

ESCAPE WP5

Technical Work Review

*Stelios Voutsinas
University of Edinburgh*

Technology Review

- WP5 related tasks:
 - JupyterHub experiments
 - Documentation, Scripts and Configurations
 - Prototype Services being tested
 - Docker Containers for VO tools
 - Can be used to deploy custom JHub environments
 - Used to discover and access data in the Virtual Observatory
 - We've started working on contributions to the ESAP Prototype.
 - First task in progress:
 - Working on allowing keyword-based searches for metadata in Virtual Observatory Services

JupyterHub

- Ongoing experiments and evaluation of running a Jupyterhub (JupyterLab) service in progress
- Service deployed on top of Openstack & Kubernetes
- Output of experiments may be in the form of:
 - Documentation
 - Running prototype service
 - Scripts & Configuration to deploy service on any cluster
 - (Openstack, AWS, GCloud etc..)



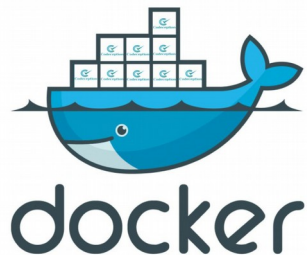
JupyterHub + Kubernetes

- Allows us to create a reproducible service
- Fault tolerant (K8s pods restart automatically when unhealthy)
- Easy to run from an Ops perspective
 - Deploying new versions
 - Platform independent
 - Customizable using Configuration files
- Service deployed using Helm Charts



JupyterHub with Docker

- Our JupyterHub services deploy custom Docker images.
- Configuration allows either of the following:
 - Fetch from Docker Hub
 - Or from local Docker registry.
 - Local Docker Registry works if we want private Docker images



JupyterHub: Authentication

- OAuth
 - Initial experiments work with Github OAuth
 - JHub with EGI-Checkin will also soon be available
- Todo: Work on integration with ESAP
 - Single sign-on
 - Propagation of tokens from ESAP to JHub
 - AuthN/AuthZ between ESAP and a JHub service



JupyterHub Persistent Storage

- Current Jhub Storage experiments with:
 - Openstack Cinder Volumes (block storage)
 - NFS on top of Cinder
- Other experiments with:
 - HDFS for persistent storage prototype in place
 - Data stored as Parquet files
 - Object-store (Openstack Swift) as an alternative also in the works

Notebooks & Containers

- Docker containers for Virtual Observatory
 - Provide set of tools and libs that help users discover, access and visualization data from VO services.
- Also Example notebooks for a few different use cases.
 - Examples for finding accessing and visualizing image or catalogue data
- Libraries used include:
 - Astropy (Astroquery), Pyvo, AladinLite (image visualization), plotly, bokeh (plotting)

Notebooks & Containers

Workflow for discovering, querying and visualizing astronomy data in the VO

```
In [ ]: from astropy.coordinates import SkyCoord
from hips import WCSGeometry, make_sky_image
from hips import HipsSurveyProperties
from ipywidgets import Layout, Box, widgets
```

Data Discovery (VO)

```
In [ ]: import pyvo
```

```
In [ ]: from pyvo.registry import search as regsearch
```

Find all TAP Services with 'quasars'

```
In [ ]: services = regsearch(keywords=['quasar'], servicetype='tap')
```

```
In [ ]: print (services)
```

Find all TAP Services with keyword "ukidss"

```
In [ ]: services = regsearch(keywords=['ukidss'], servicetype='tap')
print (services)
```

Data Access (TAP, Astropy)

```
In [ ]: from pyvo.dal import tap
service = tap.TAPService(tap_url)

query_text = """
SELECT TOP 500
sourceID, ra,dec FROM
    lasSource
ORDER by dec
"""

from astroquery.utils.tap.core import TapPlus
service = TapPlus(url=tap_url)
job = service.launch_job(query_text)
table = job.get_results()
table
```

Image Access

HIPS Image

```
In [ ]: url = 'http://surveys.roe.ac.uk/hips71/LAS/LAS_Y_OUT//properties'
hips_survey = 'CDS/P/DSS2/red'
```

```
In [ ]: print (url)
```

```
In [ ]: hips_survey_property = HipsSurveyProperties.fetch(url)
```

```
In [ ]: geometry = WCSGeometry.create(skydir=SkyCoord(217.9, -2.3, unit='deg', frame='fk5'),width=180,
height=180, fov='0.02 deg',coordsys='icrs', projection='TAN')
from hips import make_sky_image
result = make_sky_image(geometry, hips_survey_property, 'fits')
```

