Building a 100% cloud-based alert broker:

Pitt-Google

Troy Raen
University of Pittsburgh and the Pitt-Google Collaboration

LIneA Webinar | December 2, 2021

Pitt-Google Alert Broker Collaboration

Troy Raen
Lead Developer
Grad Student
University of Pittsburgh

on behalf of...

Michael Wood-Vasey
PI, Associate Professor
University of Pittsburgh

Ross Thomson
Solutions Architect
Google

Christine Mazzola Daher
Grad Student
University of Pittsburgh

Daniel Perrefort
Research Assistant Professor
University of Pittsburgh

Outline

1. **Pitt-Google broker's motivation**
2. **Comparison of broker approaches** and basics of the **Cloud model**
3. Pitt-Google **broker architecture** and the **Google Cloud services** we employ
4. Other Google Cloud **tools of interest**

Learning Objectives

1. Understand the Cloud model and cost/benefit tradeoffs.

2. Describe two Google Cloud services, and
   a. how Pitt-Google is using them to develop an LSST alert broker; and/or
   b. how you might use them with the LIneA IDAC

Pitt-Google broker's motivation

Pitt-Google Broker Motivation

Get the data into Google Cloud,

so that anyone and everyone

can access however much they want,

and do whatever they want with it.

Pitt-Google Broker Motivation

Get the data into Google Cloud,

so that anyone and everyone

can access however much they want,

and do whatever they want with it.

The LSST project is not scoped for this.

Pitt-Google Broker Motivation

Get the data into Google Cloud,

so that anyone and everyone

can access however much they want,

and do whatever they want with it.

Caveats:

- Internet access required (but no other hardware, software, etc.)
- Pay for what you use

Pitt-Google Broker Motivation

Get the data into Google Cloud,

so that anyone and everyone

can access however much they want,

and do whatever they want with it.

Goals:

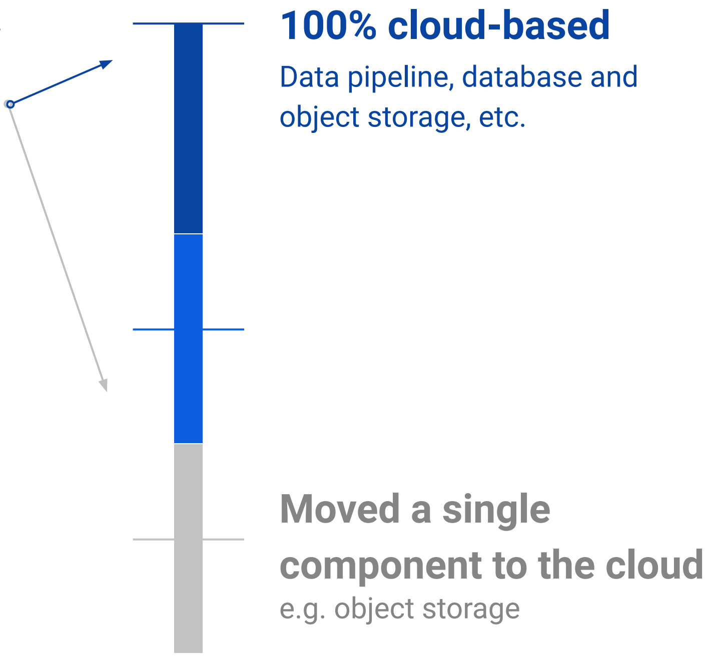
1. **Enable scientific collaboration.**
2. **Expose the data publicly at each step** of the processing pipeline, so users can access data that is pre-processed to a level of their choosing.
3. **Use Google Cloud *services*** to maximize efficiency and minimize the effort required to develop, run, and support an alert broker at LSST-scale.

Comparison of broker approaches and Basics of the Cloud model

Other brokers also run “in the cloud”...

Many other astronomical community alert brokers are also operating in a cloud environment.

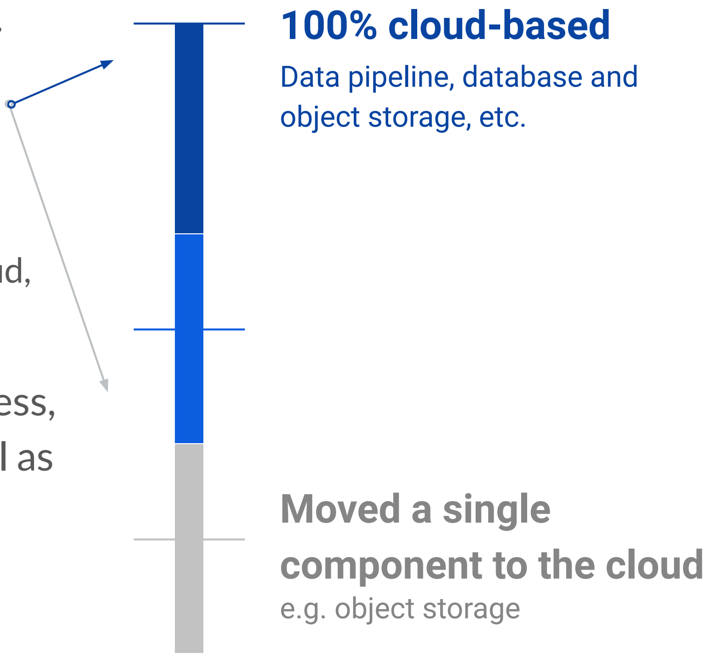
This may mean anything from



Other brokers also run “in the cloud”...

Many other astronomical community alert brokers are also operating in a cloud environment.

This may mean anything from



Fink, operating on VirtualData OpenStack-based cloud, cites **benefits of cloud computing** as:

“scalability of processing and cost effectiveness, fault tolerance, shareability of results, as well as increased portability and reproducibility”

(Möller et al., 2020)

Other brokers also run “in the cloud”... what makes Pitt-Google different?

A fundamentally different broker model based on

Google Cloud ***services***

and

event-driven processing.

Cloud Services

In cloud computing, users rely on the cloud provider to manage physical hardware and support access.

The level of support can vary greatly. Google Cloud examples:

Compute Engine (VMs):

- User writes the application code and manages the VM environment, containerization & deployment, scaling to meet demand, fault tolerance, etc.
- User has a lot of control, and a lot of responsibility.

Cloud Functions (serverless compute):

- User writes a function and deploys to the Cloud. Google manages everything else.
- User trades control and responsibility for ease-of-use.

They are both *services*,
but infrastructure management responsibilities are
split between Google and the user very differently .

Cloud Services

The Pitt-Google broker takes advantage of managed services, not VMs, as much as possible.
This results in a lightweight, highly modular, nearly serverless pipeline.
and no direct data storage or delivery responsibilities.

Jargon: Serverless

"***developers*** of serverless applications are ***not concerned with capacity planning, configuration, management, maintenance, fault tolerance, or scaling of containers, VMs, or physical servers***"

https://en.wikipedia.org/wiki/Serverless_computing

Broker high-level design models

Run the end-to-end pipeline on a *single device*

Server, VM, or Raspberry Pi

May not include persistent storage.

