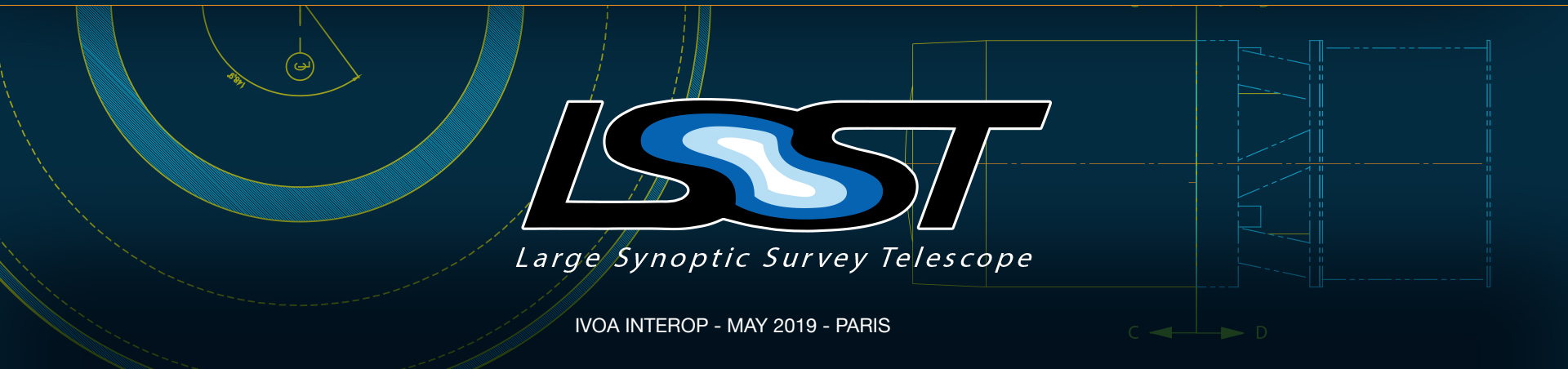
SODA ImgServ for LSST in Python

LSST-DM: Kenny Lo

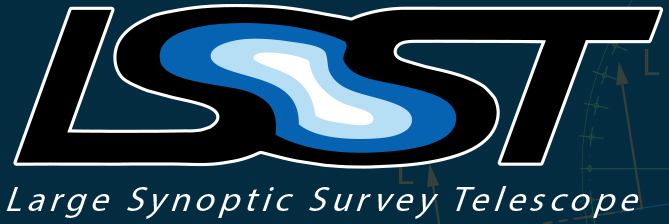


Large Synoptic Survey Telescope

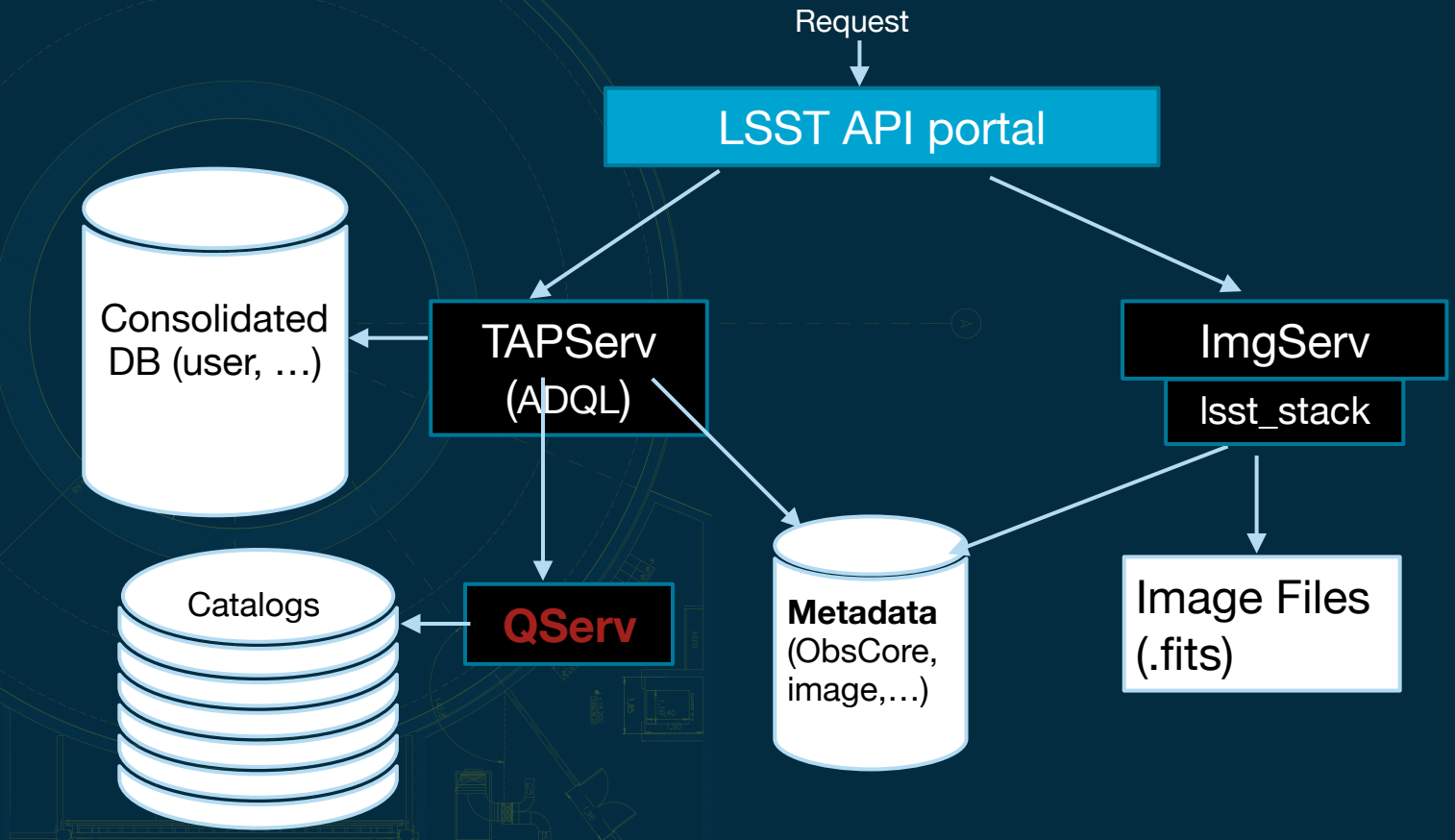
IVOA INTEROP - MAY 2019 - PARIS

LSST ImgServ, aka dax_imgserv

- DAX = Data Access Services in LSST
- LSST image files: 6.4GB ea. (3.2 gigapixel camera)
- Cutout Images for light curves (? Machine learning)
- RESTful APIs with lsst_stack dependencies:
  - afw for image object classes, geometry, WCS, …
  - dm_butler for locating and returning the Python objects of images and their metadata generated from LSST science pipelines, aka data releases.
- Datasets (test): SDSS_Stripe82, WISE_All_Sky, HSC
- ImgServ VO API Interfaces
  1. SIAv2 (or TAP/ADQL?) for search queries
  2. SODA for image cutouts/mosaics/retrieval (LSST v1 - deprecated)



