

Science Platform from the User Perspective

Xiuqin Wu, David Ciardi, & Gregory Dubois-Felsmann And the IPAC SUIT Team

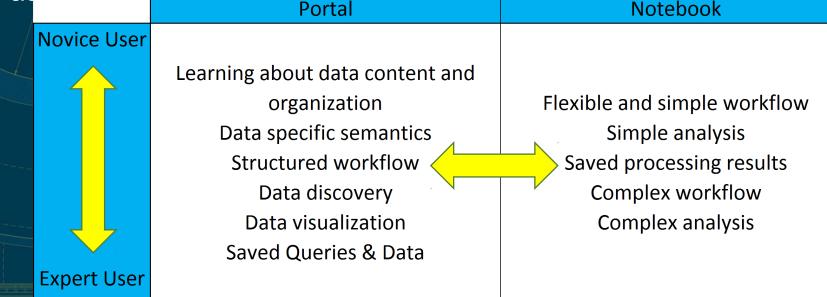
Purpose of the Science Platform



- Enable access to the LSST data products
- Enable visualization and exploration of the LSST data
- Provide an interface to added-value processing and analysis close to the data

Customers of the Science Platform

- LSST Data Rights Scientists, Science Collaborations, and LSST Project
- The Platform needs to be simple enough to engage the general users and flexible enough to meet the needs of the experienced users
- There is a continuum of users, and users will grow and adapt with time.
- Platform needs to enable the users to utilize the contextual information of the LSST data so the users understand how the data are integrated and related



Scientific Priorities for a Minimum Viable System

- Provide a portal with a structured workflow that utilizes the LSST semantic contextual information about the data to enable users to explore and understand the data
- Provide a flexible computational environment (Python Notebook) that has access to the Data Products and the computational infrastructure
- Enable the users to move between the portal and the flexible environments as their needs dictate

Vision of the Science Platform

Portal

- Enables discovery and exploration of the LSST data with structured workflows.
- Contextual information about the semantics of the LSST data is used to guide the workflow and the users.

Make it easy to share, analyze, and visualize results between the structured workflow environment and the flexible workflow environment

Notebook

- Enables exploration and analysis of the LSST data with flexible workflows.
- Environment has access to tools developed by LSST as well as outside tools

Unifying Infrastructure

The Aspects of the Science Platform

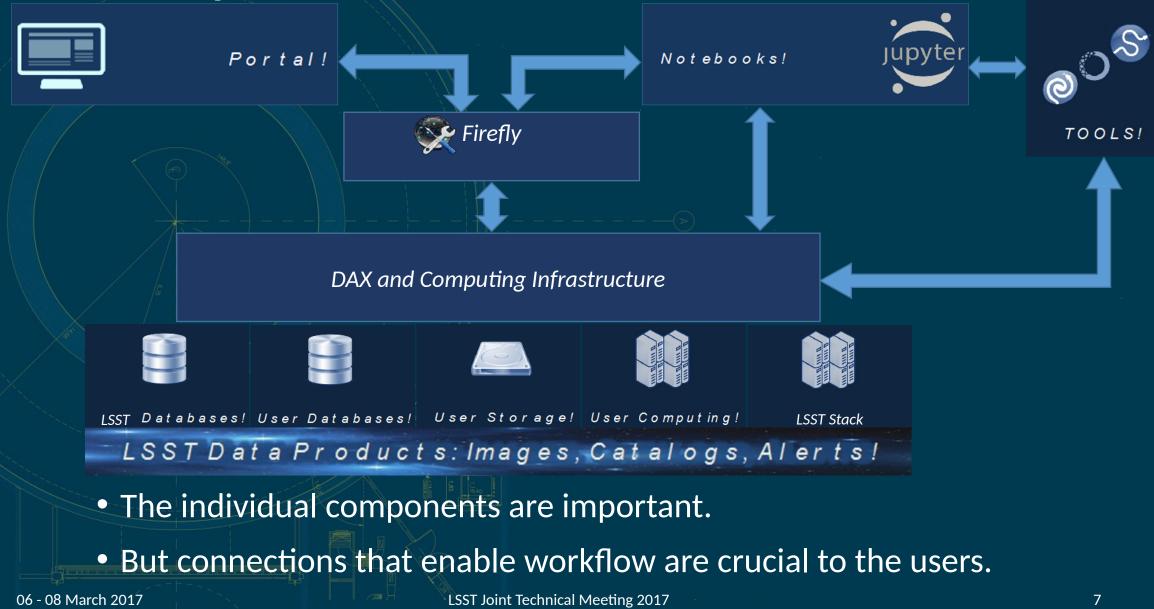
- The Science Platform concept rests on the provision of two* windows into the LSST data products, storage, and computing: the Portal and the Notebook
 - Both rest on a foundation of the DAX services for data access, and the NCSA provisioning services and identity management for flexible computing and storage access
 - Both are client-server web applications
 - Asynchronous DAX services and user identity management allow queries and results, and user data, to be shared between the two
 - The common use of web technologies allows components to be used equally well in either environment

Portal	Notebook
Firefly JavaScript App	Jupyter front end
Firefly servers and load balancer	Jupyter and JupyterHub servers
Python servers for extensions	IPython kernel servers
	Flexible computing for parallelized work

• "Aspects" emphasizes the underlying unity of the system

* There is and present a structure as the set of the se

The Aspects of the Science Platform



The Portal Aspect of the Platform



- Data Discovery lays out the full breadth of the LSST data with documentation
- Data Connections presents the semantic links between data
- Data Query supports both basic (UI-driven) and ADQL/SQL query building
 - Provides access to both the LSST data and standards-compliant external data
- Data Visualization provides LSST-aware visualizations
- Exploratory Data Analysis brushing and linking, filtering, synthetic columns, histogramming, basic astronomical tools such as time-series analysis
- Client-server model enables scalable handling of large result sets
- User Data Access enables read-write use of workspace and Level 3 Data Products
- Connection to Python uses the Science Platform's computing resources to enable LSST-specific extensions to the core Firefly capabilities
 - Example: the visualization of the LSST footprint / de-blending model
- Provides structured environment for the novice
- Provides a starting point for the expert, with a seamless transition to interactive analysis in the Notebook Aspect
- Dedicated portals support Observatory processes (QC/QA, commissioning)