Package the broker in *multiple containers*, and run on *clusters of VMs*

Containers may include:

- Data pipeline
- Persistent storage
- Communication + stream ingestion

Run the broker on *cloud services*

Services may include:

- Modular data processing
- Persistent storage
- Communication
- VM management
- User accounts and data access

Types of software and services brokers use

Software the user directly downloads, manages, and runs

Apache Cassandra, Apache Kafka, Apache Spark, MongoDB, Kubernetes

Service that manages and runs traditional software in the cloud

MongoDB Atlas, Google Kubernetes Engine

Service built explicitly for cloud computing at massive scale

Google Pub/Sub, BigQuery, Cloud Run

Relationship between design choice and software/service

Software/
Service

Software the user directly downloads, manages, and runs

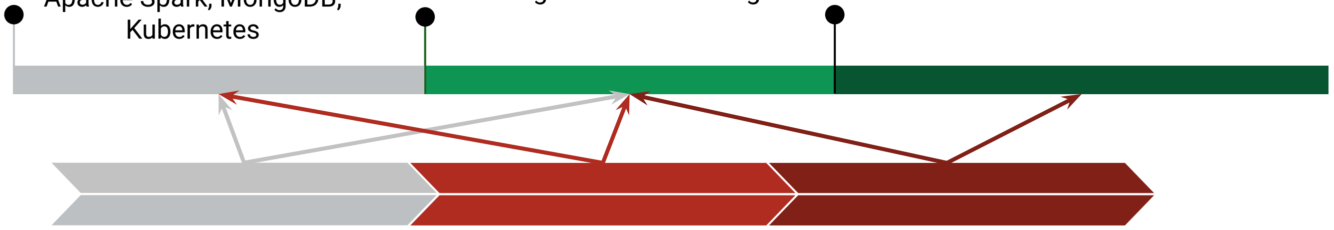
Apache Cassandra, Apache Kafka, Apache Spark, MongoDB, Kubernetes

Service that manages and runs traditional software in the cloud

MongoDB Atlas, Google Kubernetes Engine

Service built explicitly for cloud computing at massive scale

Google Pub/Sub, BigQuery, Cloud Run



Design

Run the end-to-end pipeline on a *single device*

Server, VM, or Raspberry Pi

May not include persistent storage.

Package the broker in *multiple containers*, and run on *clusters of VMs*

Containers may include:

- Data pipeline
- Persistent storage
- Communication + stream ingestion

Run the broker on *cloud services*

Services may include:

- Modular data processing
- Persistent storage
- Communication
- VM management
- User accounts and data access

Relationship between design choice and software/service

Software/ Service

Software the user directly downloads, manages, and runs

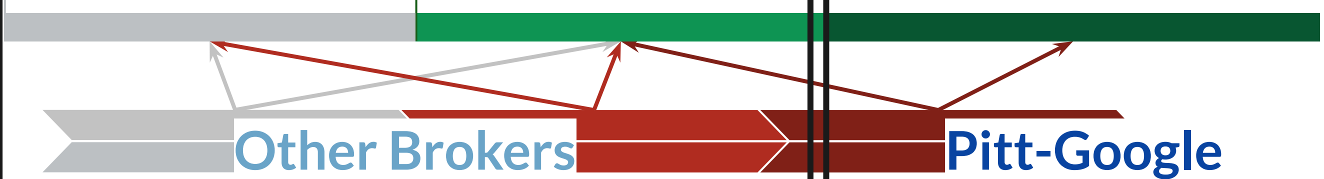
Apache Cassandra, Apache Kafka, Apache Spark, MongoDB, Kubernetes

Service that manages and runs traditional software in the cloud

MongoDB Atlas, Google Kubernetes Engine

Service built explicitly for cloud computing at massive scale

Google Pub/Sub, BigQuery, Cloud Run



Design

Run the end-to-end pipeline on a *single device*

Server, VM, or Raspberry Pi

May not include persistent storage.

Package the broker in *multiple containers*, and run on *clusters of VMs*

Containers may include:

- Data pipeline
- Persistent storage
- Communication + stream ingestion

Run the broker on *cloud services*

Services may include:

- Modular data processing
- Persistent storage
- Communication
- VM management
- User accounts and data access

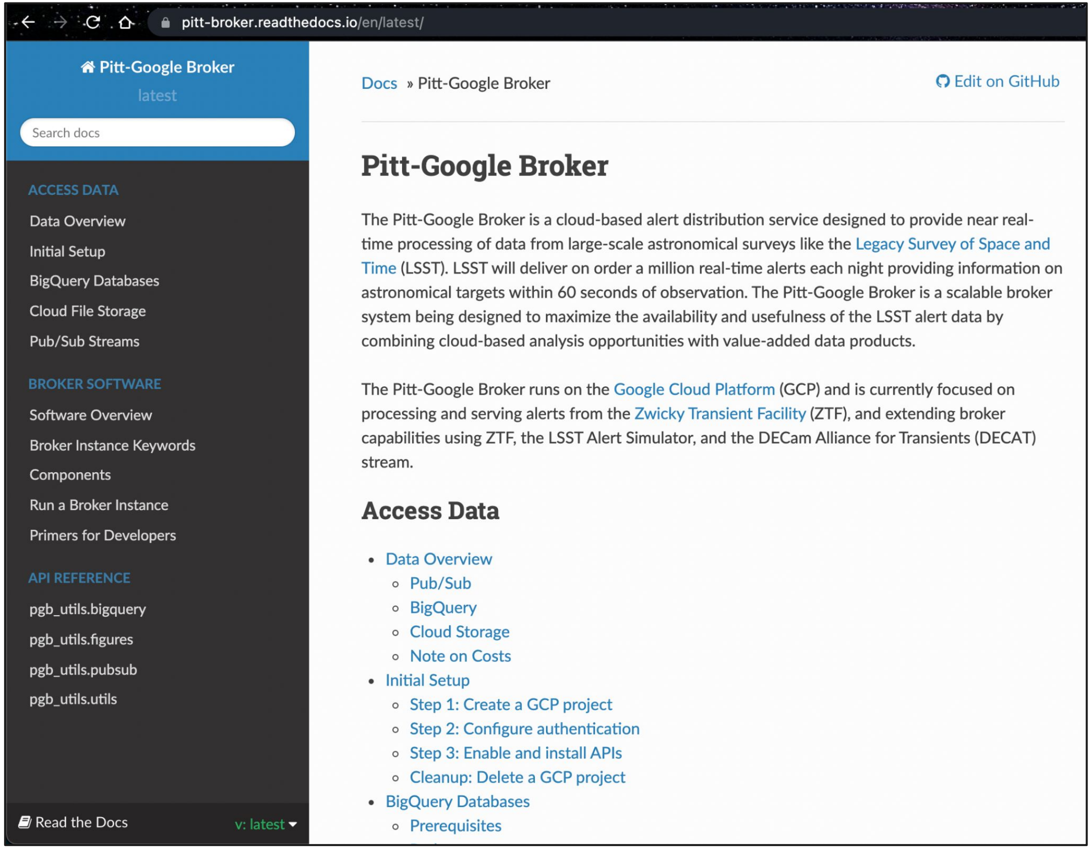
Data Access from Google Cloud Services

- Pitt-Google stores data in Google Cloud services (BigQuery, Pub/Sub, and Cloud Storage).
- Google manages the data storage, data delivery, and user accounts & credentials.
- To access data, one first creates an account and obtains credentials from Google Cloud.
- By contrast, other brokers manage their own data storage/delivery and user accounts/access. Their users contact the broker for data access, even if the broker is running in the cloud.

How? Methods to access data from Google Cloud:

1. APIs developed and supported by Google. Access from anywhere, in or out of Google Cloud.
2. Within Google Cloud, many of their tools and services are integrated, making it easy to write streaming data pipelines or compute/analyze data at the source, without having to think about servers or data management.