ESAP Prototype: VO Integration

- Plan to assist in the development of the Prototype
- Initial work in building a keyword based search for VO Services

ESAP

Archives Datasets Telescopes Query Settings

ESAP Query

Institute

IVOA

Keyword

2mass

Title

Target

m81

RA (degrees)

1

dec (degrees)

10

search radius (degrees)

10

DataProduct Type

all

DataProduct Subtype

all

Start Date

02 / 07 / 2004

End Date

02 / 08 / 2004

Instrument

all

Create Queries

Clear

Created Queries

	dataset uri	dataset name	service connector	status	query
	vo_reg	HSCv2 TAP	vo_reg_vo_registry_conn..	n/a	http://vo.stsci.edu/hscv2tap/TapService.aspx/sync?LANG=ADOL&REQUEST=doQuery&QUERY=SELECT+TOP+10+*%2A+from+ivoa.observatory+WHERE+target_name%3D%27m81%27+AND+CONTAINS%28POINT%28%27ICRS%27%2Cs_ra%2Cs_dec%29%2C+CIRCLE%28%27ICRS%27%2C1.0%2C10.0%2C10.0%29%29%3D1
	vo_reg	GAVO DC TAP	vo_reg_vo_registry_conn..	n/a	http://dc.zah.uni-heidelberg.de/tap/sync?LANG=ADOL&REQUEST=doQuery&QUERY=SELECT+TOP+10+*%2A+from+ivoa.observatory+WHERE+target_name%3D%27m81%27+AND+CONTAINS%28POINT%28%27ICRS%27%2Cs_ra%2Cs_dec%29%2C+CIRCLE%28%27ICRS%27%2C1.0%2C10.0%2C10.0%29%29%3D1
	vo_reg	6dF DR2	vo_reg_vo_registry_conn..	n/a	http://wfaudata.roe.ac.uk/6df-dsa/TAP/sync?LANG=ADOL&REQUEST=doQuery&QUERY=SELECT+TOP+10+*%2A+from+ivoa.observatory+WHERE+target_name%3D%27m81%27+AND+CONTAINS%28POINT%28%27ICRS%27%2Cs_ra%2Cs_dec%29%2C+CIRCLE%28%27ICRS%27%2C1.0%2C10.0%2C10.0%29%29%3D1
	vo_reg	6dF DR3	vo_reg_vo_registry_conn..	n/a	http://wfaudata.roe.ac.uk/6dfdr3-dsa/TAP/sync?LANG=ADOL&REQUEST=doQuery&QUERY=SELECT+TOP+10+*%2A+from+ivoa.observatory+WHERE+target_name%3D%27m81%27+AND+CONTAINS%28POINT%28%27ICRS%27%2Cs_ra%2Cs_dec%29%2C+CIRCLE%28%27ICRS%27%2C1.0%2C10.0%2C10.0%29%29%3D1
	vo_reg		vo_reg_vo_registry_conn..	n/a	http://wfaudata.roe.ac.uk/twompz-dsa/TAP/sync?LANG=ADOL&REQUEST=doQuery&QUERY=SELECT+TOP+10+*%2A+from+ivoa.observatory+WHERE+target_name%3D%27m81%27+AND+CONTAINS%28POINT%28%27ICRS%27%2Cs_ra%2Cs_dec%29%2C+CIRCLE%28%27ICRS%27%2C1.0%2C10.0%2C10.0%29%29%3D1
	vo_reg		vo_reg_vo_registry_conn..	n/a	http://wfaudata.roe.ac.uk/vhdR4-dsa/TAP/sync?LANG=ADOL&REQUEST=doQuery&QUERY=SELECT+TOP+10+*%2A+from+ivoa.observatory+WHERE+target_name%3D%27m81%27+AND+CONTAINS%28POINT%28%27ICRS%27%2Cs_ra%2Cs_dec%29%2C+CIRCLE%28%27ICRS%27%2C1.0%2C10.0%2C10.0%29%29%3D1

Rows per page: 10

1-6 of 6

< >

ISTRON - version 0.0.4 - 2 apr 2020

Other Technologies used for our Science Platforms

- Science Platforms being built for GAIA & LSST

- Apache Spark & Zeppelin

- We're running a service with Zeppelin as User interface with a Spark Cluster behind it.
- Initial prototype built with Hadoop (HDFS/Yarn)
- We're currently looking into replacing the Hadoop components.
 - Plan is to run Spark on Kubernetes, with Object storage for persistent data