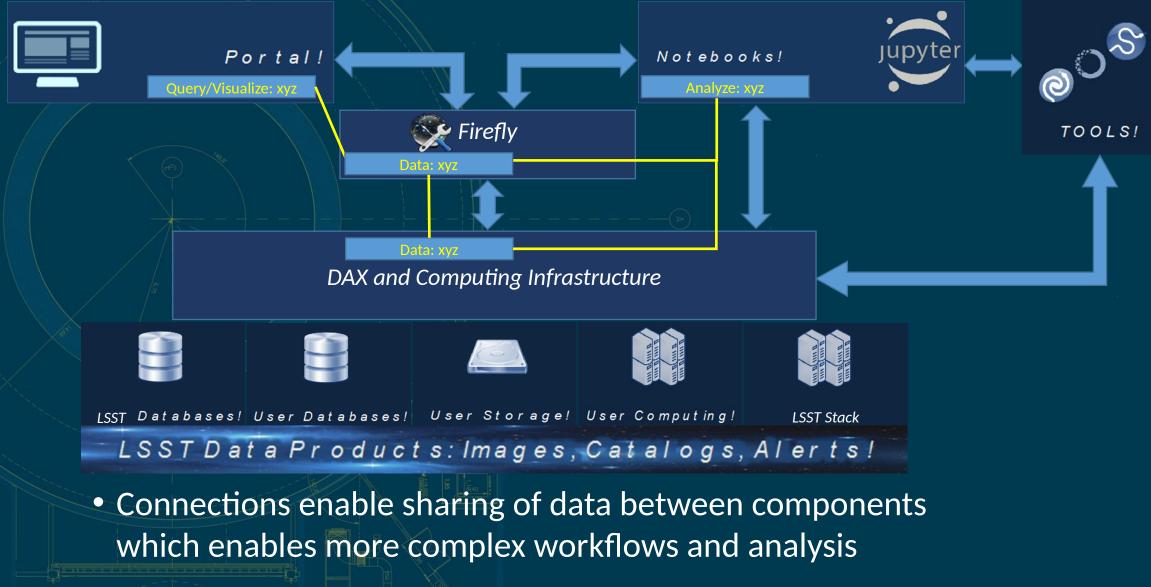
The Notebook Aspect of the



• Interactive Extran computing environment

- JupyterHub-based session management
- Python processes run with the user's identity
- User-customizable and persistent environments
 - Default configuration for new users based on pre-installed LSST stack
 - Allows the use of the full variety of community data analysis tools (e.g., SciPy, AstroPy, matplotlib, Bokeh)
 - Should support multiple saved environments per user to match stack releases to data releases
- Access to LSST data products via both Butler Python API and DAX services
- Access to additional compute resources at the Data Access Centers
 - Should support both "batch" and "interactive parallel" models (c.f. Dask)
- Access to "user workspace" storage as well as DAX-mediated Level 3 data products
- Visualization
 - Evolved version of afw.display supports a variety of back ends
 - Firefly back end supports both in-notebook visualization and push-to-portal (i.e., both 1-window and 2-window approaches). JupyterLab environment supports visualizations in flexible tabs.
 - Users are free to use any other Python-accessible notebook-aware visualization toolkits

Portal-Notebook Aspect Connections



Portal-Notebook Aspect Connections

Users are not confined to one aspect or the other

- We can easily achieve seamless connections between the two aspects of the Science Platform
 - Enabled by the common DAX architecture and computing infrastructure
 - Enabled by the use of Javascript tools for visualization, and the rich Firefly APIs
- Find or create data in one aspect; view or analyze that data in the other aspect
- Queries are shareable across the Portal and the Notebook, e.g.:
 - Build a query in the Portal UI; verify the results by browsing it in the UI; access the results from the Notebook for further analysis
 - Code a complex SQL query in the Notebook; browse the results in the Portal
 - Capture a query formulated in the Portal as code reusable later as a Notebook-driven query
 - Connections can be made either through identity management (DAX knows the queries you have recently performed) or through simple UI actions (e.g., copy/paste of a query token between windows)
- Moving from data discovery to analysis
 - We expect to provide a "one click" means of launching a fresh notebook with pre-loaded access to the results of data identified in the Portal

Portal-Notebook Aspect Connections

- The Firefly data model is accessible from the Notebook, e.g.:
 - afw.display calls can be used to set up overlays of tabular data and images
 - Results of Firefly operations (synthetic columns, interactive filtering/brushing) are retrievable in the notebook
- The Portal can be driven from the Notebook, e.g.:
 - Results of complex computations can be pushed to the Portal and explored, with immediate linkage to associated LSST data products
 - Multiple users can share a Portal view the system can be used for collaboration
- LSST-aware visualizations developed for the Portal are all naturally usable in the Notebook
 - User experience is common across aspects.
 - Duplication of development effort is minimized.
 - Some additional effort is required to package visualizations as widgets

An open question



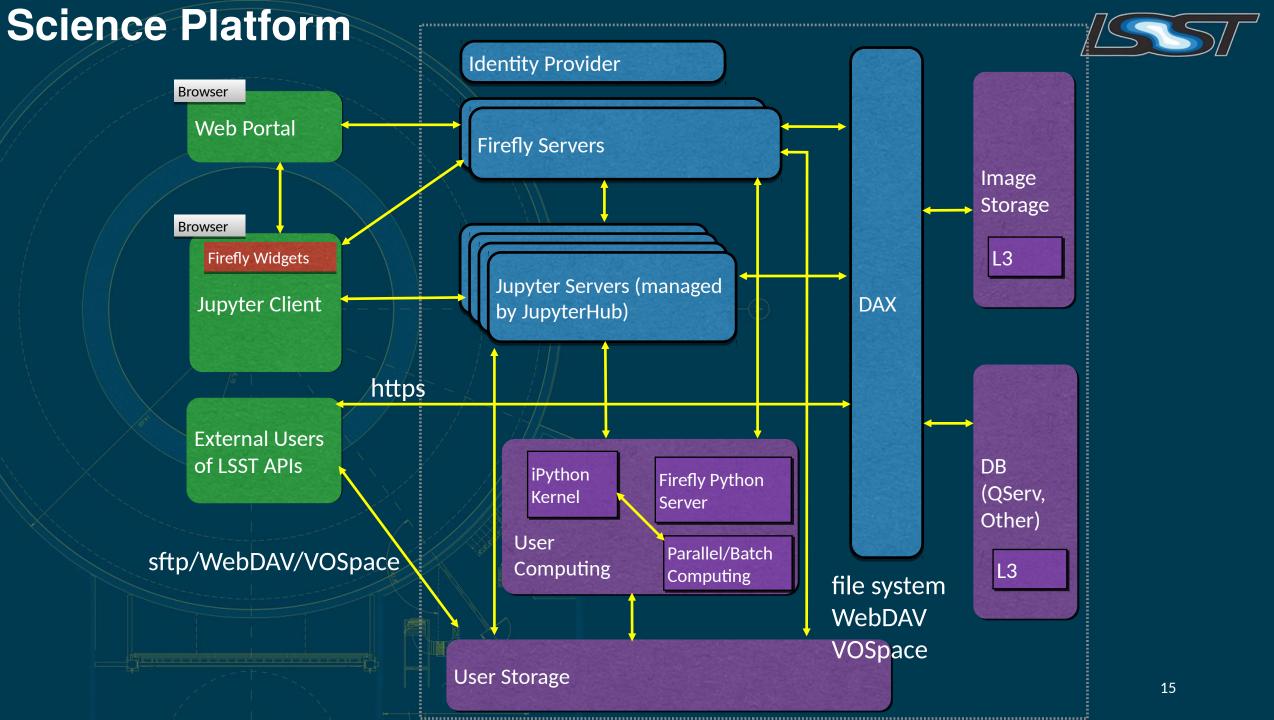
What exactly is the Python (Notebook) user's interface to the data?

