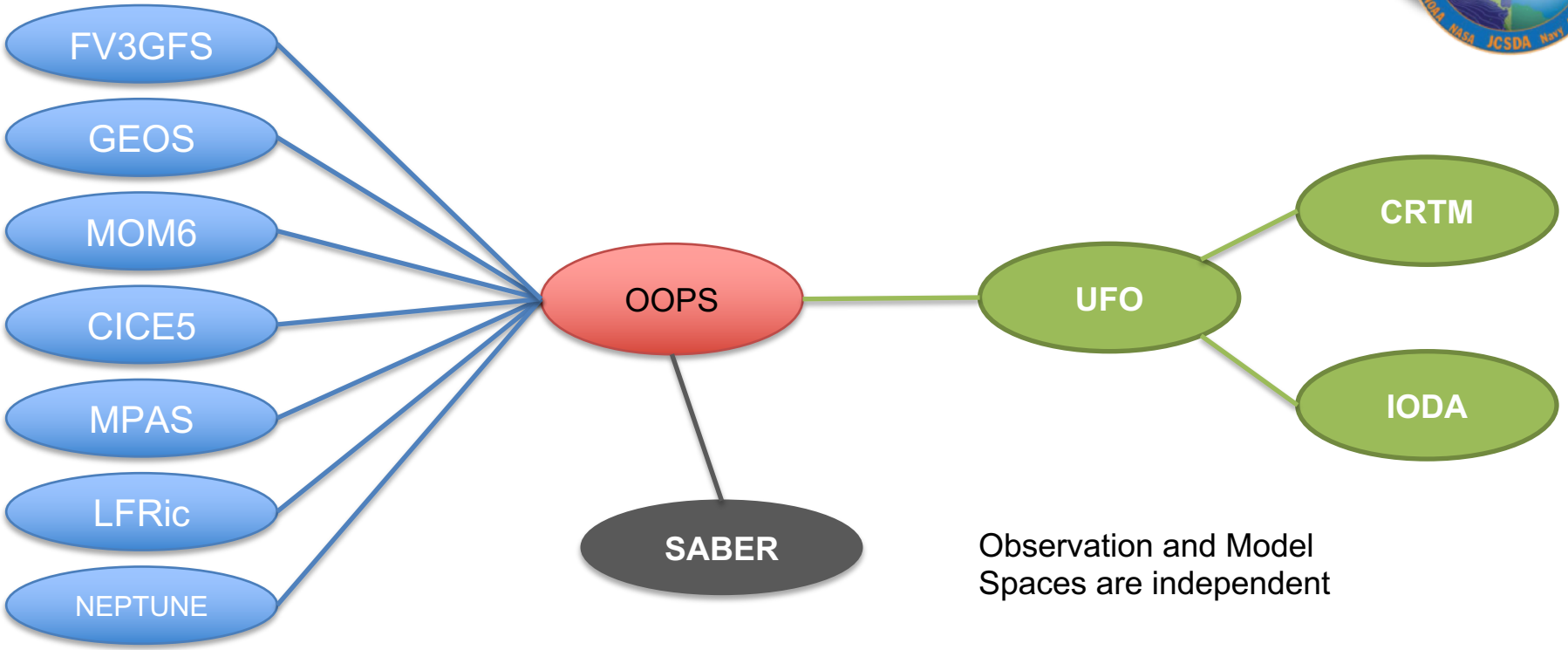
- Adapted to distributed teams and part time members

Collaborative environment

- Easy access to up-to-date source code (github)
- Easy exchange of information (zenhub)
- Regular meetings by video
- Flexible build system (ecbuild, cmake-based)

Object-oriented programming and independence of code components (separation of concerns)

Code and repositories



The end of the monolithic gigantic jumble of code

Infrastructure, working practices

- Enforce software quality
  - Correctness, coding norms, efficiency
- Continuous Integration, Testing framework
  - Toolbox for writing tests
  - Automated running of tests (on pull requests)
- Effort on portability
  - Automatically run tests with several compilers
  - JEDI available in containers (singularity, charliecloud)
- Documentation
  - Doxygen, sphynx, (readthedocs)

