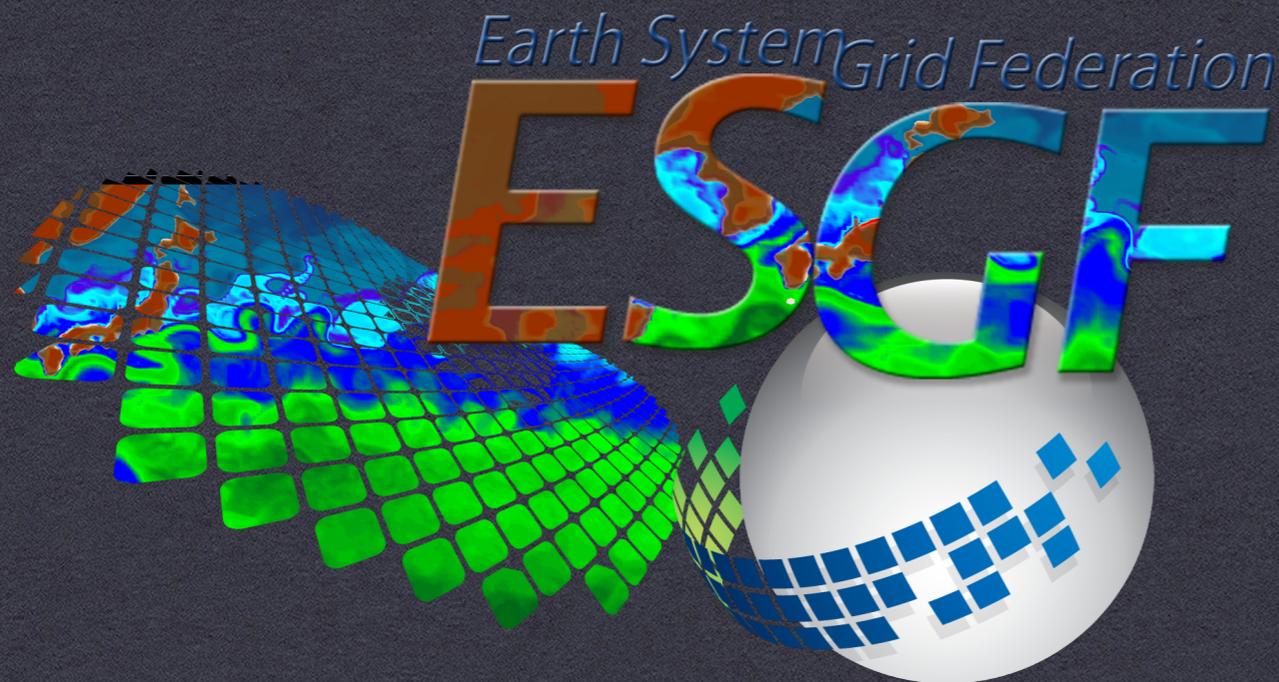




**Jet Propulsion Laboratory**  
California Institute of Technology



**DREAM**

Distributed Resources for  
ESGF Advanced Management

# THE MAGIC OF DREAMS

(... OR “ON THE USE OF CONTAINERIZATION TECHNOLOGIES IN THE DISTRIBUTED RESOURCES FOR THE ESGF ADVANCED MANAGEMENT (“DREAM”) PROJECT, DISCUSSED DURING THE MIDDLEWARE AND GRID INTERAGENCY COORDINATION (“MAGIC”) TEAM MONTHLY MEETING)