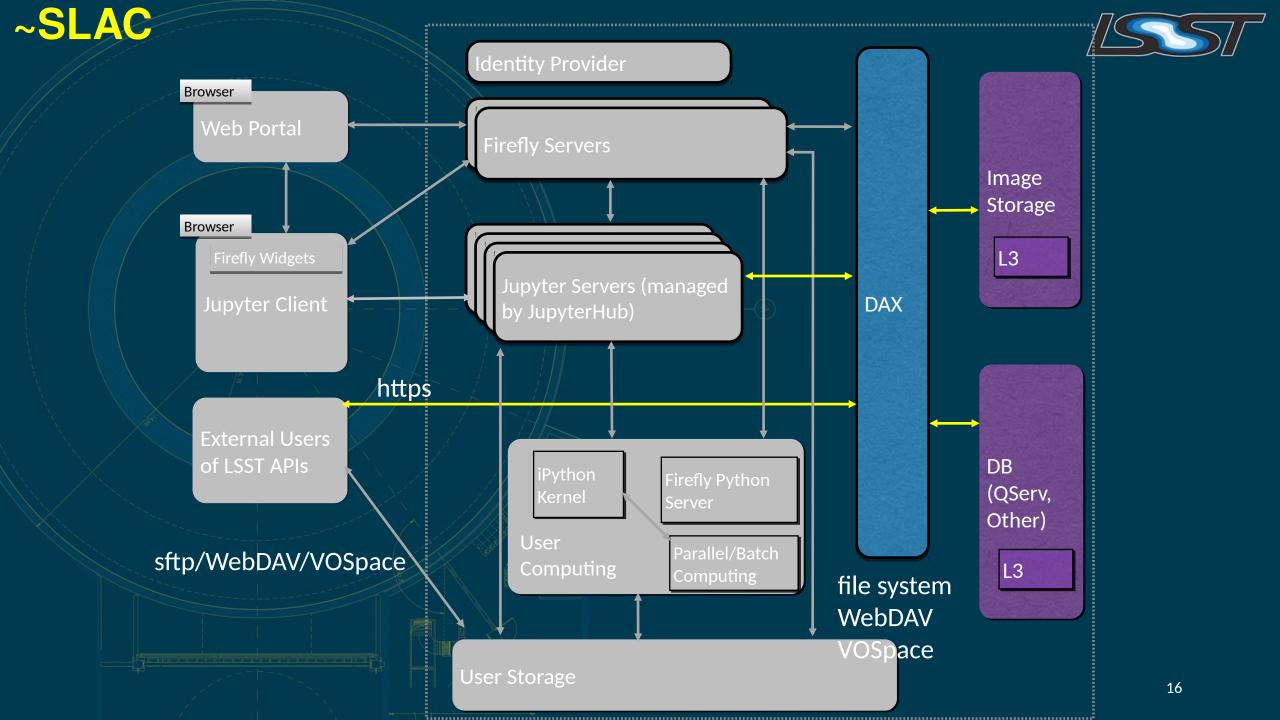
- The IPython kernels run locally *within* the Data Access Center they have access to whatever is visible to machines on the internal net
- They also have full access to the DAX APIs (VO and otherwise)
 - However, we currently don't plan LSST-aware Python wrappers for these
 - People can (already) call these (Web) APIs from Python and parse the results, but that seems like a below-threshold offering from a project of this scale
- Will we provide the means to "reconstitute" LSST-Python-domain versions of all the data products?
- Will we provide access to all of the Level 1 and Level 2 DPs through the Butler?