- Apache Kafka

- We're building a platform to receive alerts from LSST (& ZTF), process and annotate them, allowing users to create custom filters on them
- The platform receives the alerts using Kafka, and we also use Kafka internally in our pipeline.
- Allows us to create a scalable service, where we can add worker nodes to consume, process & produce these alerts, (horizontal scaling)



Other Technologies used for our Science Platforms

- Spark & Zeppelin

The screenshot displays the Zeppelin Notebook interface. At the top, the Zeppelin logo is on the left, and a search bar and user profile 'gaiauser' are on the right. The notebook title is '/experiments/stv/stv-hdbscan'. Below the title is a toolbar with various icons for file operations and a 'Head' dropdown menu. The main area shows a code cell with the following Spark code:

```
%spark.pyspark

# define the data frame source on the given column selection/predicates:
df = sqlContext.read.parquet(
    "/hadoop/gaia/parquet/gdr2/gaia_source/*.parquet"
).select(
    ["designation", "source_id", "ra", "ra_error", "dec", "dec_error", "parallax", "parallax_error", "parallax_over_error", "pmra", "pmra_error", "pmdec", "pmdec_error", "l", "b"]
).where(
    abs(b) < 30.0 AND parallax > 1.0 and parallax_over_error > 10.0 AND phot_g_mean_flux_over_error > 36.19 AND astrometric_sigma5d_max < 0.3 AND visibility_periods_used > 8 AND (astrometric_excess_noise < 1 OR (astrometric_excess_noise > 1 AND astrometric_excess_noise_sig < 2))"
)

# sanity check
df.show()
print("Data frame rows: ", df.count())
```

The code cell is in a 'RUNNING' state, indicated by a progress bar at 59%. Below the code, the output shows a table of data frame rows. The table has 13 columns: designation, source_id, ra, ra_error, dec, dec_error, parallax, parallax_error, parallax_over_error, pmra, pmra_error, pmdec, pmdec_error, and l. The output is truncated, showing only the first few rows.

designation	source_id	ra	ra_error	dec	dec_error	parallax	parallax_error	parallax_over_error	pmra	pmra_error	pmdec	pmdec_error	l
[Gaia DR2 40396662...]	[4039666296672658688]	272.42180060744215	[0.040014912399424604]	-33.6058067001021	[0.03368056050988915]	1.422454370334301	[0.03639988616495275]	39.07854	-0.2316953688797376	[0.07537877095615261]	-5.442599071096166	[0.05727019630225524]	358.50569414878646
[Gaia DR2 40396673...]	[4039667327461865216]	272.65050482164776	[0.036197874417770184]	-33.67244054531187	[0.03680133647540661]	1.1673137239965692	[0.05314058363370944]	21.96652	-1.7009801509860536	[0.07029452441830816]	-4.612361250689048	[0.06085092511704771]	358.53690441856696
[Gaia DR2 40396638...]	[4039663822768546432]	272.3987836204827	[0.027632276638390182]	-33.677978944324124	[0.024215370325550427]	1.485547221111856	[0.028698903012696158]	51.763206	-4.537954548081503	[0.053742838174843835]	-15.585752841174445	[0.041699306044425514]	358.4325162212831
[Gaia DR2 40396743...]	[4039674302495016832]	272.6370332645537	[0.049971393914832546]	-33.46345401655811	[0.04311907194434805]	1.4895944391179747	[0.05535383094438027]	26.910412	-0.999004938426811	[0.08901429663519801]	-20.520372339141726	[0.07278953875661565]	358.7173313727555
[Gaia DR2 40396798...]	[4039679804475284224]	272.3352958925287	[0.0382698356897684]	-33.49684273645387	[0.037562716485539]	1.8288406168863958	[0.049118014784077936]	37.2336	-8.709127050643408	[0.07206191453025368]	0.8317008127357967	[0.05431303815619029]	358.568061282215
[Gaia DR2 40396768...]	[4039676845112356736]	272.3423038848421	[0.04004613278050688]	-33.60277422977108	[0.03534959354180454]	1.331757953116398	[0.040852709757309896]	32.59901	-3.0736631636887344	[0.07887016066616566]	-10.535097449648152	[0.06167703651701124]	358.47687324908384

Started a few seconds ago.