Data Access from Google Cloud Services

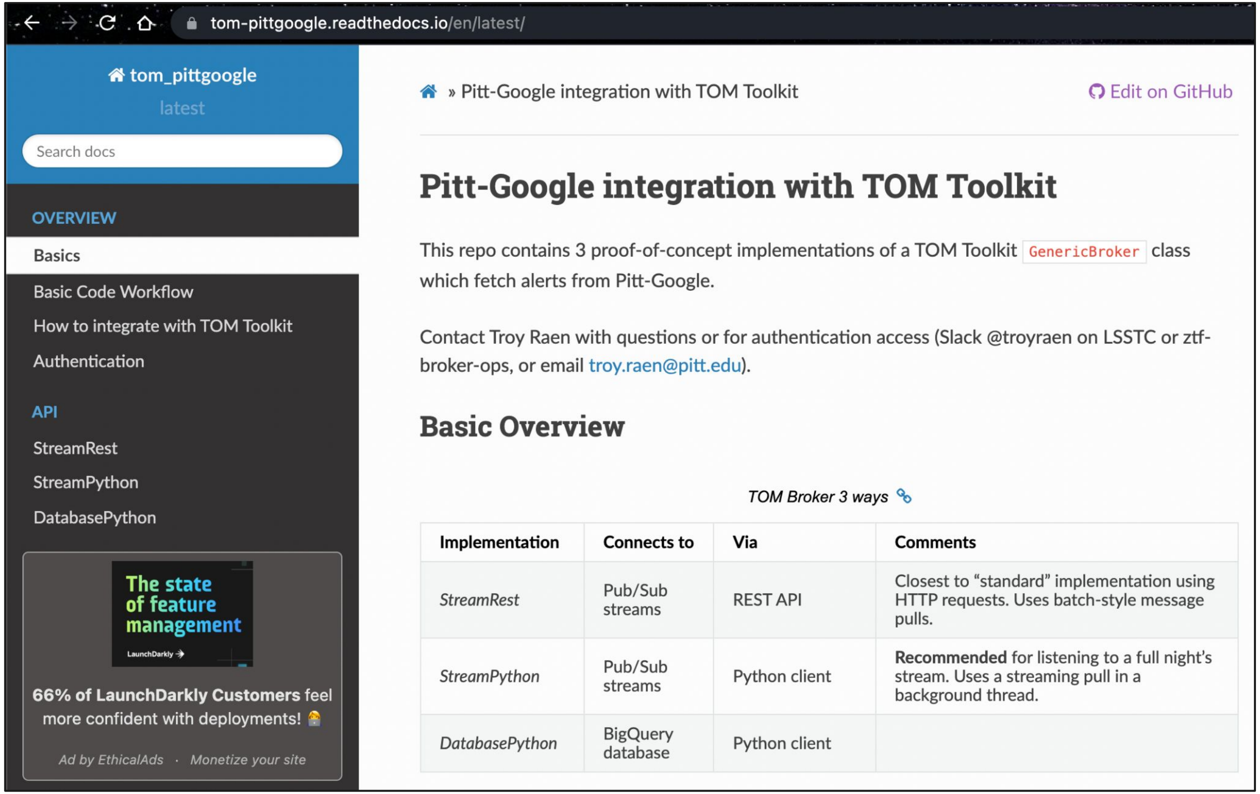


Pitt-Google does not have to write its own APIs.

(And end-users do not have to rely on Pitt-Google to write and maintain a good API.)

We do, however, provide tutorials and Python wrappers of Google APIs for accessing Pitt-Google data resources.

Data Access from Google Cloud Services



tom-pittgoogle.readthedocs.io/en/latest/

tom_pittgoogle
latest

Search docs

OVERVIEW

Basics
Basic Code Workflow
How to integrate with TOM Toolkit
Authentication

API

StreamRest
StreamPython
DatabasePython

» Pitt-Google integration with TOM Toolkit   Edit on GitHub

Pitt-Google integration with TOM Toolkit

This repo contains 3 proof-of-concept implementations of a TOM Toolkit `GenericBroker` class which fetch alerts from Pitt-Google.

Contact Troy Raen with questions or for authentication access (Slack @troyraen on LSSTC or ztf-broker-ops, or email troy.raen@pitt.edu).

Basic Overview

TOM Broker 3 ways

Implementation	Connects to	Via	Comments
StreamRest	Pub/Sub streams	REST API	Closest to “standard” implementation using HTTP requests. Uses batch-style message pulls.
StreamPython	Pub/Sub streams	Python client	**Recommended** for listening to a full night’s stream. Uses a streaming pull in a background thread.
DatabasePython	BigQuery database	Python client	

REST and client APIs make integration with external services straightforward.