LUCA CINQUINI

NASA JET PROPULSION LABORATORY AND CALIFORNIA INSTITUTE OF TECHNOLOGY  
ON BEHALF OF THE ESGF AND DREAM COLLABORATION

© 2018 CALIFORNIA INSTITUTE OF TECHNOLOGY. GOVERNMENT SPONSORSHIP ACKNOWLEDGED

# ESGF Overview

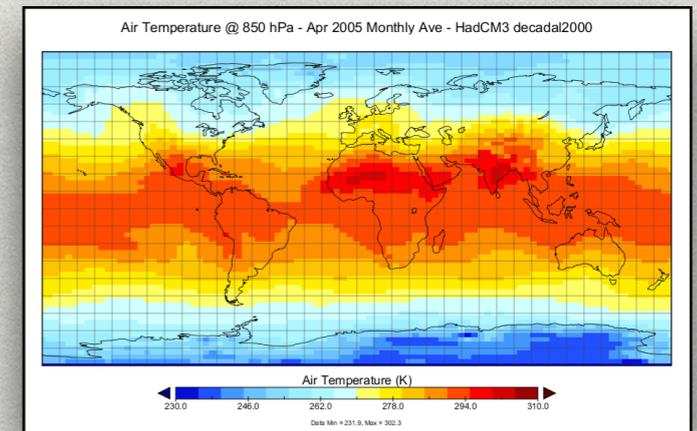
- \* ESGF is an international collaboration of climate centers working together to manage and provide access to climate data - models and observations
- \* Started more than a decade ago, now the world premier technology infrastructure in support of climate science
- \* Spanning several tens of institutions in Europe, North America, Australia and Asia
- \* Funding from DOE, NASA (U.S.), Copernicus (EU), NCI (Australia), CRIM (Canada)
- \* Winner of the 2017 “R&D 100 Award” - prestigious conference that every year recognizes the top 100 most innovative products in software, science and technology



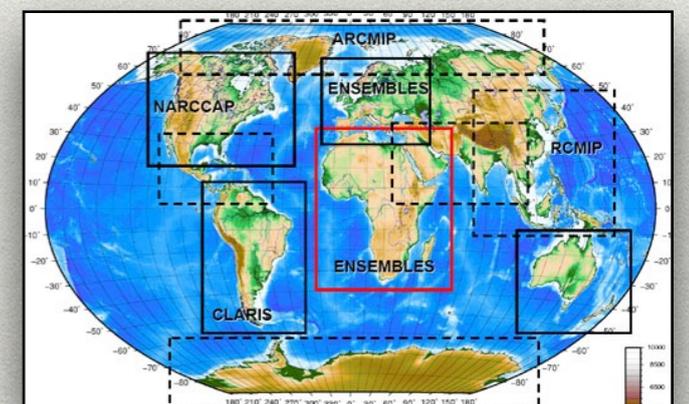
# Current and Future Data Holdings

- \* ESGF hosts some of the most prominent data collections for climate change research:
  - \* **CMIP3, CMIP5** (“Coupled Model Inter-Comparison Project”): output of global climate models used for periodic IPCC assessment reports on climate change
  - \* **CORDEX** (“COordinated Regional climate Downscaling EXperiment”): output of regional climate models, grouped by domain (N. America, Europe, Antarctica, etc.)
  - \* **Obs4MIPs** (“Observations for Model Inter-Comparison”): observational data from NASA, ESA, etc. formatted to look like climate model output
  - \* **Ana4MIPs** (“ReAnalysis for Model Inter-Comparison”): re-analysis data formatted like model output
  - \* Many other MIPs: **TAMIP, GeoMIP, DCMIP, ...**
  - \* Upcoming **CMIP6**: 25-40 PB of uncompressed data replicated at 4 “Tier-1” Nodes