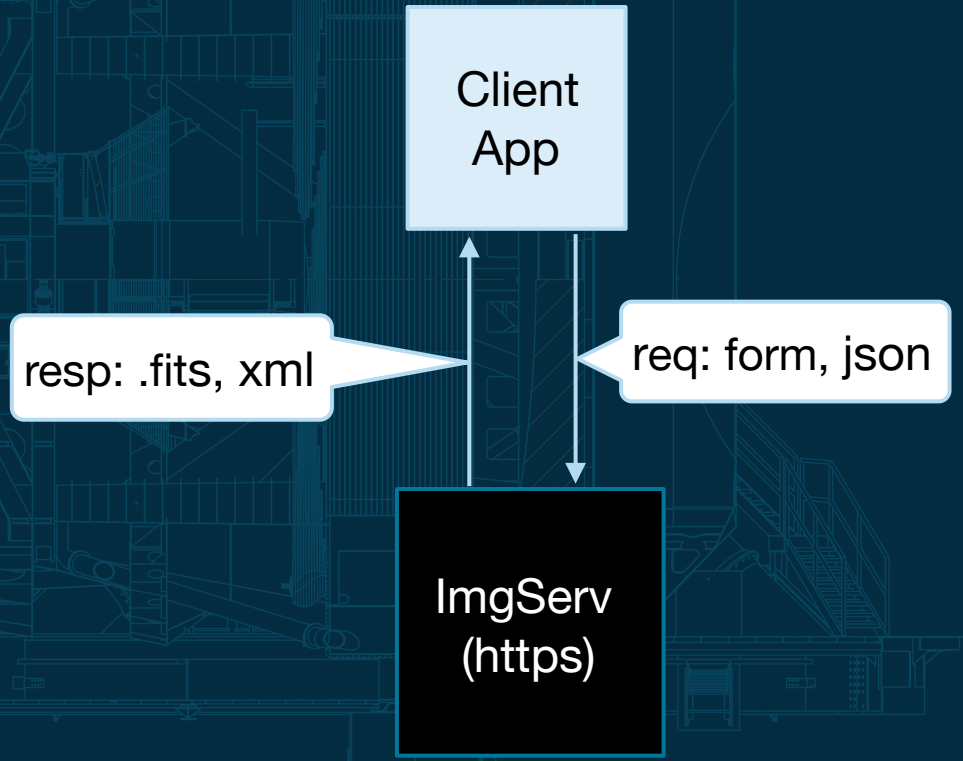
LSST API Aspect Services



Design for Scalability & Performance

App stack:

- Python 3, Flask (CentOS 7) Docker container in Kubernetes (nginx)
- uWSGI for Python app, HTTP server, 40 processes per node (high-end hardware)
- Support for A&A (JWP tokens)
- Distributed key/value store (Redis) for async



Prep for Implementing SODA

Step 1: Attend IVOA Interop Meetings (3rd year)
Step 2: review web service standards (W3C) and tools
Step 3: Read IVOA specification documents

A. UCD1+
B. SODA v1
C. SIAv2
D. DALI v1.1
E. VOSI v1.1
F. ObsCore 1.1
G. ……etc….

Step 2: Gather and examine all the XML Schemas referenced in these docs. (MISSING: VOMetadata-v0.1.dtd!)
Step 3: Search for reference implementations
Step 4: Evalulate client-side tools: TOPCAT, astropy (pyvo, astrocut, …)