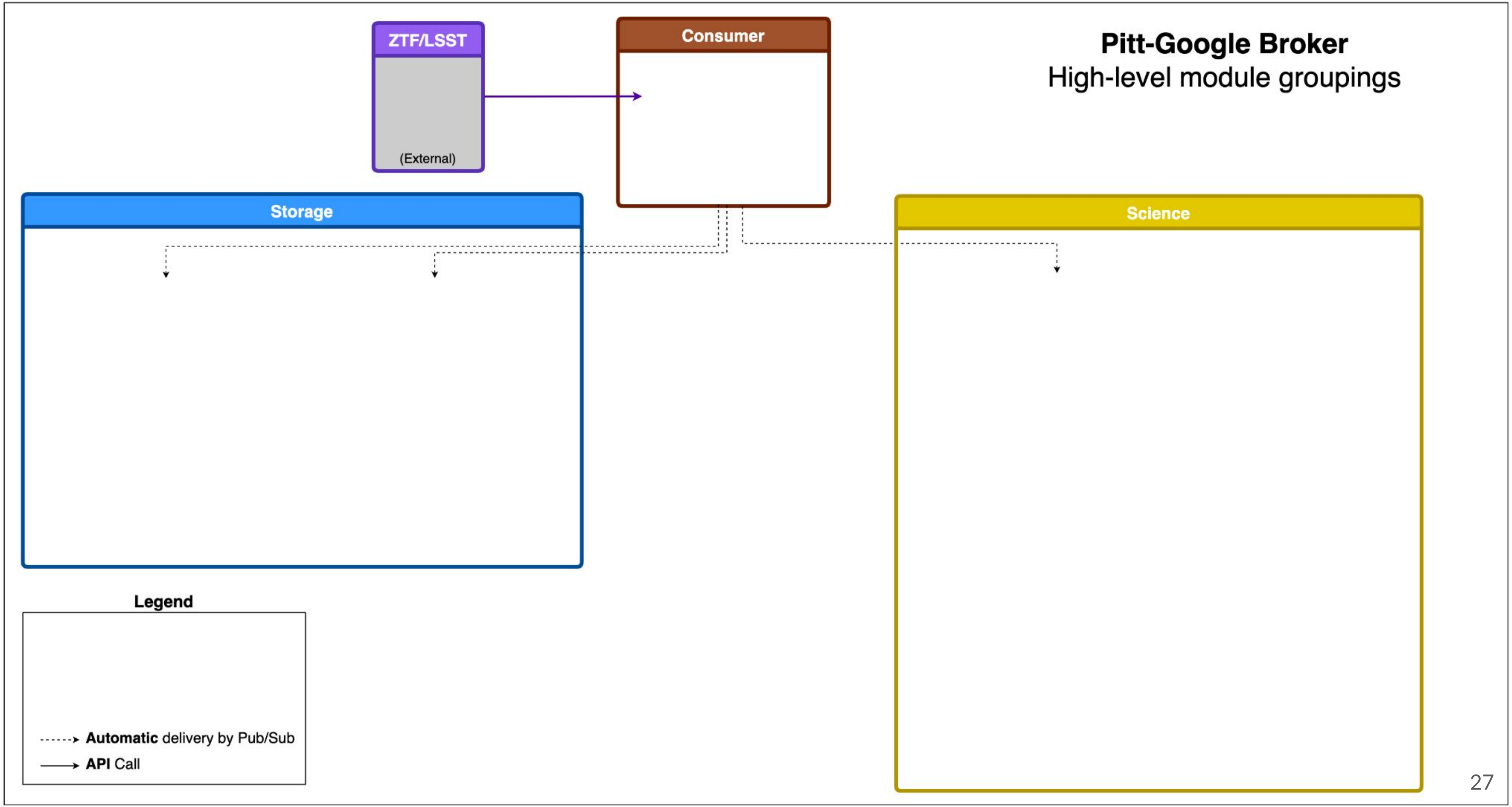
Google Cloud quotas and limits

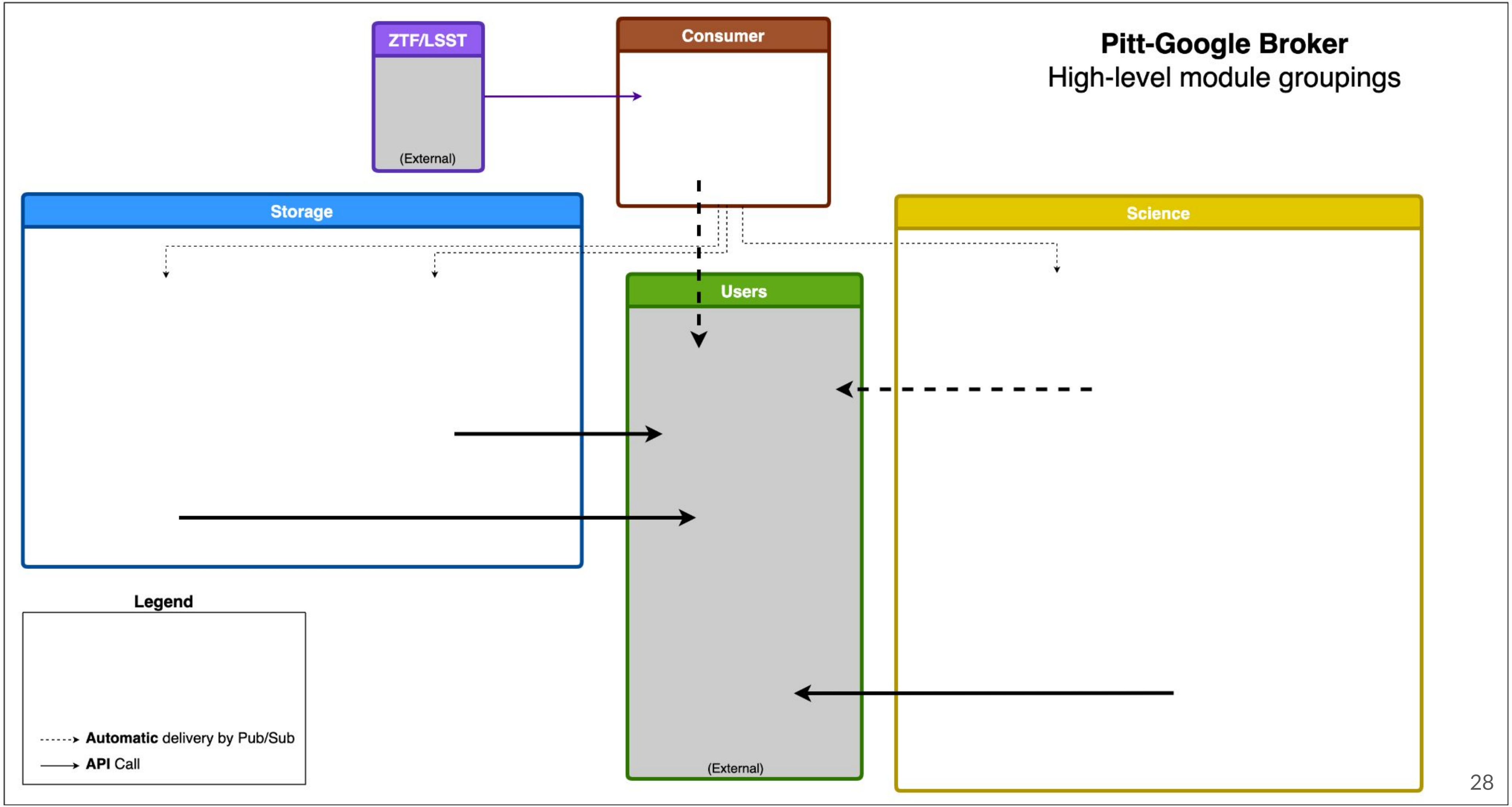
- There are quotas and limits on how much you can use. Throughput, API calls, startup time, etc.
- They are broken down into a fine level of detail.
- These are generally high, but some may be an issue at LSST-scale.
- Some you can request to increase, some you can't.
- An example:
  - https://cloud.google.com/bigquery/quotas: Maximum number of concurrent interactive queries: 100

Google Cloud cost model

- **Free to connect, test things out, run ongoing analysis at low volume** (low from LSST perspective)
  - $300 credit for new accounts
  - Free tier quotas renew monthly for everyone (e.g., BigQuery: The first 1 TB of query data processed per month is free.)
- Beyond the free stuff, **everyone pays for what they use.** Pay as you go. No upfront costs.
  - Example: We pay to store data in BigQuery, and query it ourselves. Our end-users pay for their own queries on data we have stored.
- Costs broken down to a fine level of detail. Some examples:
  - https://cloud.google.com/**bigquery**/pricing: The first 1 TB of query data processed per month is free, after which it's $5.00 per TB.
  - https://cloud.google.com/**run**/pricing: First 180,000 vCPU-seconds free per month, after which it's $0.00002160 / vCPU-second (Tier 2, Sao Paulo, Brazil).
  - Watch out for egress charges. Consider computing next to the data (in the Cloud).
- Tradeoffs:
  - Paying a cloud provider to manage infrastructure, run your jobs, and store your data.
  - Not paying to purchase/manage hardware; network bandwidth; personnel time to plan, develop, manage, and support most of the infrastructure.
  - Organizational budgets may be scoped for some things, but not others.