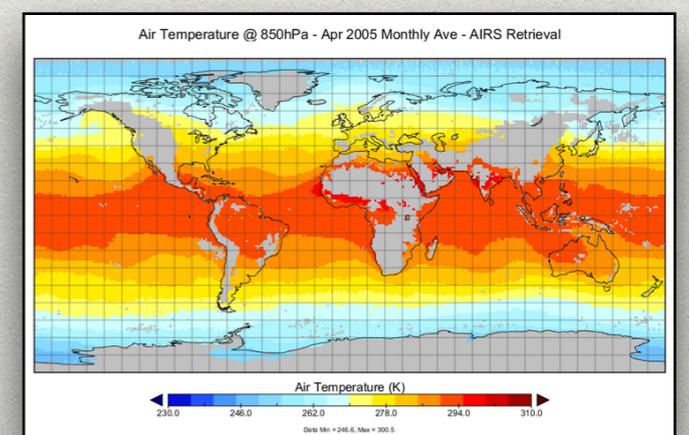
HadCM3



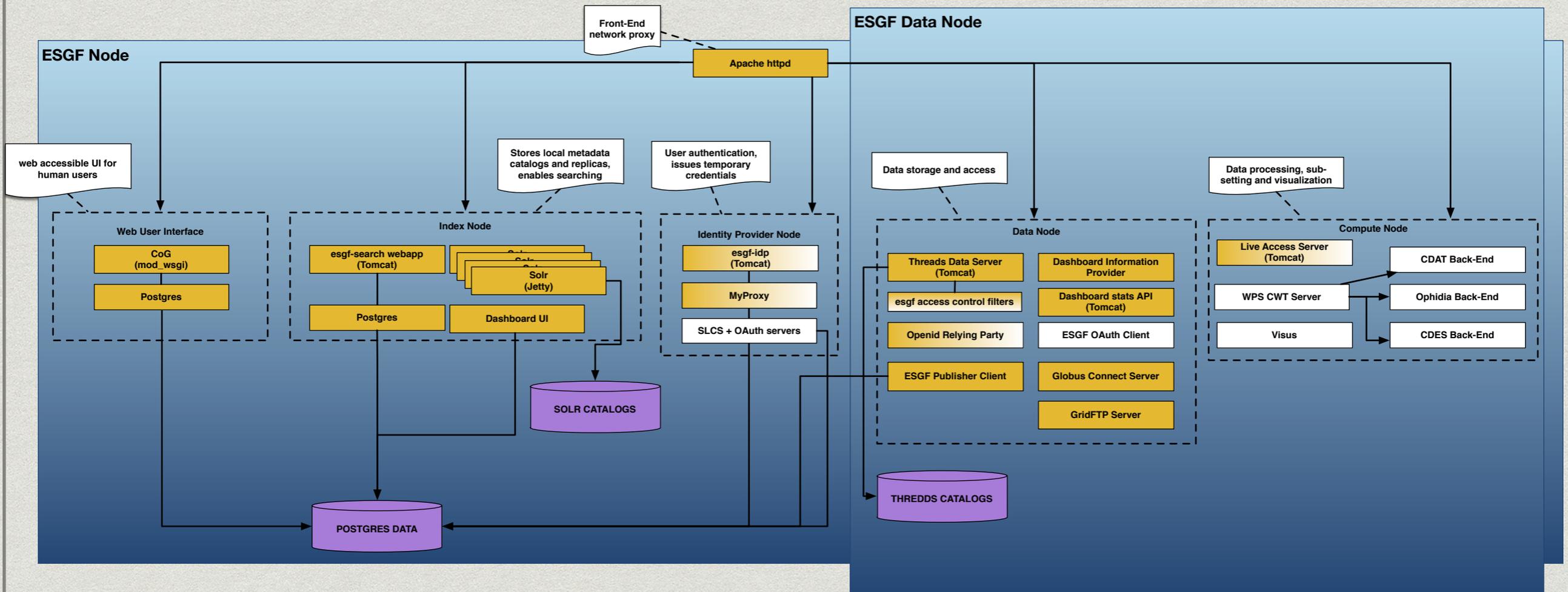
CORDEX



Obs4MIPs



# Software Stack

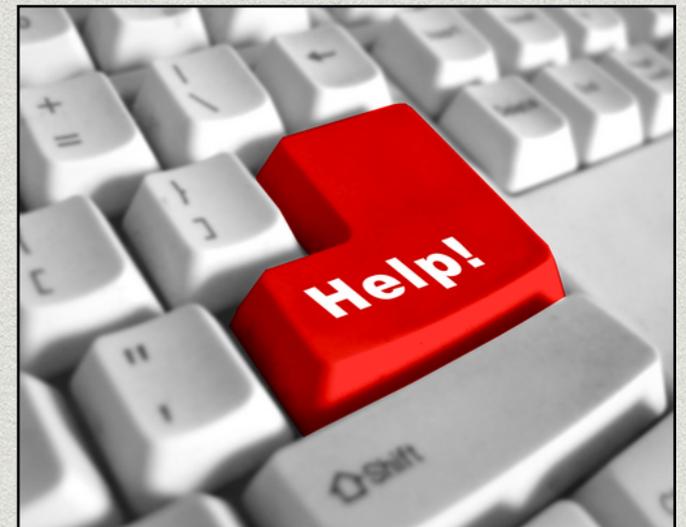


- \* Integration of several components from the open source community, climate applications, and developed by ESGF
- \* Services are grouped into 5 categories according to functionality: User Interface, Index Node, Identity Provider, Data Node, Compute Node
- \* Overall, a very large software stack that is deployed at each Node in the federation