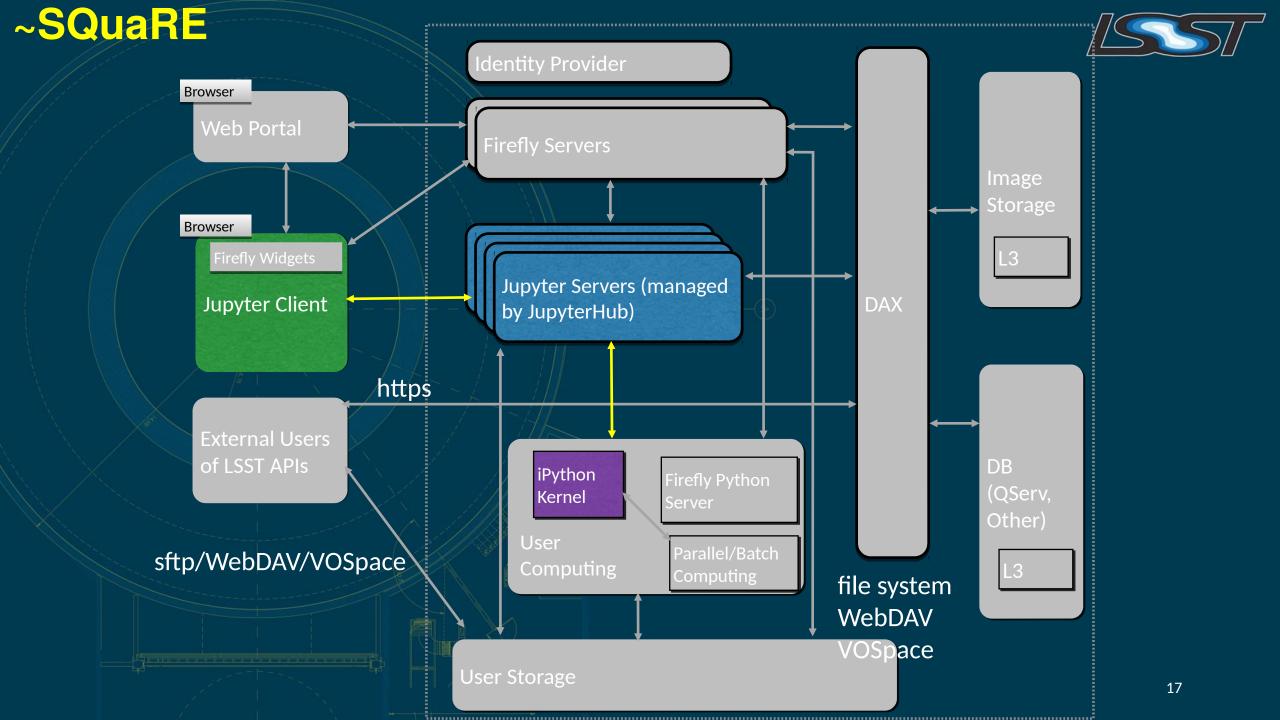
Science Platform Components

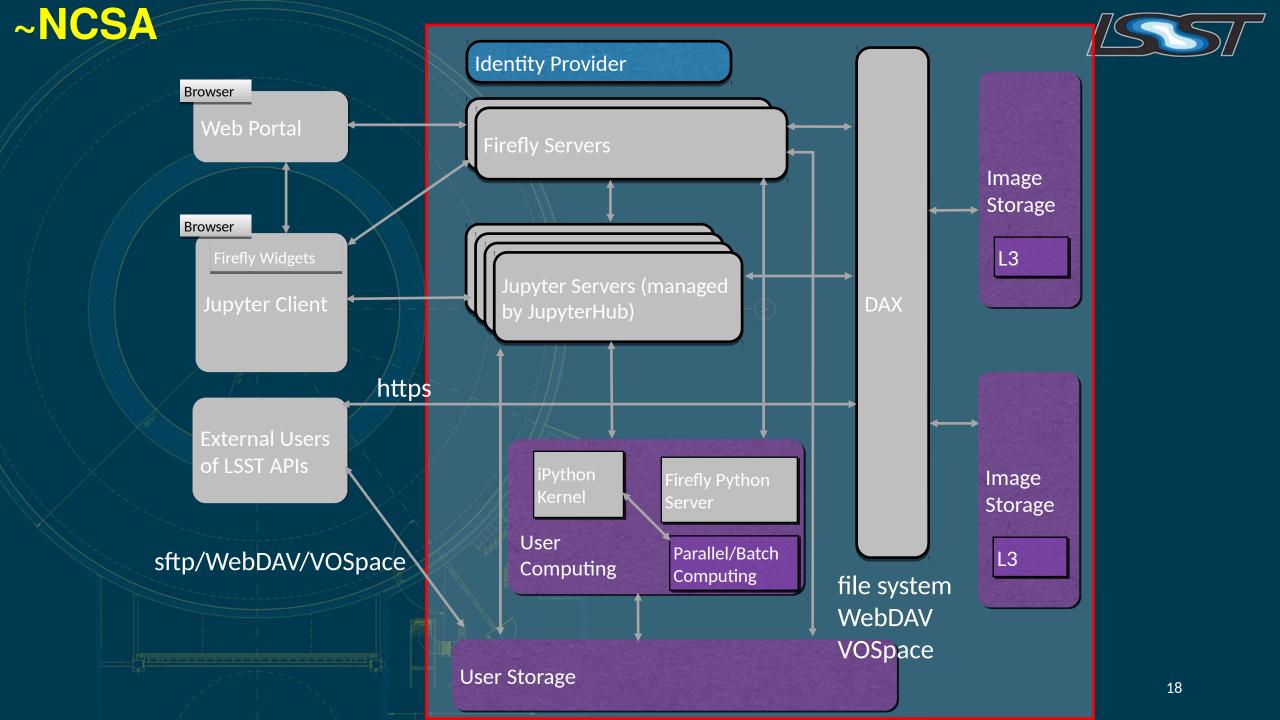


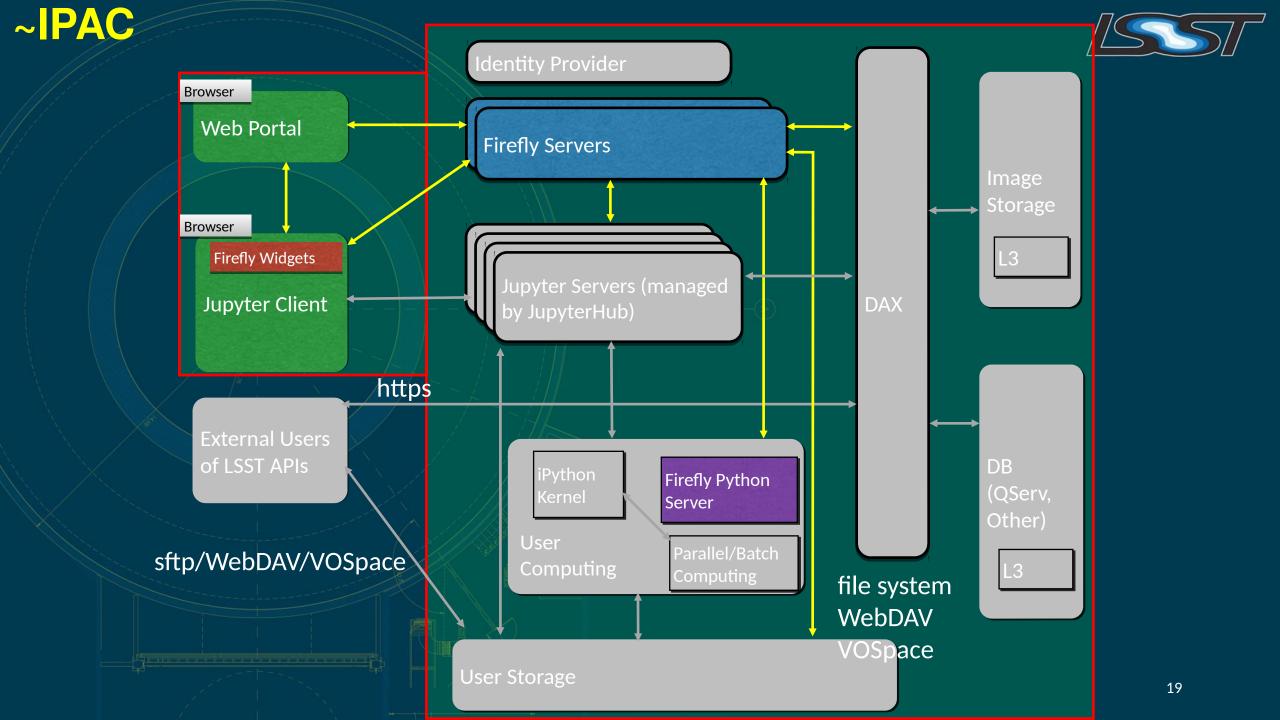
- We have identified major components necessary to support the Science Platform vision
- The following set of "wire-diagrams" outline these components and the necessary connections
- The Science Platform is an integrated service with contributions from multiple groups within the DM sub-system
- We have tried to identify areas of responsibility for the DM teams
 - These slides are a "first pass" and need input from the teams for assessment of completeness
 - The identified responsibilities are intended to be starting points of the conversation with the DM team at the JTM
 - There are areas of overlap between teams that need to be identified and scoped











SUIT Functionality for a Minimum Viable System

- Searching and access to Level 1 and Level 2 data products (catalogs, images, and alerts)
- Integration with Authentication and Authorization of Users
- Portal-based workflow accounting for the semantics of the LSST data
- Common computational and storage resources shared across the Science Platform visible from both the portal and the notebook environments
- Data exploration connected across the portal and the notebook
- Distributed computational model to enable analysis next to the data
- Alert subscription front-end to the alert mini-broker
- Availability of the Project Documentation

Demo: Creating an Integrated Environment

- Purpose: Demonstrate the distributed computing model comprising the Jupyter-Python Notebook, the Firefly visualization tools, and the Database APIs all within the single environment of the PDAC
- Demo Science Case: Forced Photometry Driven from Firefly using the Stack
- Highlights
 - Queried the PDAC from both portal and the notebook
 - Invoked a Firefly visualization in the portal from the notebook
 - Used Firefly visualization tools within the notebook
 - Enabled portal to invoke LSST stack and return results through call-back
 - Enabled use of data from call-back in the portal and the notebook
 - Within the portal enabled an LSST-contextual workflow by connecting data types
 - Demonstrates the Science Platform vision of Firefly, Notebook, and Python kernel all running remotely at PDAC, only the browser is local

• Caveats

- Demonstration is about functionality and does not represent the APIs the project expects to deliver
 - Some Python helper-code written as glue to query the DAX and run the photometry task from the stack
 - Incomplete APIs: afw.display, Butler access to tabular data,
- Admin tunnels into PDAC used to simulate a JupyterHub environment
- Not yet available to the general PDAC user