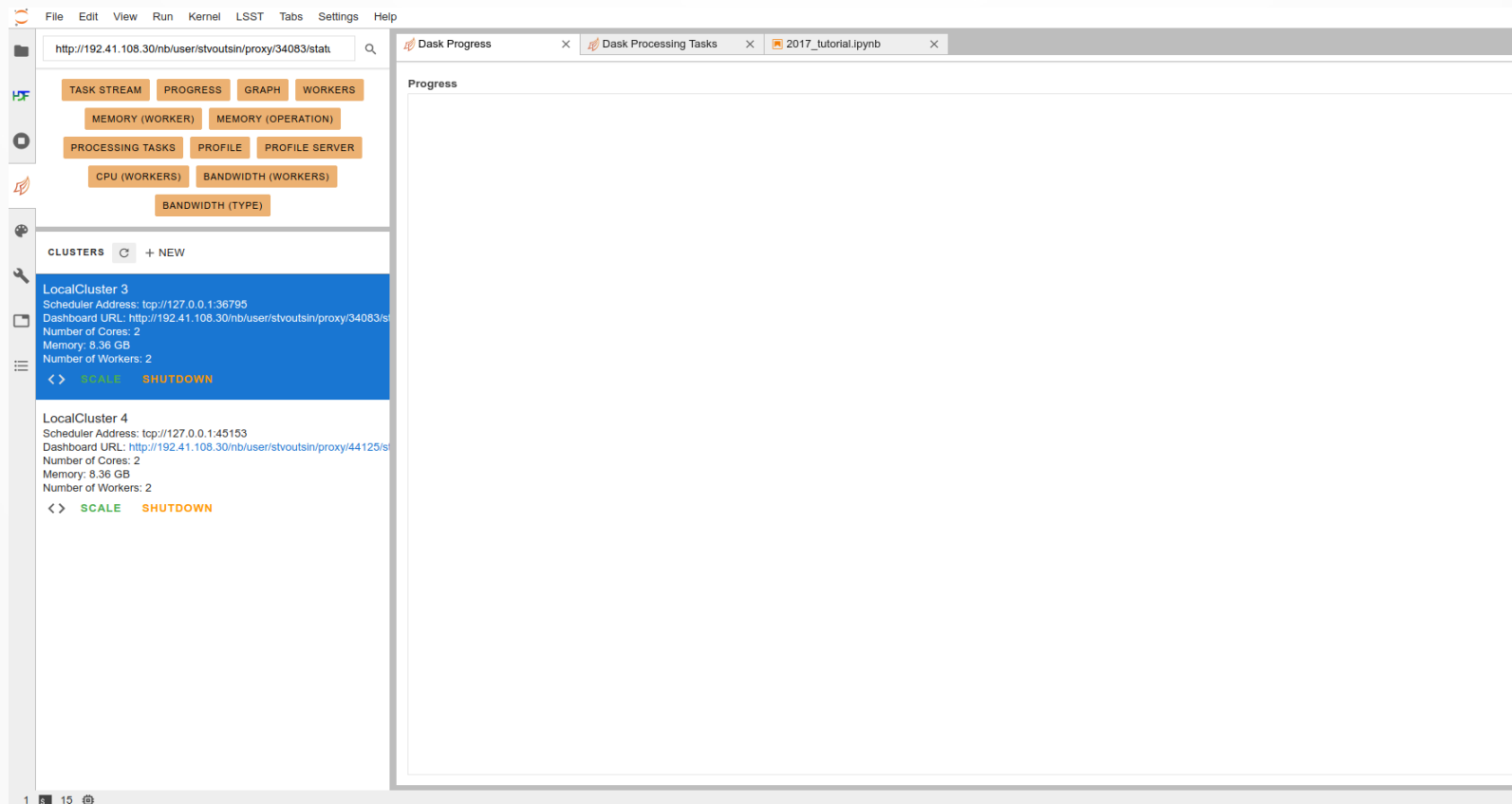
Other Technologies used for our Science Platforms

- Dask
 - Provided as part of one of our experimental JupyterHub services
 - Powerful and flexible tool for scaling Python analytics across a cluster.
 - Uses existing Python structures and APIs (pandas, numpy, etc..)
 - Works out-of-the-box with JupyterHub
 - JupyterLab extension



Other Technologies used for our Science Platforms

- Dask



Technology Review

- Questions?