# Some Problems with current ESGF System

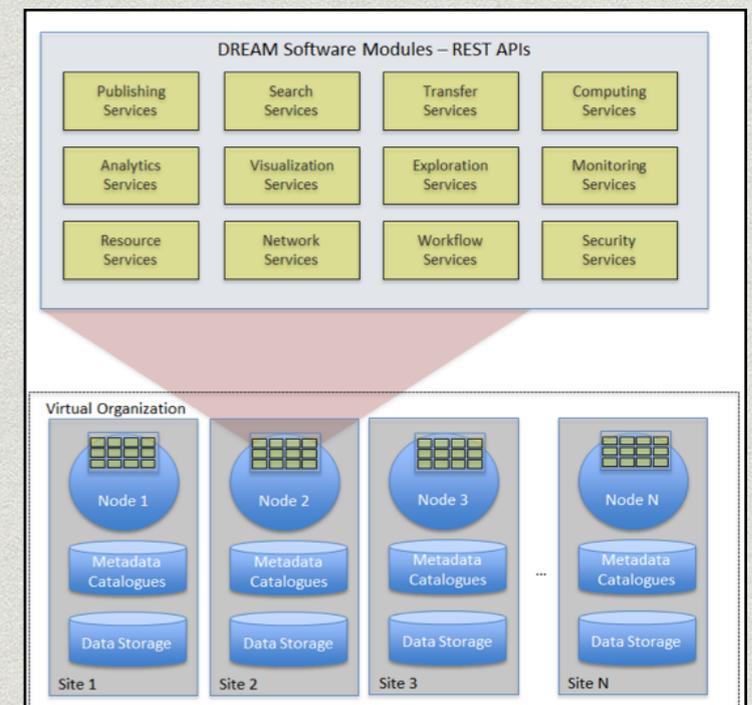
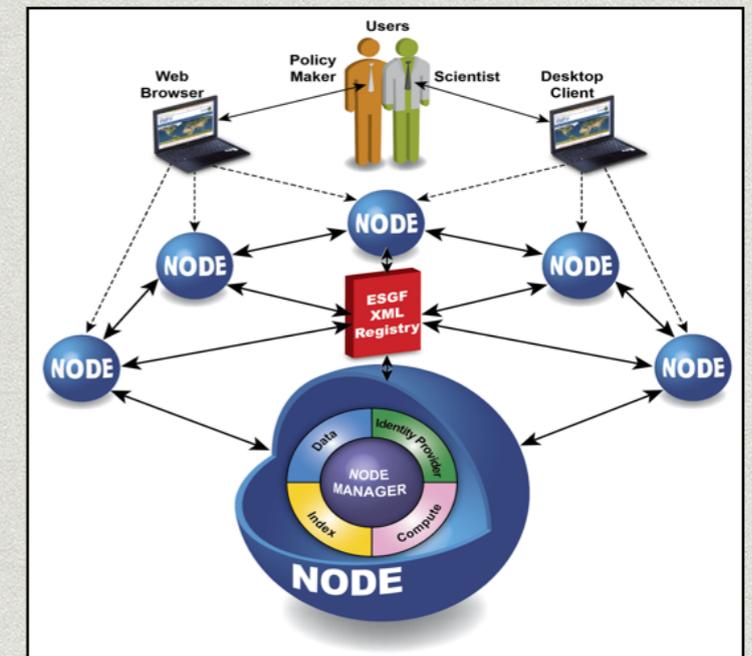
---

- \* Difficult to install and upgrade, impossible to roll back
- \* Installation driven by a gigantic, unmaintainable shell script
- \* Architecture is not very flexible or easy to scale
- \* Because of the installation barrier, it's difficult to add new software modules and functionality



# DREAM: Distributed Resources for the ESGF Advanced Management

- \* Goal: design and implement the next generation ESGF architecture: more modular, scalable, easier to install and upgrade, applicable to other science domains
- \* Strategy: modularization and interaction via RESTful APIs
- \* Implementation: use containerization technologies to design a new ESGF architecture based on micro-services