Demo: Initial Setup

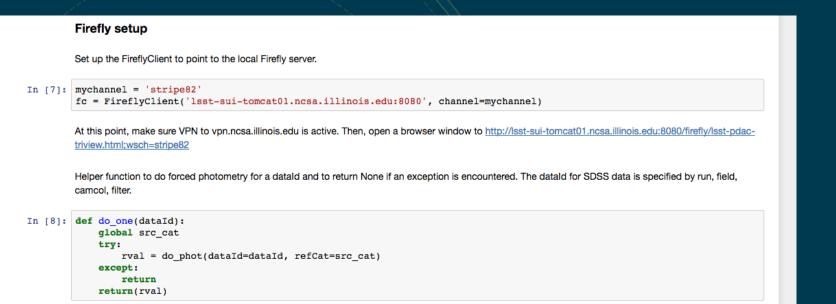


 Provide the state of the state	Jupyter Pick_filter_m	Iltiple_coadds Last Checkpoint: 4 minutes ago (unsaved changes)		ŕ	Logout		
 Light (Light) (E) (Light) (E)	e Edit View Insert	Cell Kernel Widgets Help			Python 2 O		
 User clickable Forced Photometry from the stack. The prote of whether a user selected coadd. The prote for Python 28 compatibility. The firsttransmitter is a user selected coadd. The firsttransmitter is a user selected coadd coadd. The firsttransmitter is a user selected coadd co	+ % 4 6 🔺 🗸	Code + CellToolbar					
<pre>imports imports imports for Python 23 compatibility. import for Python 23 compatibility. import for</pre>	This notebook uplo	ads a table of deep coadds from Stripe 82 data, and sets up Firefly to retur		y from the LSST stack by clicking on a	a	 Astropy table and time functions Firefly client 	
In [1]; from _future_ import print_function, division, absolute_import Standard library imports, and allowing the notebook server to find needed modules. In [2]: import seg import concurrent.futures import tempfile Import stank. In [5]: @ys.path.insert(0, '/home/shupe/projects/firefly_client/firefly_client') import isst.afv.display as afv0isplay import list.afv.finage as afv0isplay import stank. In [3]: import list.afv.finage as afv0isplay import stank. In [3]: from list.daft.perintence import Butler Imports from Astropy. In [4]: from astropy.table import Table, vstack, join, Column In [4]: from astropy.table import Time In [6]: from dax.utils import query_tap_ison, get_image_table In [6]: from dax.utils import query_tap_ison, get_image_table In [6]: from dax.utils import query_tap_ison, get_image_table, do_phot In [6]: from d	•					 DAX query helper functions 	
Standard library imports, and allowing the notebook server to find needed modules. In [2]: import sys import concurrent.futures import concurrent.futures import tempfile Import fast.afv.display as afwDisplay import last.afv.display import last.afv.display as afwDisplay import last.afv.display as afwDisplay import last.afv.display import last.afv.display import last.afv.display import last.afv.display import last.afv.display import last.afv.displa	Imports for Python	2/3 compatibility.					
<pre>In [2]: import sys import concurrent.futures import concurrent.futures import tempfile Imports from the LSST stack. In [3]: import list.afw.display as afwDisplay import list.afw.display as afwDisplay import list.afw.display as afwDisplay import list.afw.display as afwTmage import list.afw.display as afwTmage imp</pre>	In [1]: fromfuture_	_ import print_function, division, absolute_import			_		
<pre>In [2]: import sys import concurrent.futures import concurrent.futures import tempfile Import from the LSST stack. In [3]: import lsst.afw.display as afwDisplay import lsst.afw.display as afwDisplay import lsst.afw.display as afwDisplay import lsst.log from lsst.daf.persistence import Butler Imports from Astropy. In [4]: from astropy.table import Table, vstack, join, Column from astropy.table import Table, vstack, join, Column In [6]: from dax_utils import query_tap_json, get_image_table In [6]: from dax_utils import query_tap_json, get_image_table In [6]: from dax_utils import guery_tap_json, get_image_table In [6]: from dax_utils import make_refcat, parse_phot_table, do_phot</pre>	Standard library imp	ports, and allowing the notebook server to find needed modules.					
<pre>import concurrent.futures import tempfile Import tempfile Import stack. In [3]: import lsst.afw.display as afwDisplay import lsst.afw.display as afwDisplay import lsst.afw.image as afwImage Im [4]: from astropy.table import Table, vstack, join, Column from astropy.time import Table, vstack, join, Column from astropy.table import make_refcat, parse_phot_table, do_phot Im [6]: from dax_utils import query_tap_json, get_image_table from stripe82phot import make_refcat, parse_phot_table, do_phot </pre>				Import a development version of the	e Firefly Python cli	ent.	
In [3]: import lsst.afw.display as afwDisplay import lsst.afw.image as afwTmage import lsst.df.persistence import Butler Imports from Astropy. In [4]: from astropy.table import Table, vstack, join, Column from astropy.time import Time Import lime import Time Im [6]: from datutils import query_tap_json, get_image_table from stripe82phot import make_refcat, parse_phot_table, do_phot	import concurre		In [5]:	<pre>import firefly_client</pre>			
<pre>import lsst.afw.image as afwImage import lsst.log from lsst.log from lsst.daf.persistence import Butler</pre> imports from Astropy. In [4]: from astropy.table import Table, vstack, join, Column from astropy.time import Time In [6]: from dax_utils import query_tap_json, get_image_table, do_phot In [6]: from dax_utils import query_tap_json, get_image_table, do_phot	Imports from the LS	SST stack.		Import custom functions for forced	photometry.		
Imports from Astropy. In [4]: from astropy.table import Table, vstack, join, Column from astropy.time import Time In [6]: from dax_utils import query_tap_json, get_image_table from stripe82phot import make_refcat, parse_phot_table, do_phot	<pre>import lsst.af import lsst.log</pre>	w.image as afwImage		the error message is printed. get_image_table is a conver	enience function th	at uses <code>query_tap_json</code> to return a Pandas dataframe of images at a given coordinate, o	
In [4]: from astropy.table import Table, vstack, join, Column from astropy.time import Time In [6]: from dax_utils import query_tap_json, get_image_table from stripe82phot import make_refcat, parse_phot_table, do_phot	Imports from Astrop	y.		make_refcat makes a referen parse_phot_table converts	nce catalog for for	bed photometry.	ulating
from stripe82phot import make_refcat, parse_phot_table, do_phot					etry for a given data	ald and reference catalog.	
06 - 08 March 2017 LSST Joint Technical Meeting 2017 22			In [6]:				
	06 - 08 Marc	h 2017	LSST Jo	oint Technical Meeting 20	017	22	2

n imports including

- T-stack modules
- ropy table and time nctions
- efly client
- X query helper functions

Demo: Initialize Firefly and Define Function



• Initialize Firefly client to point to the Firefly server on the PDAC

- Add the callback function on the Firefly server for the Forced Photometry function.
- Add an extension to the Firefly browser to make a Forced Photometry button appear in Point mode.

Add the callback (on the Firefly server).

In [11]: plistner = fc.add_listener(callback_forcedphot)

Add the Forced Phot extension for all image displays

Notebook

Coordinates and filter specification, and image retrieval

A very nice variable source is at RA=45.804433, Dec=0.905573. (For name resolvers, it is V* GI Cet.) Here it is used just for image retrieval, along with the specified filter name.

Other high-amplitude variables with P > 50 days, rAmpl > 3.0:

25.612807 0.291621, ID=1340590, rAmpl=3.217, P=2958d. (2SLAQ J014227.07+001729.8 -- Star.) 28.930942 0.468687, ID=1261335, rAmpl=4.146, P=3321 d. (V* FL Cet -- CV of AM Her type (polar) -- the period is really <u>87 minutes</u>.)

Shorter periods:

10.62013 -0.869339, ID=196130, iAmpl=8.059, P=0.6d. (SDSS J004228.83-005210.4, White Dwarf.)

QSOs:

2.87667 0.964417 ID=68411, iAmpl=2.218, P=2875 days, ([VV2006] J001130.4+005751 -- Quasar.) 319.572495 0.221337, id=2655567, iAmpl=3.113, P=1453 days UGC 2479 is 45.167525 0.020461

You may specify any coordinates in the DC_2013 Stripe 82 processing.

```
In [13]: myra = 25.6
mydec = 0.3
```

Query the DeepCoadd table for coadds covering the user-specified RA and Dec, for any filter

In [14]: df_coadds = get_image_table(ra=myra, dec=mydec, filter_name=None, table_name='DeepCoa df_coadds.sort_values(['deepCoaddId'], inplace=True)

Verify that df_coadds has at least five entries for the five filters

Query the DAX on the
DeepCoadd table for Stripe
82 at the starting RA and
Dec.