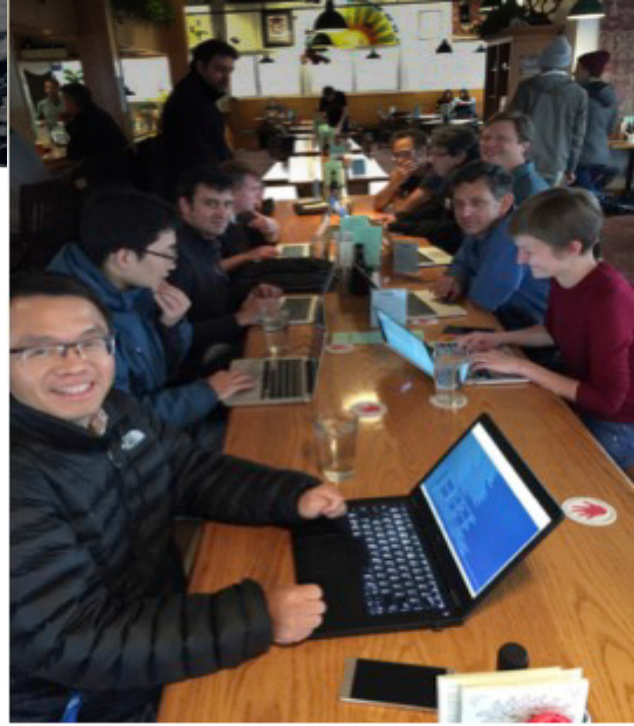
Bringing people in: JEDI Academy

- June 2018 (Boulder), Nov. 2018 (College Park)
- 4 days of lectures and practical sessions with the code

Bringing people in: Code Sprints

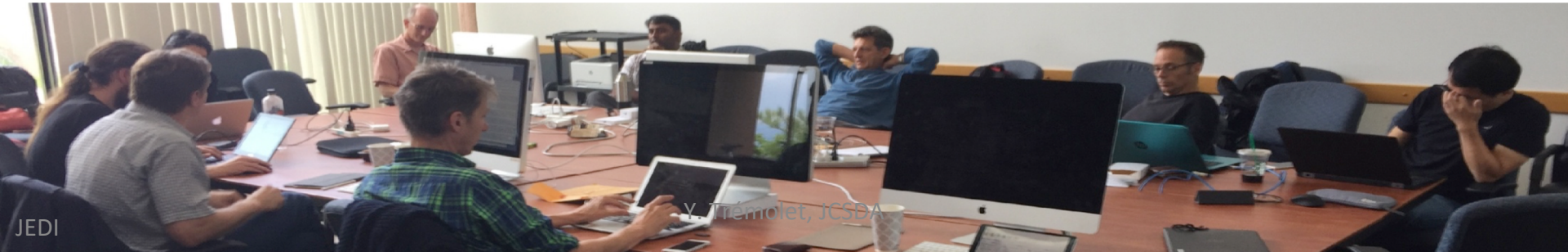
- One specific topic
- Gather 8-10 people in a room for 1 or 2 weeks
- Develop solutions (not a workshop)

- Efficient use of part time contributors
- Involve people from partner institutions
- Very motivating (before, during, after)



NCAR, JCSDA, OAR, EMC, NRL, NASA

JCSDA, NCAR, GMAO, OAR, EMC, Météo-France, Met Office, NRL



Long term collaborations

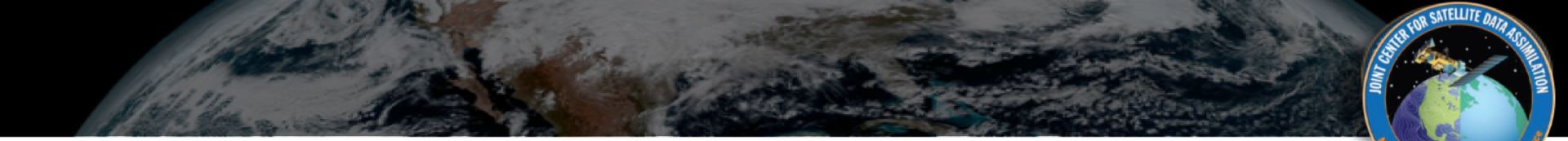
Implies/requires

- Quick code reviews (a few hours to a few days)
- Code is in, doesn't mean it is used in operations

Distinguish steps

- Technical developments
- Scientific evaluation
- Move to operations
- With constant feedback (Agile, DevOps)

JEDI is making DA available to (operational) JCSDA partners.



Final Comments

Final Comments

JEDI is bringing modern software development technologies and working practices to the data assimilation community

- The technologies in use are all proven in the software industry
- Changing working habits/practices is the most challenging aspect, it takes time…

In the future joint data assimilation environment:

- Technical infrastructure is shared as much as possible
- Components (**H**, **B**, **R**…) are made available to all the partners when/where it makes sense
- Each partner keeps their own applications and choice of data assimilation algorithm

The keys to success are **separation of concerns** and **interfaces**