SODA Service Endpoints

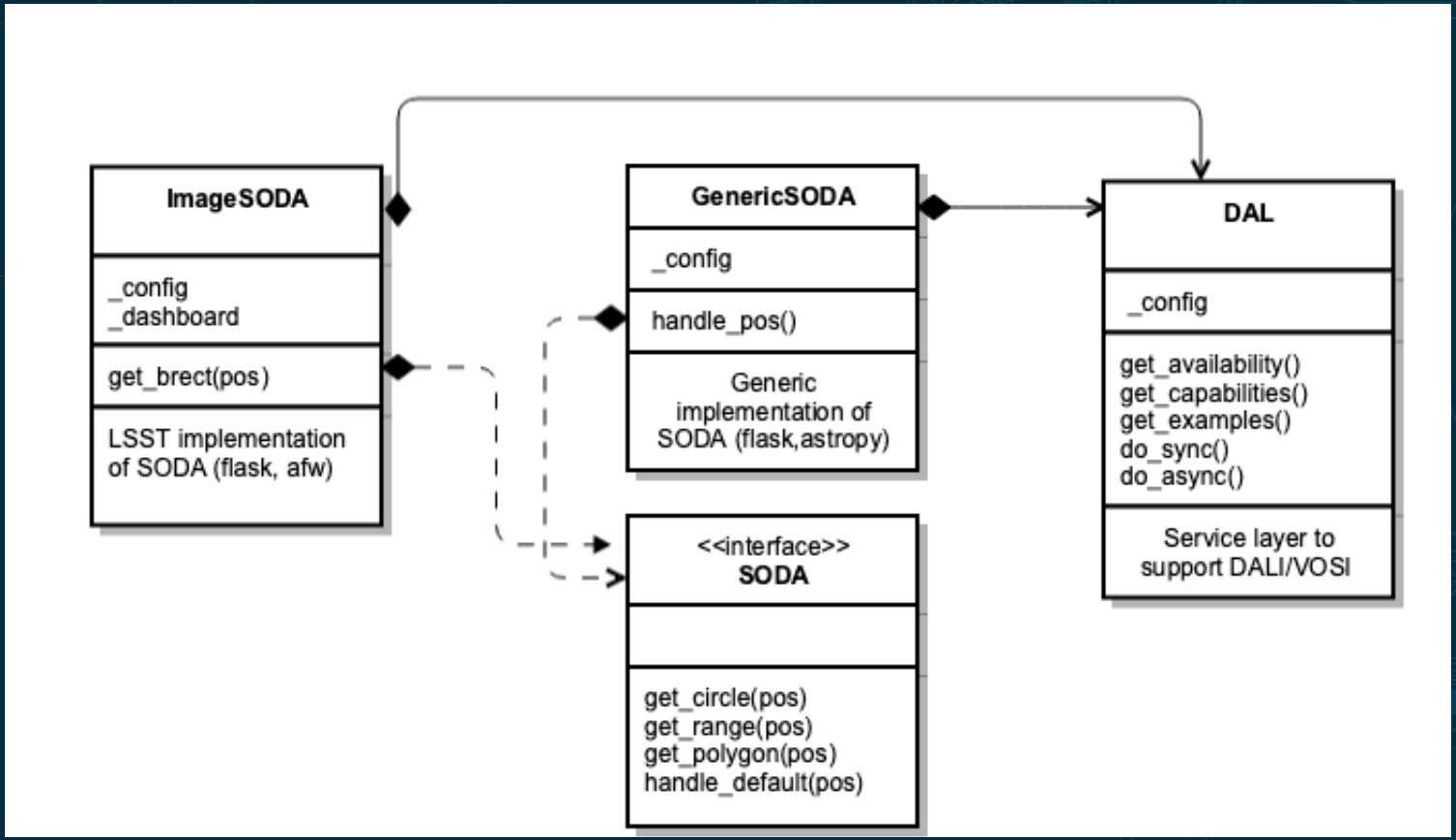
Supported:

- /availability
- /capabilities
- /examples
- /sync
- /async (in progress)
- /sia (in progress)
- /tables(TBD)

Implemented SODA shapes: POS=CIRCLE, RANGE, POLYGON

LSST DAX-defined shape with bounding box (bbox):
BRECT <ra, dec> <w> <h> <unit> (bounding rectangle)

VO Object Diagram



DEMO

SODA client:

-Jupyter notebook in LSST Science Platform (lsst-lsp-int)

ImgServ as SODA service:

-Docker container orchestrated by Kubernetes in lsst-lsp-int.

ImgServ in GitHub

GitHub repo: lsst/dax_imgserv
https://github.com/lsst/dax_imgserv

Build Docker image (~7GB, CentOS, lsst_stack), for example:

```
:~$ cd dax_imgserv
:~$ docker build -t websev/imgserv .
```

Configure settings, then:

```
:~$ docker pull | push | run | exec -it [image ID]
```

Closing Remarks

What would help a new VO implementor?

- streamlined documentation of key VO specs
- centralized and complete schema repo for all VO protocols for validation
- critical: validation tools (TOPCAT, STILTS lib for VOTable)
- reference implementations in GitHub

Generic SODA service in Python? (replacement for butler.get(), astropy for afw, etc).

Contact: kennylo@slac.stanford.edu

THANK YOU