	Search Catal	ogs	L	oad Catalo.	q					
	Catalogs O Ima Deep Coadd Science CCD E	-		Name	or Position 25.53)d16m48s Er or 1h42m9.0s,	Try NED then Simbad ᅌ +0d16m48.0s Equ J2000	×
			Search	Type (Regi	ion Inters	ection):		Image	e contains target	
										45 0.3147
										45 0.3147
				-		lank for full i		360	arcsecond	 45 0.3147 45 0.3147
			Return t	the most o	entered in	mage contain	ning the ta	rget: 💿 Yes	○ No	45 0.3147
Res		constraints	Type	Null	Key	Default	Fytra	Unit	Description	
	Field	constraints	Type	Null	Key	Default	Extra	Unit	Description	
	Field deepCoaddId	constraints	bigint(20)	NO	PRI	Default	Extra	dummyUnit1	description of deepCoaddId	
	Field deepCoaddId tract	constraints	bigint(20) int(11)	NO NO		Default	Extra	dummyUnit1 dummyUnit1	description of deepCoaddId description of tract	
	Field deepCoaddId tract	constraints	bigint(20)	NO	PRI	Default	Extra	dummyUnit1 dummyUnit1 dummyUnit1	description of deepCoaddId	
	Field deepCoaddId tract patch	constraints	bigint(20) int(11) char(16)	NO NO NO	PRI MUL	Default	Extra	dummyUnit1 dummyUnit1 dummyUnit1 dummyUnit1	description of deepCoaddId description of tract description of patch	
	Field deepCoaddId tract patch filterId	constraints	bigint(20) int(11) char(16) tinyint(4)	NO NO NO	PRI MUL	Default	Extra	dummyUnit1 dummyUnit1 dummyUnit1 dummyUnit1 dummyUnit1	description of deepCoaddId description of tract description of patch description of filterId	
	Field deepCoaddId tract patch filterId filterName	constraints	bigint(20) int(11) char(16) tinyint(4) char(3)	NO NO NO NO	PRI MUL	Default	Extra	dummyUnit1 dummyUnit1 dummyUnit1 dummyUnit1 dummyUnit1 dummyUnit1	description of deepCoaddId description of tract description of patch description of filterId description of filterName	
C C C C C C C C C C C C C C C C C C C	Field deepCoaddId tract patch filterId filterName ra decl tional constraints () 3ser>7 and (w2mg e_id mf = '1861pi		bigint(20) int(11) char(16) invint(4) char(3) double double	NO NO NO NO NO NO	PRI MUL MUL			dummyUnit1 dummyUnit1 dummyUnit1 dummyUnit1 dummyUnit1 dummyUnit1	description of deepCoaddId description of tract description of patch description of filterId description of filterName description of ra	

Demo: Results Return in Portal or



tri-view

25.529944999

25.5299449999

25,52994499999999

25,529944999999998

Lsst-DeepCoadd-Center([25.53...

ima-tbl ima-xv xv-tb

0.3147115824

0.3147115824

0.31471158249

Vlatabaak

Add a URL for downloading the image by ID

Verify that df_coadds has at least five entries for the five filters

In [15]: df_coadds

Out[15]:		deepCoaddId	tract	patch	filterId	filterName	ra	decl	htmld20	equinox	raDeSys		corner3Ra	corner3Decl	corner4Ra
	0	19202104	0	293,7	0	u	25.529945	0.314712	17147177442654	2000.0	ICRS		25.416645	0.422899	25.416645
	1	19202105	0	293,7	1	g	25.529945	0.314712	17147177442654	2000.0	ICRS		25.416645	0.422899	25.416645
	3	19202106	0	293,7	2	r	25.529945	0.314712	17147177442654	2000.0	IC		Q Q Q (Q	Q Q \$	Coverage
	2	19202107	0	293,7	3	i	25.529945	0.314712	17147177442654	2000.0		Dins:		Photometry	
	4	19202108	0	293,7	4	z	25.529945	0.314712	17147177442654	2000.0	IC	<u>.3x</u>			

5 rows × 34 columns

• Results available in portal and notebook

- Data specific connections enabled in context of portal: images connected to table in portal
- Output driven by notebook commands

deenCoadd

0

Prepare Download

1920210

1920210

1920210

19202106

19202108

deepCoaddId

• Forced Photometry function available for images

Forced Phot

4 4

293,7

293,7

293,7

293,7

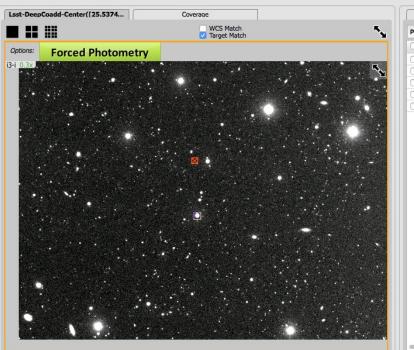
293,7

(1 - 5 of 5)

	WCS Match Target Match	۳.
Forced Photometry		
0.3x		
		· · · · · · · ·
		Constant and the
	····	a financia a contra
· • . · · · · · · · · · · · · · · · · ·		
	• 🗢	
the second second second second		
A CALLER		

Demo: Function Iriggers Python





	deepCoa	dds	×	Forced Phot	×	Lsst-DeepCoadd-Cente	er([25.53 ×
Pre	pare Download		••	1 of 1) (1	- 5 of 5)		
	deepCoaddId	tract	patch	filterId	filterName	ra	dec
	19202104	0	293,7	0	u	25.529944999999998	0.31471158249
	19202105	0	293,7	1	g	25.529944999999998	0.31471158249
	19202107	0	293,7	3	i	25.529944999999998	0.31471158249
	19202106	0	293,7	2	r	25.529944999999998	0.31471158249
	19202108	0	293,7	4	z	25,529944999999998	0.31471158249

Forced Photometry function triggers execution of Python LSST-stack process

ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement							
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
 ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	
ForcedPhotExternalCatalogTask.measurement							
ForcedPhotExternalCatalogTask.measurement	INF0:	Performing	forced	measurement	on 1	sources	

Demo: Data Interaction Enables



		Table widget of photometry
In	[19]:	<pre>from firefly_widgets import connect connect('http://lsst-sui-tomcat01:8080')</pre>
In	[20]:	<pre>from firefly_widgets import TableViewer</pre>
In	[21]:	tval = 'placeholder'
In	[221:	<pre>tv = TableViewer(url or path=tval, title='Forced Phot',</pre>
		<pre>tbl_id = 'Forced Phot', width='600px', height='400px')</pre>

♀️⃣ ♬ 唸 ╲

\${temp-files}/upload_1919959...

1 of 1 (1 - 87 of 87)

base PsfFlux

69.3174216179

1191060853

55.6572279341

14.6632598778

49.7973404920

23,3733538785

23.0182229028

19.6829789459

43,5970849158

32 0574752448

28.3745018288

34.0630089545

101.2390207980

base_PsfFlux_flux base_PsfFlux

ct

float64

21.17705909

32,744811299

27.08565881

20.013489445

In [23]: tv

placeholder

mjd

52170.3895975

52196.3882552

52207.3548217

52224.2830506

52231.2683537

52234,2397681

52261.2031724

52282.1297735

52287.0836163

52552.4022462

52557.3767055

52908.3512813

52934.2828505

In [24]: tv.selection()

Out[24]: <Table length=87>

mid

float64

52912 3305059 64 4638100228

52936.2789393 72.9645301282

ct

52170.3895975 66.784811304

52196.3882552 11.1389714544

52207.3548217 47.1346952678

52224.2830506 6.54654962581

52231.2683537 47.7502228036

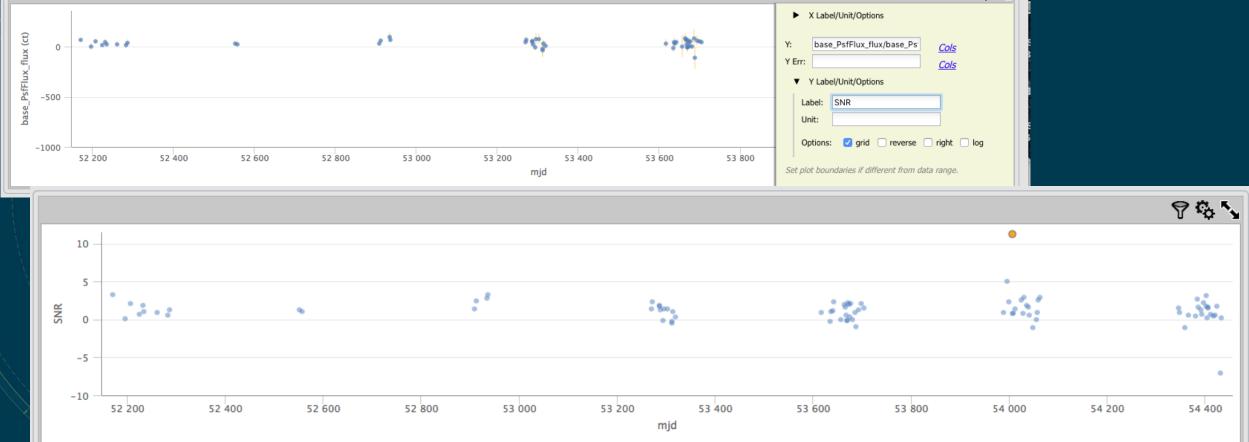
float64

- Results available in Notebook as Python object and as • Firefly table
- Interactive tools available inline in notebook as well as a • structured workflow in browser

