The LSST Science Platform

Gregory Dubois-Felsmann
LSST Science Platform Scientist

Caltech/IPAC

October 24, 2017

The LSST Science Platform (LSP)

- Purpose
  - Enable access to the LSST data products
  - Enable visualization and exploration of the LSST data
  - Provide an interface for added-value processing and analysis close to the data
    - Interactive Python-based interaction with the data is key! LSST DM code base is Python (and C++ below).
  - Provide access to documentation:
    - Data content, data quality, data processing, software, survey, Observatory
    - Contextual navigation of the data model
- Customers
  - LSST Data Rights Scientists, Science Collaborations, and LSST Project staff
  - The Platform needs to be simple enough to engage general users and flexible enough to meet the needs of experienced users; users will grow and adapt with time.
  - The Platform is used internally for science validation, commissioning, calibration.

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 Portal environment and the flexible Notebook 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

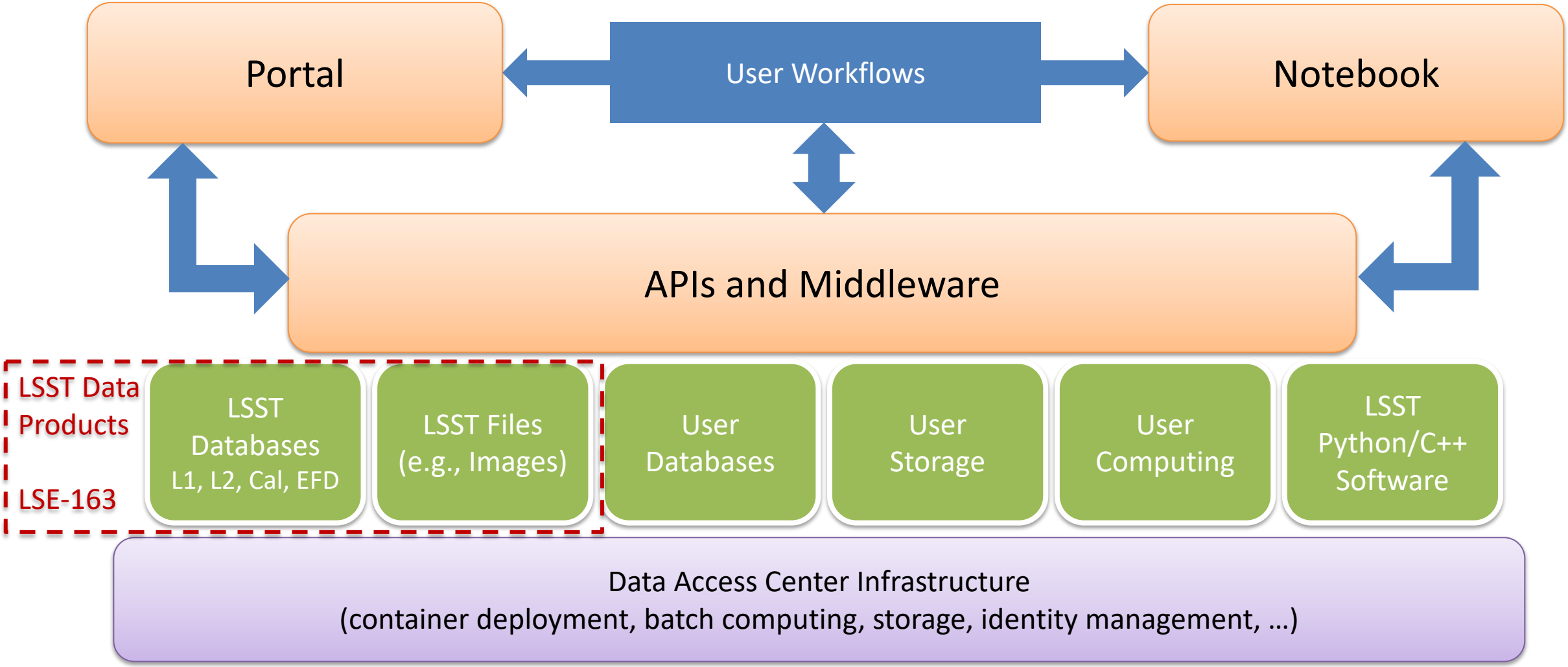
APIs

Unifying Infrastructure

The Science Platform Vision

- The Science Platform concept rests on the provision of three "*Aspects*" of access to the LSST data, storage, and computing: Portal, Notebook, and APIs
  - A Portal with access to all LSST data products, visualization and exploration tools, and structured workflows that guide users to discover and understand the data and their semantic connections.
  - A Notebook-style flexible, interactive Python computational environment (JupyterHub/Lab) with access to the LSST Data Products, computing, and storage, and with pre-provisioned access to installed versions of the LSST Python/C++ software stack.
  - A set of APIs underpinning these user environments, both internally and externally available, providing access to the Data Products, primarily based on IVOA standards.
- All these rest on a computational infrastructure; a flexible LSST and user storage infrastructure; and a database system supporting both the Project data products and users' own databases.
- Enable the users to create workflows that cross between the Portal and Notebook environments and the use of the APIs as their needs dictate.