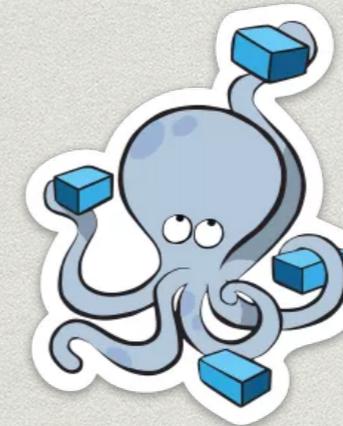


# Containerization Technologies

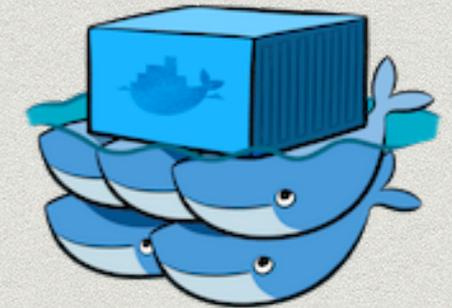
- \* Docker to containerize each ESGF service



- \* Docker Compose to prototype service interaction on single host



- \* Docker Swarm and Docker Stack to deploy all ESGF services on multiple hosts



- \* Kubernetes, as alternative orchestration engine on multiple hosts



- \* Helm for configuring, packaging and deploying Kubernetes resources



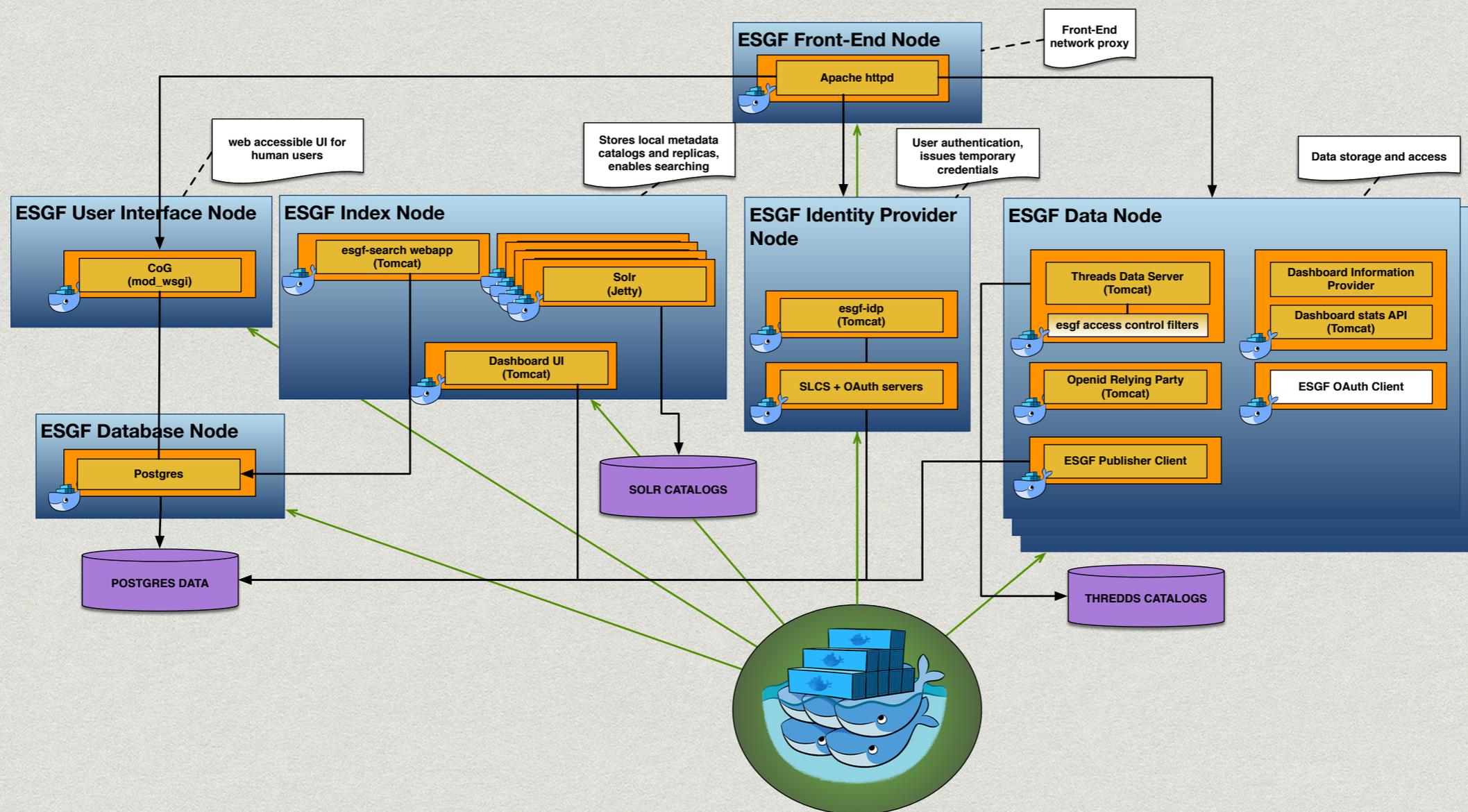
- \* OpenShift Origin, as prototype Platform-As-A-Service for managing the K8s cluster



Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

# ESGF/Docker Architecture w/ Swarm

- \* All ESGF services packaged as Docker containers
- \* Services deployed on hosts of specific type using metadata labels



Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

EC2 Management Console Visualizer

airborne-elb1.dyndns.org:8080

Apps Saddle Ridge My AWS CO2 COG JPL EDNR ESGF tmp Docker CE-ACCE DREAM NOAA VIFI Ski Other Bookmarks

● ip-172-20-0-... manager 31.423G RAM  
 ● ip-172-20-12... worker 31.423G RAM  
 ● ip-172-20-2... worker 31.423G RAM  
 ● ip-172-20-7... worker 31.423G RAM  
 ● ip-172-20-7... worker 31.423G RAM

esgf\_front\_no... esgf\_index\_no... esgf\_data\_no... esgf\_idp\_node... esgf\_db\_node...  
 esgf\_solr\_nod...

- esgf-stack\_esgf-httpd**  
 Image : esgf-httpd:1.4@sha256:a72c...  
 tag : 1.4@sha256:a72c778465abeb...  
 updated : 1/12 12:57  
 32928d77c2db5d3af89192f40aa261...  
 state : running
- esgf-stack\_visualizer**  
 Image : visualizer:stable@sha256:bc...  
 tag : stable@sha256:bc680132772c...  
 updated : 1/12 12:57  
 732ba714fd4fb40c892dd681a78ded...  
 state : running
- esgf-stack\_esgf-cog**  
 Image : esgf-cog:1.4@sha256:f02770...  
 tag : 1.4@sha256:f0277064d8bd9826...  
 cmd : airborne-elb1.dyndns.org,true...  
 updated : 1/12 12:58  
 3406f4f922f36a0ac3a4d5394da5546...  
 state : running
- esgf-stack\_esgf-auth**  
 Image : esgf-auth:1.4@sha256:4168...  
 tag : 1.4@sha256:4168c95884b99ef...  
 updated : 1/12 12:57  
 7f06c2c4ee31a233fec6682b7f43e21...  
 state : running
- esgf-stack\_esgf-solr**  
 Image : esgf-solr:1.4@sha256:f88b04...  
 tag : 1.4@sha256:f88b0c9eae684d9f...  
 updated : 1/12 12:57  
 3d77632d296b602aaa378d6f6cb9e3...  
 state : running
- esgf-stack\_esgf-index-node**  
 Image : esgf-index-node:1.4@sha256...  
 tag : 1.4@sha256:841fec505c83f448...  
 updated : 1/12 12:57  
 6091cca9aaf063520e2cc4e0888b22f...  
 state : running
- esgf-stack\_esgf-orp**  
 Image : esgf-orp:1.4@sha256:fe7720...  
 tag : 1.4@sha256:fe772068944fd8b3...  
 updated : 1/12 12:57  
 9fa46c021fb2a615fc652849186d800...  
 state : running
- esgf-stack\_esgf-dashboard**  
 Image : esgf-dashboard:1.4@sha256...  
 tag : 1.4@sha256:82fcd17f882ec864...  
 updated : 1/12 12:58  
 2bc138a8765bd05f1e30d55cd3119e...  
 state : running
- esgf-stack\_esgf-publisher**  
 Image : esgf-publisher:1.4@sha256...  
 tag : 1.4@sha256:e10f2b939cbe29b...  
 updated : 1/12 12:58  
 b01f8828d939051e4baeebe0c9e872...  
 state : running
- esgf-stack\_esgf-tds**  
 Image : esgf-tds:1.4@sha256:d5690...  
 tag : 1.4@sha256:d5690cc4d160125...  
 updated : 1/12 12:58  
 f86bb69ca0b2d0024d941db6cbb5...  
 state : running
- esgf-stack\_slcs-postgres**  
 Image : postgres:latest@sha256:540...  
 tag : latest@sha256:540ddf85a5e25c...  
 updated : 1/12 12:57  
 b377d9277edda85f2a0db13127a7eC...  
 state : running
- esgf-stack\_slcs-nginx**  
 Image : nginx:latest@sha256:b81f31...  
 tag : latest@sha256:b81f317384d73...  
 updated : 1/12 12:56  
 dfc8dde3814c25500b9a5ac1edc975...  
 state : running
- esgf-stack\_esgf-idp-node**  
 Image : esgf-idp-node:1.4@sha256:2...  
 tag : 1.4@sha256:299d9c4cd7cb801...  
 updated : 1/12 12:57  
 683a7b6403b32e4fedbcb4352a55e...  
 state : running
- esgf-stack\_esgf-slcs**  
 Image : esgf-slcs:1.4@sha256:e939e...  
 tag : 1.4@sha256:e939ef48617f7e7b...  
 cmd : -sn,airborne-elb1.dyndns.org,...  
 updated : 1/12 12:57  
 5b9c54408237f91b58466c69fd268c...  
 state : running
- esgf-stack\_esgf-postgres**  
 Image : esgf-postgres:1.4@sha256:8...  
 tag : 1.4@sha256:8421c3c99f49885e...  
 updated : 1/12 12:57  
 dd5971684f1ce14ce4ffe2f607c0df24...  
 state : running