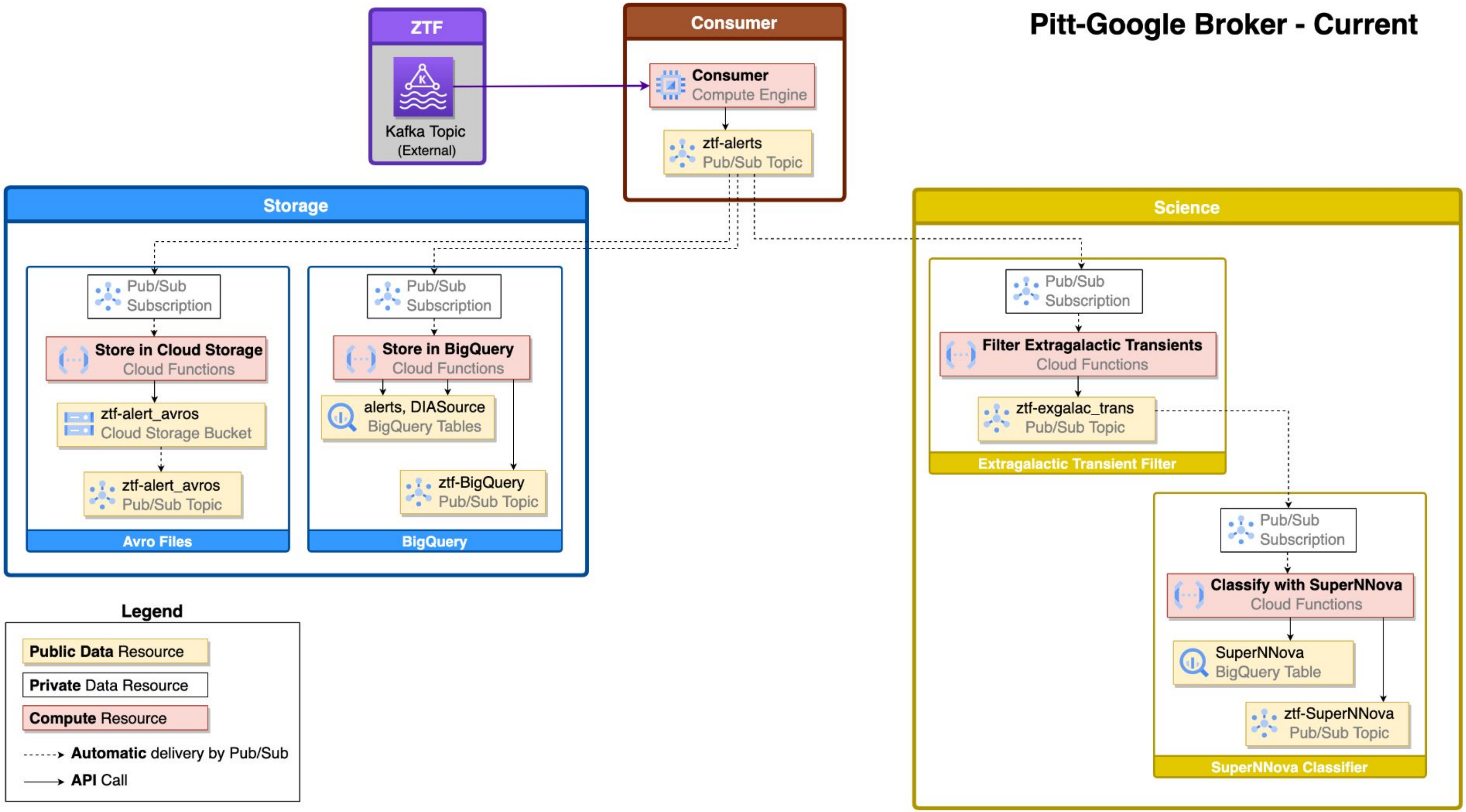
Pitt-Google Broker's Architecture and Google Cloud Services


Pitt-Google Broker
High-level module groupings
ZTF/LSST
(External)
Consumer
Storage
Science
Legend
Automatic delivery by Pub/Sub
API Call

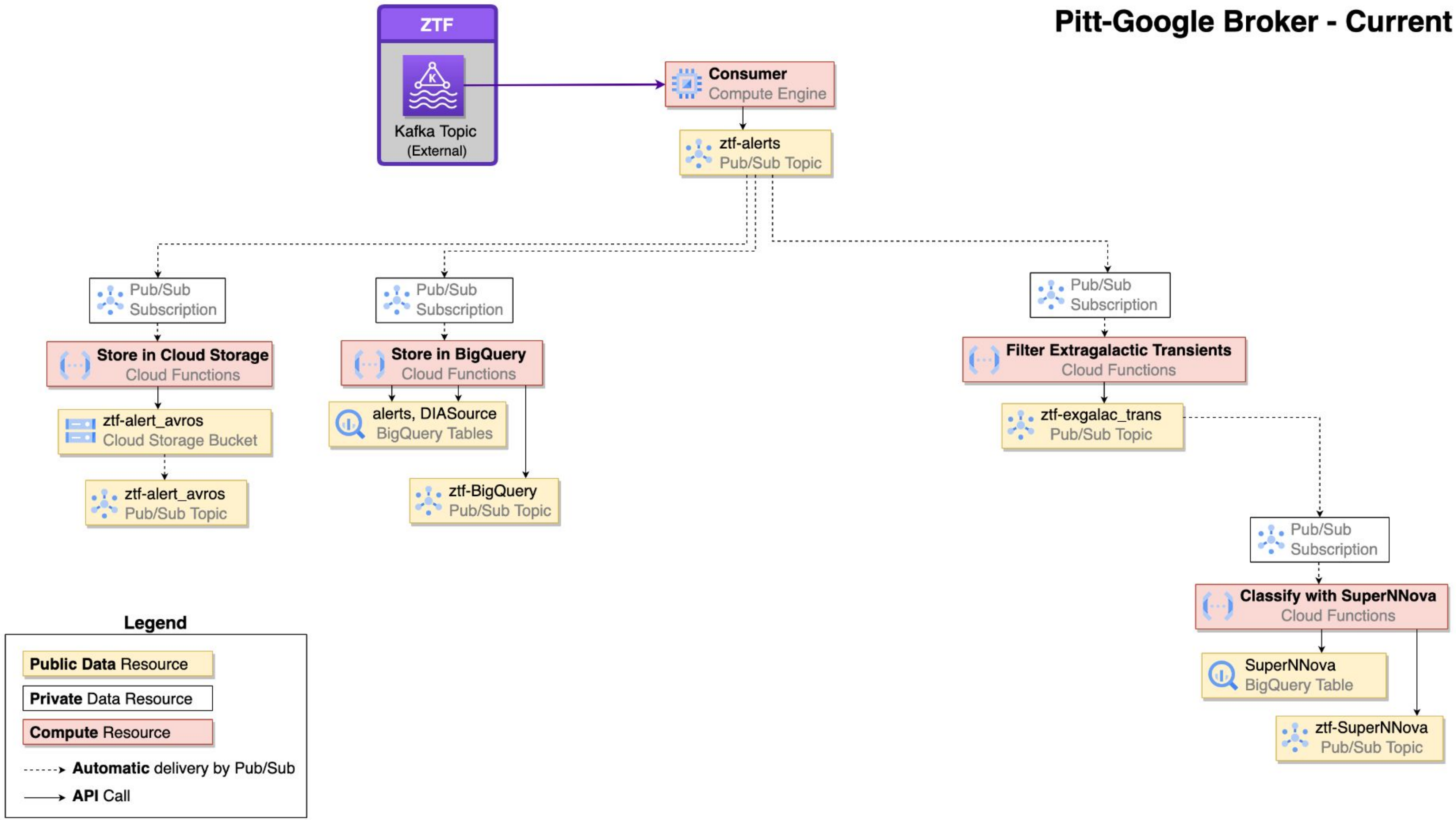


Pitt-Google Broker
High-level module groupings
ZTF/LSST
(External)
Consumer
Storage
Science
Users
(External)
Legend
Automatic delivery by Pub/Sub
API Call

Pitt-Google Broker - Current



Pitt-Google Broker - Current



Event-driven microservices model

Jargon: Microservices

"is an architectural style that ***structures an application as a collection of services*** that are ...
loosely coupled, ***independently deployable***, ..."

https://microservices.io/

"Events" are alerts or messages.