817754532 906791036 977233224 830151562	25.53856	69 0.312346547081	2				- C 🕆 🕅							
77233224	25,53856													t
		59 0.312346547081	:		Forced Phot		Coverage			deepCo	adds ×	Forced Phot	×	
51562	25.53856	59 0.312346547081	4				WCS Match		5		14.4	of 1 🕨 🚺 (1 - 87 of 87)		
	25.53856	0.312346547081	1				Target Match		<u>``</u>	🗌 mjd	base_PsfFlux_flux		ia ra	
97822180	25.53856	0.312346547081	2	Image:	IMAGE 1/3					52170.3895975		21.1817754532	25.538569	
61034770	25.53856	0.312346547081	3	ra=25.53	38569&dec=0.31234654	7081&wi 3.49	0	新行用的 和目的		52196.3882552		32.6906791036	25.538569	
912110035	25.53856				Contraction of the		the state of the state	all states and		52207.3548217	55.6572279341	27.1677233224	25.538569	
			1		10 10 10 K	1.50	and the second second		936	52224.2830506	14.6632598778	20.1330151562	25.538569	
8563386405	25.53856		4		2.47.42.642	1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 -			5453	52231.2683537	49.7973404920	26.1997822180	25.538569	
18131562	25.53856	59 0.312346547081	1		Street 2.55			1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 -	2012 C	52234.2397681	23.3733538785	23.3361034770	25.538569	
297455658	25.53856	59 0.312346547081	1		Constant and	Sec. 1		1 Sugar 1 St		52261.2031724	23.0182229028	26.2912110035	25.538569	
9661989	25.53856	59 0.312346547081	:		STREET, STREET	100 C		A State of the	1000	52282.1297735	19.6829789459	33.8563386405	25.538569	
2403439	25.53856		3		CHEER W	52/4/2	and an and the second sec	949 C	2005	52287.0836163	43.5970849158	33.2118131562	25.538569	
069153		59 0.312346547081			Contraction and	1.46	States in the	Statistics of the	1221	52552.4022462		25.7297455658	25.538569	0.
			1		12 C (1)	1.1.1		the states		52557.3767055	28.3745018288	29.1009661989	25.538569	0.3
73294	25.53856		4		State of the second	14 900	ALL STREET, SAN	Call of the Call of the	352	 52908.3512813 52912.3305059 	34.0630089545 64.4638100228	24.2982403439 25.9105069153	25.538569 25.538569	0.3
2808986	25.53856	59 0.312346547081	2		100000000	1000	- <mark>0</mark> - 1		8365	52912.3305059	101.2390207980	35.9924473294	25.538569	
					一、	$r \rightarrow r$		2	333	52936.2789393	72.9645301282	22.4702808986	25.538569	0
					- 200 States	3.6.43				53270.3429448	50.9650720325	37.5595373816	25.538569	0.3
					North Control of	1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1	1 (C) (C)	and the second second	1998 - C	53272.3930058	69.4382844986	30.6474712111	25.538569	0.3
					10000	2 - A			225	53286.3614394	58.0092546776	31.6890617451	25.538569	0.3
					10000	Sec. Sec.				53286.3618539	56.5414200228	31.5513680318	25.538569	0.31
					Call Call	12.5.64	a state in the second	Constanting.	1000	53288.3199770	27.5865258228	22.0547801568	25.538569	0.31
					2565.552	1.5723	· 在在一些市场的。		852	53294.3679894	-3.5372555698	36.1540002460	25.538569	0.31
Flux_fluxSigm	a ra	dec	expMidpt		111111111111		1 - C - C - Z - Z -	19 A 4 1 1 1 1 1	2003	53296.3149000	75.1712281523	54.4485957289	25.538569	0.3
					R. M. L. M.				1246	53302.3159223	75.8132466992	55.8261727393	25.538569	0.:
		deg			CONTRACTOR OF		12.2	14.5 19 2 3	125	53312.2711300	-20.8340698865	73.2038947481	25.538569	C
		-			中国的政治	100.00		and the second	121	53312.2715446	-33.5822446786	74.9460290701	25.538569	
	float64	float64	str30											
			2001-09-											ļ
8008	25.538569	0.312302546427											•	
			18T09:21:	Ð								L.	1.12	
) Xn	0		•	e :		157				
			2001-10-	Ľ,							Ţ			
94	25.538569	0.312302546427		Elu -										
			14T09:19:	a -5	500									
				base										
				~										
79	25.538569	0.312302546427	2001-10-	-10										_
	20.000000	CIC ILCOLO ICHEI	25T08:30:		52 200	52 400	52 600	52 800 53 0	00 53 200	53 400 mjd	53 600	53 800 54	4 000	
										inju				
	05 500500	0.312302546427	2001-11-		2700	36 4								
6			11T06:47:3											

- Data specific connections enabled in context of portal: images connected to table in portal
- Connected data model • enables selection and visualization of single visit images

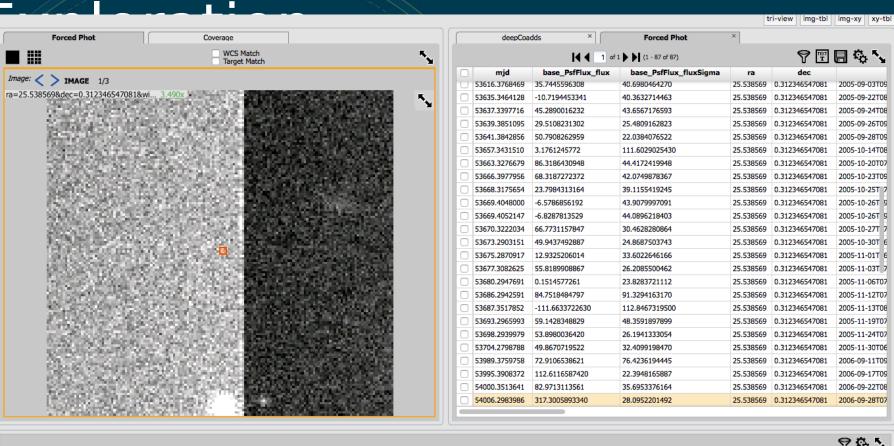
Demo: Data Interaction Enables

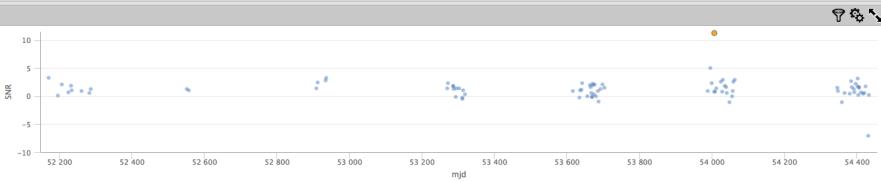


• Interactive plotting tools enable math for immediate exploration of the data

• Flux with errors plotted vs time changed to plotting SNR vs time

Demo: Data Interaction Enables







 Connection of images, tables, and plots enables exploration of data

۲

In this case, high SNR data point caused by image artifact clearly seen when data point selected and associated image displayed