# ESGF/DOCKER/SWARM TESTBED ON AMAZON WEB SERVICES

USING AN ECS (ELASTIC CONTAINER SERVICE) CLUSTER OF 5 NODES

# Advantages of Micro-Services Architecture

---

- \* Easier installation: simply download and deploy the Docker images through Docker Stack or Kubernetes (no more complex compilation and installation at each site; also, everybody runs exactly the same binary)
- \* Easier upgrade: download a new version of the images, start the containers
- \* Rollback an upgrade, if needed
- \* Scalable, as multiple containers on the same host, or on multiple hosts
- \* More flexible architecture options: Complete Node, Data Node, Index Node, ...
- \* Better testing
- \* Deployable anywhere

# **ESGF/DOCKER/KUBERNETES TESTBED ON OPEN-SHIFT CLUSTER**

**USING SINGLE-NODE MINI-SHIFT CLUSTER ON DEVELOPER'S LAPTOP**

# Lessons Learned

---

- \* In most cases, legacy applications are not built to be deployed in a dynamic, distributed environment
  - \* Assume application data are stored in a persistent location (which is *not* the default case with containers or pods, which can be moved from host to host) - must configure Docker or Kubernetes “volumes” to persist data
  - \* Assume static IP addresses/hostnames (while IP addresses and hostnames are assigned dynamically in a containerized architecture) - must configure Docker or Kubernetes “services”
  - \* May use “customized” versions of open source software packages (Postgres, Tomcat, Solr, ...)
- \* Consequently, legacy applications need to be partially re-architected or re-configured to be deployed and scaled as containers
- \* Steep learning curve for application developers and system administrators
  - \* It’s relatively easy to write a Docker file and create a Docker image
  - \* It’s more difficult to run a set of Docker containers with Docker Compose or Docker Swarm, must understand the semantics and capabilities of the framework (especially, networking and persistent storage)
  - \* It’s even more difficult to run them in a Kubernetes cluster



# Best Practices

---

- \* Run a single process per container, whenever possible
- \* Use multi-stage builds to create Docker images with reduced size
- \* Use official Docker images when available (smaller and more secure)
  - \* Tomcat, Solr, http, etc...
- \* Persist application data with Docker Volumes or Kubernetes Persistent Volumes
- \* Use Docker Services, Kubernetes Services or OpenShift Routes to address the applications with persistent endpoints
- \* Run also administration tasks/commands inside containers
  - \* No “residual” OS dependencies
- \* Express dependencies between containers startup
  - \* Use Docker-Compose “dependsOn” semantics
  - \* Or Kubernetes “initContainers” to delay the start of dependent containers



# Conclusion

---

- \* Despite these difficulties, ESGF and DREAM are fully committed to the micro-services architecture model
- \* Goals:
  - \* Release a feature-complete ESGF/Docker architecture by the end of 2018
  - \* Run at least some operational sites with the ESGF/Docker architecture, either in-premise or on the Cloud