Per-message processing. No batching.

Benefits: Fast. Lightweight (no VM/cluster management). Per-alert processing easy to think about and develop. Fault-tolerant (failures automatically redelivered by Pub/Sub, other messages not affected). Scales (from zero) automatically and nearly instantaneously to meet demand.

Drawbacks: Cross matches are a challenge within the pipeline. Can't query an external service like CDS 10 million times per night. Per-alert lookups in BigQuery are expensive.

Pub/Sub

https://cloud.google.com/pubsub/docs/overview

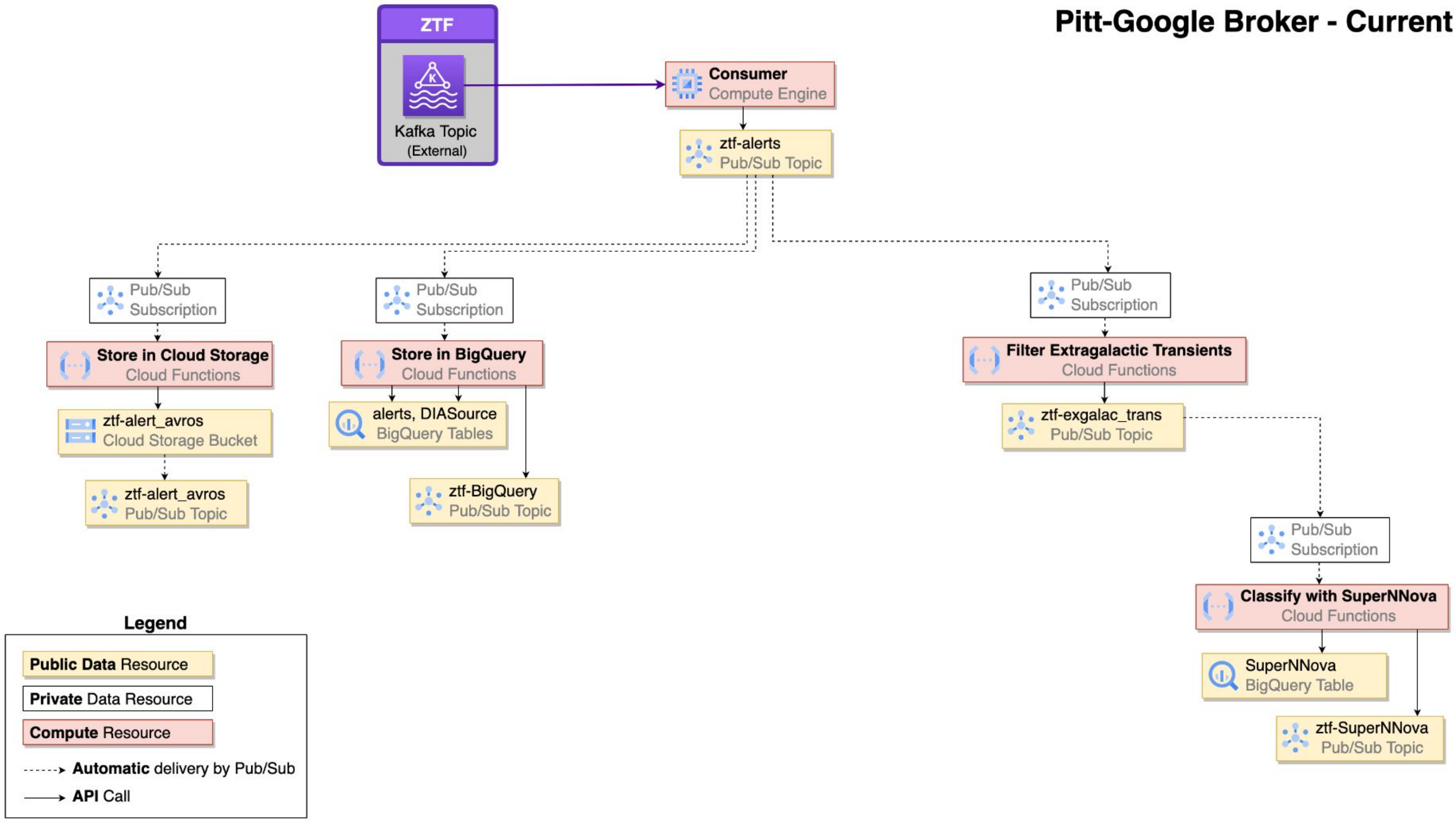
- Pub/Sub is a streaming message service.
- Asynchronous publish-subscribe. Publishers are decoupled from subscribers.
- Messages retained in subscription for (max) 7 days, or until acknowledged by subscriber.
- **Push** to any HTTP endpoint for event-driven processing/storage.
  - Google Cloud Functions and Cloud Run
  - Dataflow
  - AWS Lambda (should be doable, but haven't tried it yet)
- **Pull** from anywhere with network access. (APIs: Python, REST, gRPC, CLI, ...).
- Pricing basics:
  - Throughput: First 10 GiB/month is free. After that, $40 /TiB.
  - Egress for throughput that crosses Google Cloud regions. $0.08 - $0.12 per GB in North and South America
  - No storage fees, unless you turn on special retention options.
  - https://cloud.google.com/pubsub/pricing

BigQuery

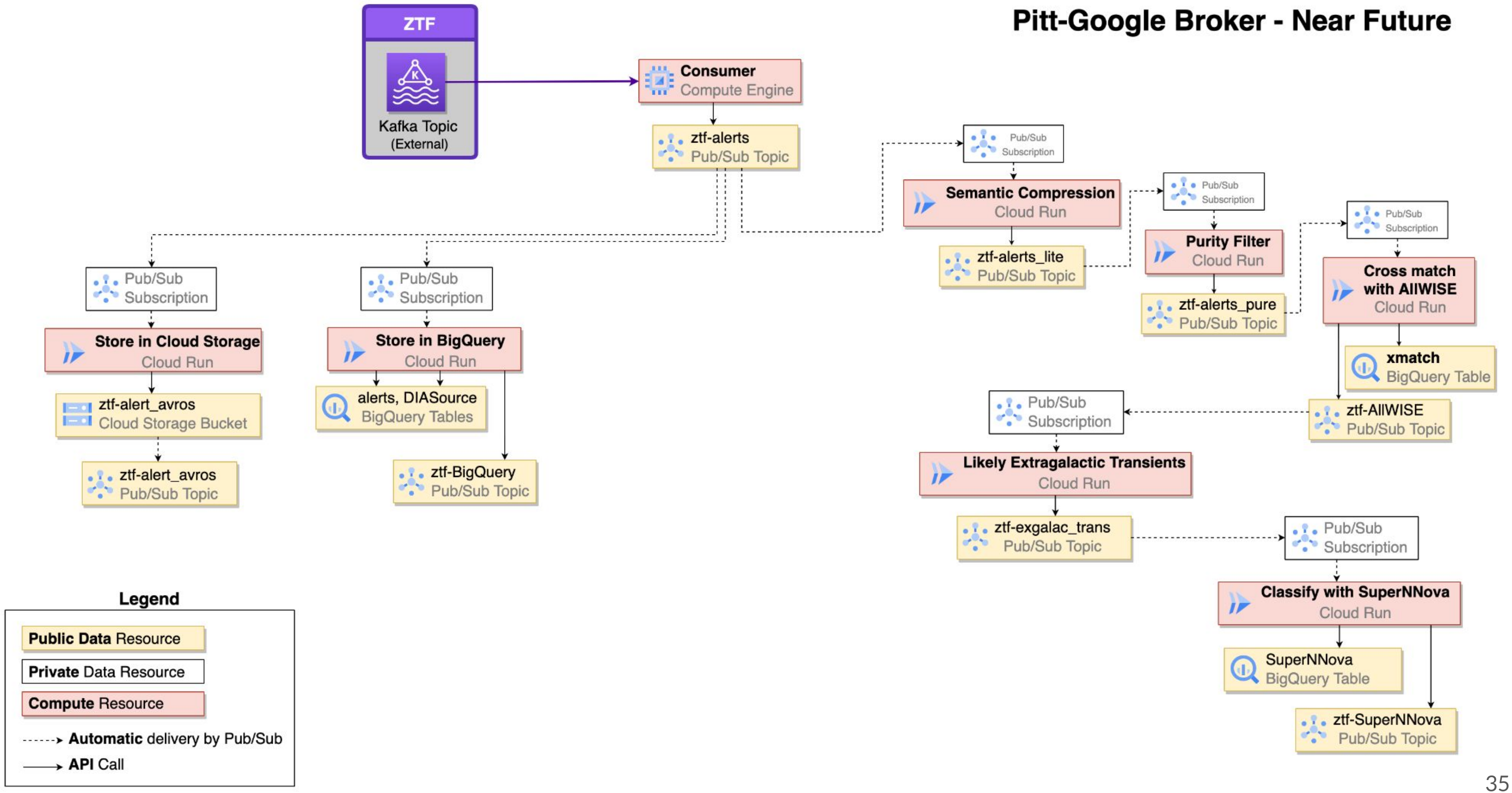
https://cloud.google.com/bigquery/docs/introduction