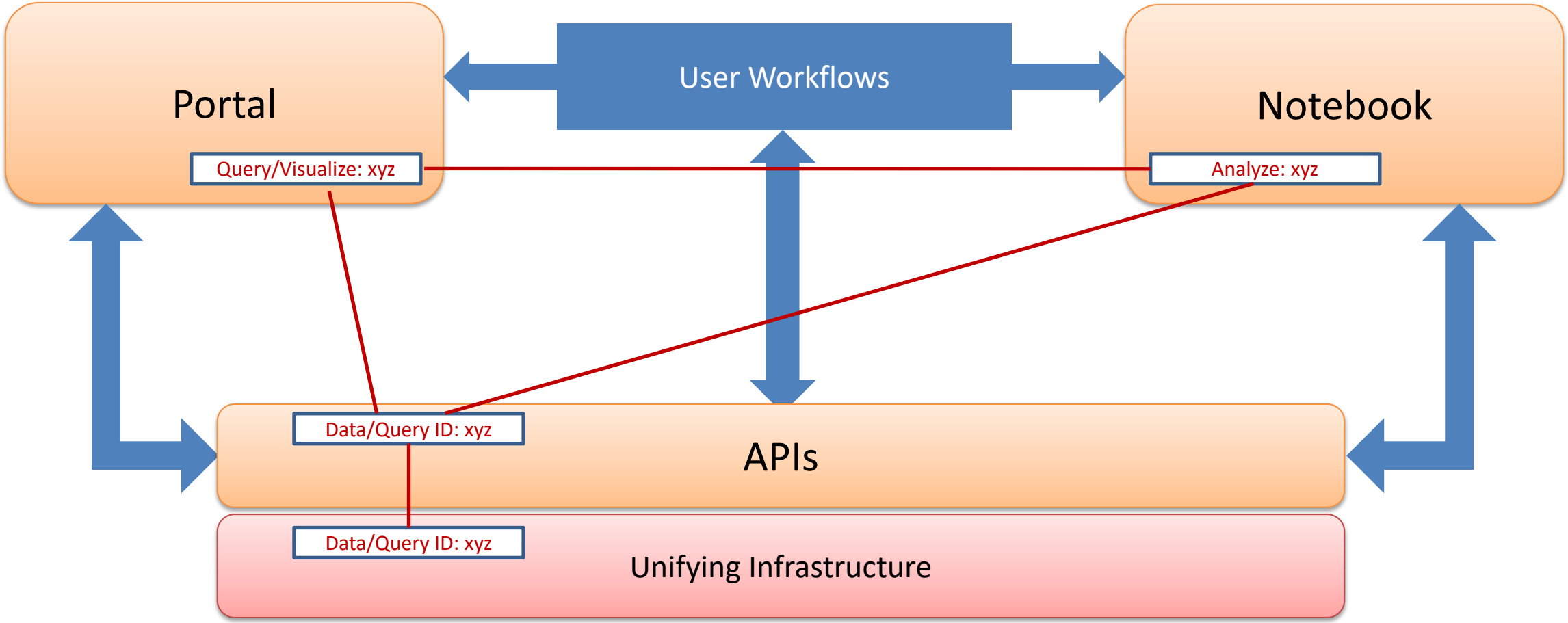
Science Platform Underpinnings



Technologies

- Portal
  - Based on an evolution of the open-source Firefly toolkit that is used in the IRSA archives
    - Native astronomical image visualization with data overlays
    - Tabular data viewing, 2D & 3D plotting, filtering, sorting, synthetic columns, with brushing and linking
    - All-sky visualization based on HiPS and MOC
  - Data exploration workflows illuminating the connections among the LSST data products
- Notebook
  - Based on JupyterHub (for login, session control) and JupyterLab
  - Container-based deployment of the LSST software stack
  - Python processes execute close to the data at the LSST Data Access Centers
- APIs
  - Preferring IVOA protocols (e.g., TAP, SODA, VOSpace) wherever feasible
  - May need some extensions for the richness of the data model or performance, still evaluating

Portal-Notebook Aspect Connections



- Connections enable sharing of data between components and therefore more complex workflows and analysis

Cross-Aspect Connections

- Users are not confined to one aspect or the other
  - The design provides simple data connections among all the aspects of the Science Platform
    - Enabled by the common Data Access API architecture and computing infrastructure
    - Enabled by the availability of interfaces allowing transfer of data and control from Aspect to Aspect
  - Find or create data in one aspect; view or analyze that data in another 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 ADQL 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 (the system remembers the queries you have performed, and caches recent results) or through simple UI actions (e.g., copy/paste of a query token between windows)
    - Asynchronous TAP is very useful as part of this
- Moving from data discovery to analysis
  - We expect to provide a “one click” means of launching a fresh notebook with pre-configured access to the results of data identified in the Portal

Cross-Aspect Connections

- 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
- The Notebook can access data from the Portal:
  - Results of Firefly operations (synthetic columns, interactive filtering/brushing) are retrievable via Python APIs in the notebook
  - Users can set up callbacks in the Notebook driven by extensible UI actions in the Portal
- LSST-aware visualizations developed for the Portal are all readily adaptable for use in the Notebook
  - Jupyter(Lab) widgets and extensions
  - Helps make the user experience common across aspects
  - But the whole ecosystem of community tools is available as well: Bokeh, datashader, matplotlib