06 - 08 March 2017

Demo: Creating an Integrated Environment

• Purpose: Demonstrate the distributed computing model comprising the Jupyter-Python Notebook, the Firefly visualization tools, and the Database APIs – all within the single environment of the PDAC

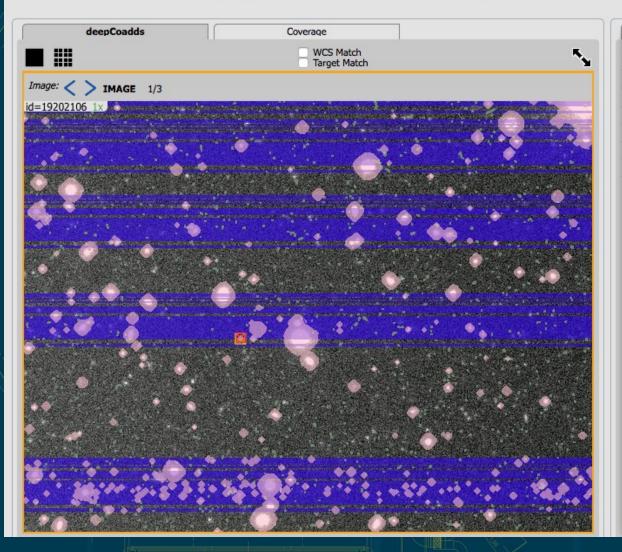
Demo Science Case. Forced Photometry Driven from Firefly using the Stack

i ligi ligi lis

- Queried the PDAC from both portal and the notebook
- Invoked a Firefly visualization in the portal from the notebook
- Used Firefly visualization tools within the notebook
- Enabled portal to invoke LSST stack and return results through call-back
- Enabled use of data from call-back in the portal and the notebook
- Within the portal enabled an LSST-contextual workflow by connecting data types
- Demonstrates the Science Platform vision of Firefly, Notebook, and Python kernel all running remotely at PDAC, only the browser is local

Caveats

- Demonstration is about functionality and does not represent the APIs the project expects to deliver
 - Some Python helper-code written as glue to query the DAX and run the photometry task from the stack
 - Incomplete APIs: afw.display, Butler access to tabular data,
- Admin tunnels into PDAC used to simulate a JupyterHub environment
- Not yet available to the general PDAC user



Layers- id=19202106		
Jbit # 0 - MP_BAD	Color	<u>Delete</u>
☑ bit # 1 - MP_SAT	Color	<u>Delete</u>
Jbit # 2 - MP_INTRP	Color	<u>Delete</u>
☑ bit # 3 - MP_CR	Color	<u>Delete</u>
✓ bit # 4 - MP_EDGE	Color	<u>Delete</u>
Jit # 5 - MP_DETECTED	Color	<u>Delete</u>
<pre>viit # 6 - MP_DETECTED_NEGATIVE</pre>	Color	<u>Delete</u>
✓ bit # 7 - MP_SUSPECT	Color	<u>Delete</u>
Jbit # 8 - MP_NO_DATA	Color	Delete
Selected Point	Color Symb	ol <u>Delete</u>
✔ 1h42m7.19s +0d18m53.0s Equ J2000	Color O Symb	ol <u>Delete</u>
Sorced Phot	Color D Symb	ol Delete
Show All Hide All		



Image data masks controllable by the API and the GUI Prioritized the functionality to support the Pipeline development Operational functionality on the PDAC for the Stripe82 data

The API aspect of the Science Platform



Encompasses all externally visible APIs provided from the Data Access Centers

• Web APIs

- "Zero-install" access to LSST data for remote users
- VO standard data retrieval protocols (TAP, SIA, etc.) implemented by the DAX team
- Additional LSST-specific DAX APIs
 - Currently cutouts are made available in this way; we should evaluate a SODA solution
 - May include a Web "Remote Butler" API which provides the data-naming and data-relationship functions of the Butler
- "Workspace" APIs for user data space access and management (WebDAV or VOSpace, tbd)
- Enable the use of community tools that are based on VO standards

Python APIs

- "The stack" is intended to be installable by remote users
- An unresolved question for the re-plan: Python APIs for retrieving / reconstituting LSST-domain Python data objects via remote use of the Web APIs (VO et al.)
 - Could be implemented by wrapping the Web Remote Butler in Python

Support for remote operation

- When you run in the Notebook Aspect, your Python process is at the DAC
- People who start working in the DAC environment and create notebooks will reasonably ask: can use my notebook off-site?
 - This is clearly related to the "API Aspect" of the Science Platform
 - We also should consider the case of completely offline operation
- Some considerations:
 - Read-only viewing: we will use Jupyter's widget facilities to ensure that a frozen snapshot of any LSSTdeveloped visualizations is saved with notebooks
 - Remote operations: we need to understand the differences between data access capabilities in Python at the DAC and remotely
 - Will we support remotely-running Butlers accessing data in the DACs?
 - Will we design a system that facilitates exporting data subsets to remote systems and setting up Butler access there?
 - Disconnected operations:
 - If you have local data and run a local Jupyter server and IPython kernel, you should be able to work

Summary



- Defined a vision for the Science Platform
 - Fulfills the purposes of the platform
 - Enables use of the platform by a continuum of users from novice through expert
- Vision is have a portal and notebook environment connected via shared resources to enable a seamless workflow for both the novice and expert users
- Defined a minimum viable system for the science platform
 - Scientific Drivers
 - Minimum set of functionality
 - Identified the components and the connections necessary
- Built demonstration system that demonstrates the aspects of the Science Platform by connecting the Jupyter-Python Notebook, the Firefly visualization tools, and the Database APIs





About Firefly...



- Firefly is open-source software
- A set of tools in a client-server architecture
 - Integration of native astronomical image visualization (no pre-processing required)
 - Visualization of tabular data as plots, histograms, and image overlays integrated with image visualization
 - Interactive and distributed
 - Brushing and linking shared data model across multiple images and tables

Server	Client
 Data model management, I/O, user session management Data model supports server-side assistance with large data sets (e.g., decimation, paging) Robust multithreaded Java framework serves large number of simultaneous users Scalable deployment Search Processor abstraction handles many sources of data – VO interfaces, files, custom Framework for calling external code to extend server's capabilities (e.g., in Python) Remote APIs for bidirectional data transfer, visualization, and session control 	 Visualization and interactivity Built on open-source React.js framework, Highcharts interactive plotting package JavaScript APIs for customization of layout, features, and interactivity Readily integrated into Jupyter / JupyterLab environment as "widgets"

Firefly Capabilities



Portal-building Toolkit	API	Server Infrastructure	Central Data Model
 Query-building Name resolution Layout and UI 	 Embedded Firefly and Jupyter widgets (JavaScript API) Remote control and data exchange (Remote/Python API) 	 Service and data integration Extensible and scalable Security layer Cache System Background management 	 Definable data relations Multiple, synchronized views of data
Image Visualization	Tabular Data Visualization	2D Charts and Histograms	Analysis Tools
 Distributed astronomical image visualization Sensitive to multiple WCS representations Native FITS visualizer - no preprocessing required Supports many DS9 like features: zoom, rotate, magnifier, thumbnail, color, stretch, 3-color, region annotations, etc. Overlays tabular data 	 Distributed data Supports large data sets, greater then 1 million rows DB-like filtering and sorting Table viewing 	 Tied to Data Model Supports decimated plotting for large data sets 	 Computations on columns Adaptive histogramming Time series analysis

Firefly Applications...



Specific collections of tools and behavior configured for different purposes

• In production in the IRSA archives

- Many Tools: WISE, Spitzer, FinderChart, IRSAViewer, Planck, PTF
- PDACv1
 - Basic application demonstrating integration with DAX data sources
 - Time Series Viewer

• The Science Platform "Portal aspect" will evolve from these