- BigQuery is a fully managed data warehouse. (can analyze external data sources)
- SQL queries (ANSI:2011 compliant)
- Columnar storage, optimized for complex analytical queries on large datasets
- Supports views & materialized views, and user-defined functions
- Streaming row inserts (data available to query within seconds), or batch load
- Full API access
- Built-in GIS, which we can exploit for positional matching with static catalogs… but…
- No indexes
  - Does a full column scan for every query. Single point lookups are expensive. Can reduce this by partitioning and clustering the tables
  - Cross matching within the event-driven pipeline is challenging. Currently we are working on adding a spatial index to AllWISE catalog and clustering. Then will test (HEALPix. Also explored HTM)
- https://cloud.google.com/bigquery/pricing: First TB of query data processed per month is free, then $5.00 per TB. Active storage: first 10 GB/month is free, then $0.02 per GB.

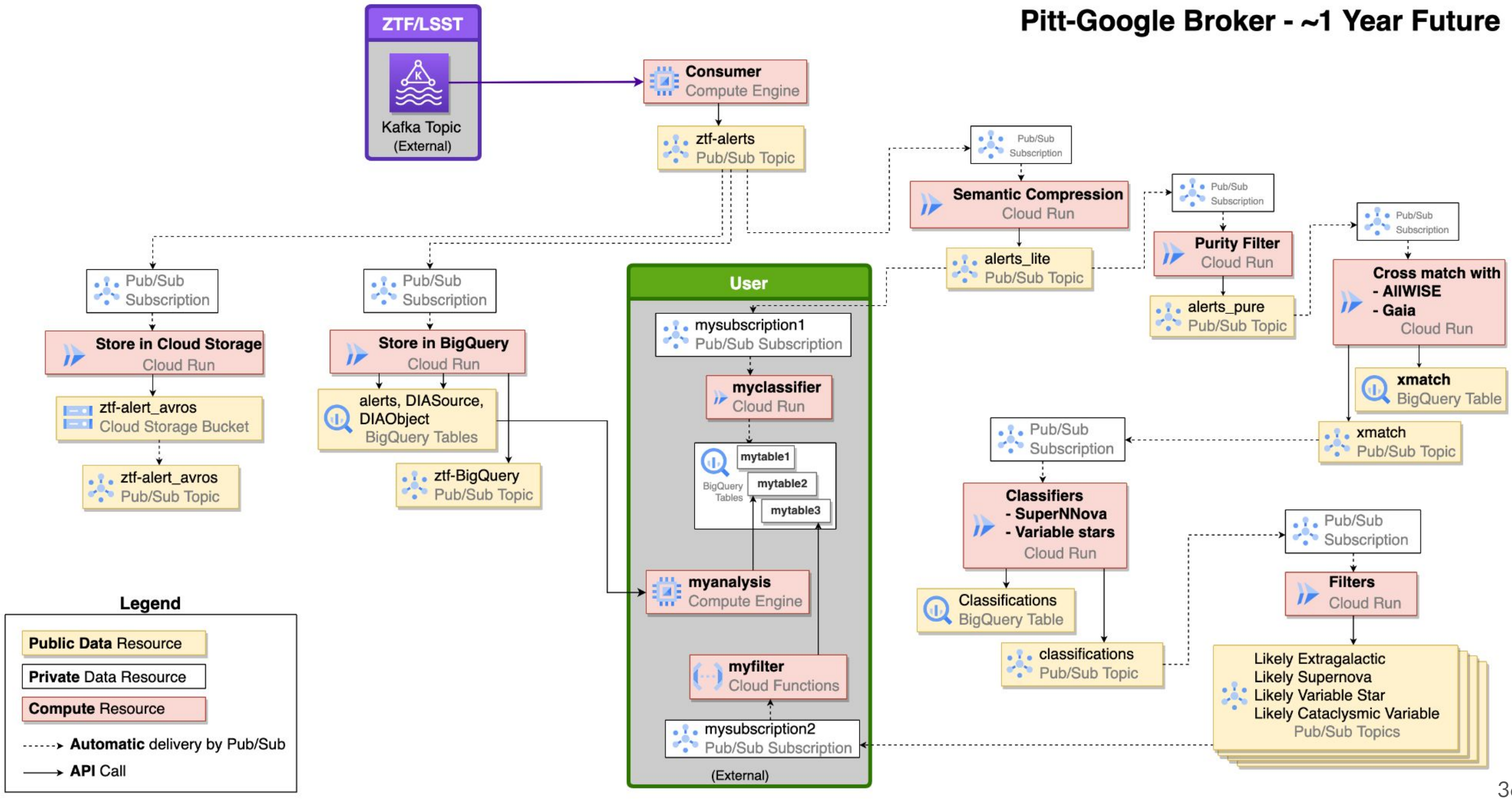
Pitt-Google Broker - Current



Pitt-Google Broker - Near Future



Pitt-Google Broker - ~1 Year Future



Filters on Pub/Sub streams

Reason: Ease analysis by curating streams to deliver only the subset of alerts that are likely of interest to a particular researcher/working group.

Options:

1. Pitt-Google broker runs a filter producing a curated stream.
2. End user places a filter directly on a subscription.
   a. Restrictions: Filter can only access data explicitly placed in message attributes, not the full alert packet . A limited set of operators is available (e.g.: = and !=, but not > or <)
3. End user listens to a stream and runs their own filter.
   a. Simple to implement using Cloud Functions/Run

Pitt-Google will:

1. Produce multiple curated streams supporting research in LSST's major science goals.
2. Support filters directly on subscriptions by thinking carefully about which data to expose and how.
3. Enable users running their own filters by providing tutorials and guidance.

Cloud Functions and Cloud Run

Cloud Functions

https://cloud.google.com/functions/docs

Serverless, highly scalable, **functions** as a service.

Triggered by Pub/Sub messages (or HTTP requests). Event-driven processing.

Max runtime for single invocation: 9 minutes

No runtime environment management. Limited to (Python runtime available)

A single instance can process one request at a time. Number of instances automatically scales to handle all incoming requests.

Cloud Run

https://cloud.google.com/run/docs

Serverless, highly scalable, compute to run **containers**.

Triggered by requests. Event-driven processing.

Max runtime for single invocation: 60 minutes.

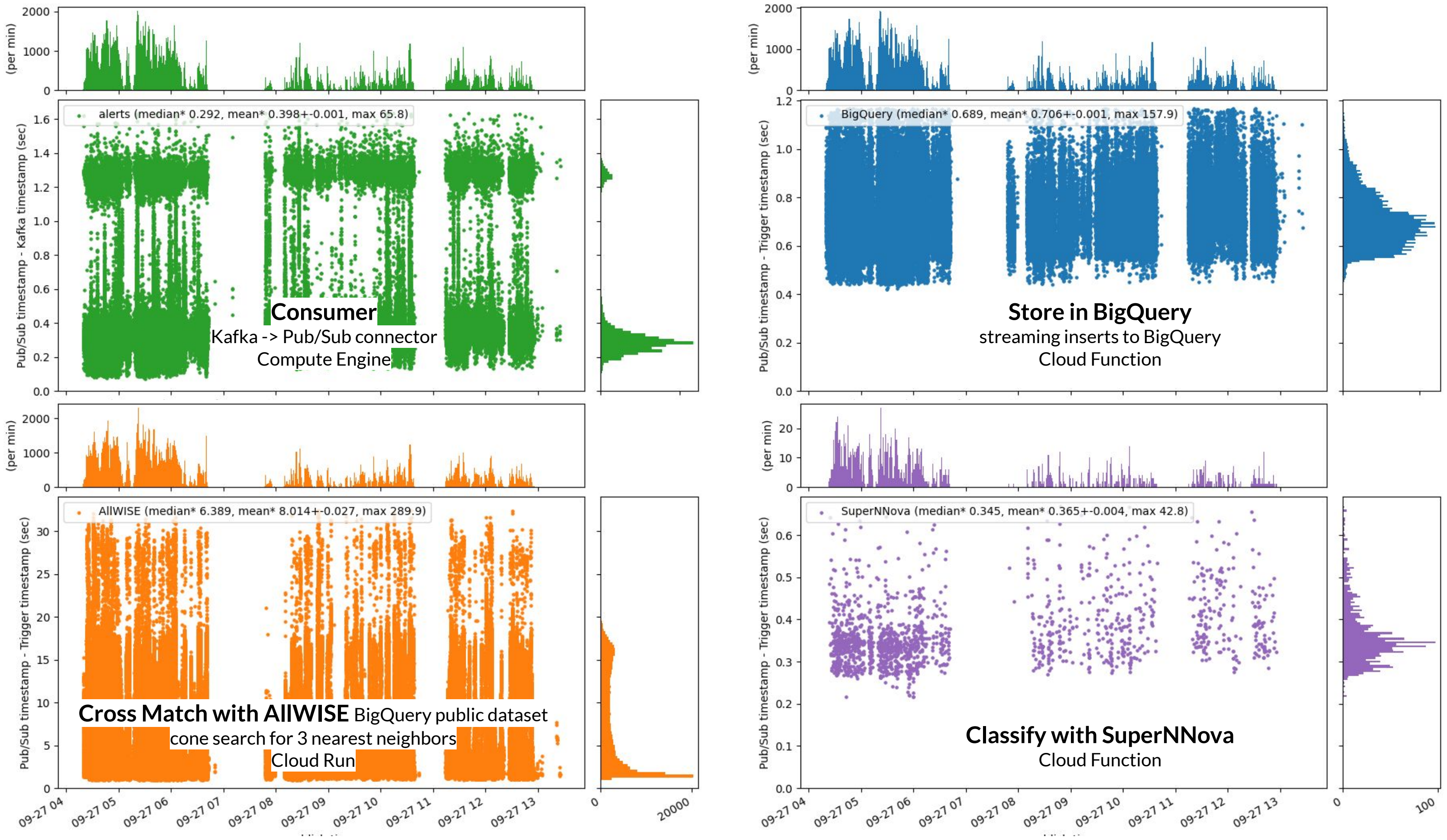
Any environment, language.

A single container instance can process up to 1000 concurrent requests. Number of instances automatically scales to handle all incoming requests.

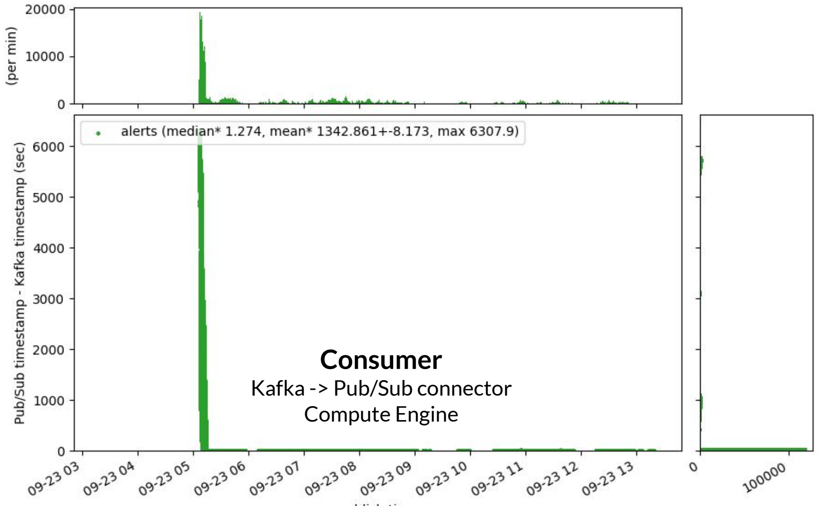
Consumer: Kafka -> Pub/Sub Connector

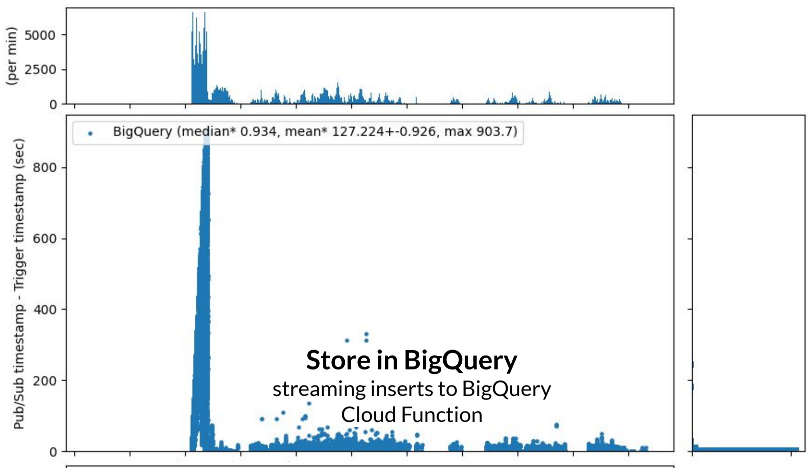
- Single "g1-small" VM
  - 0.5 vCPU
  - 1.70 GB memory
  - ~$8/month
- Running a Kafka Connect plugin
  - Passes alert bytes straight through to Pub/Sub, no decoding. Also passes message attributes.

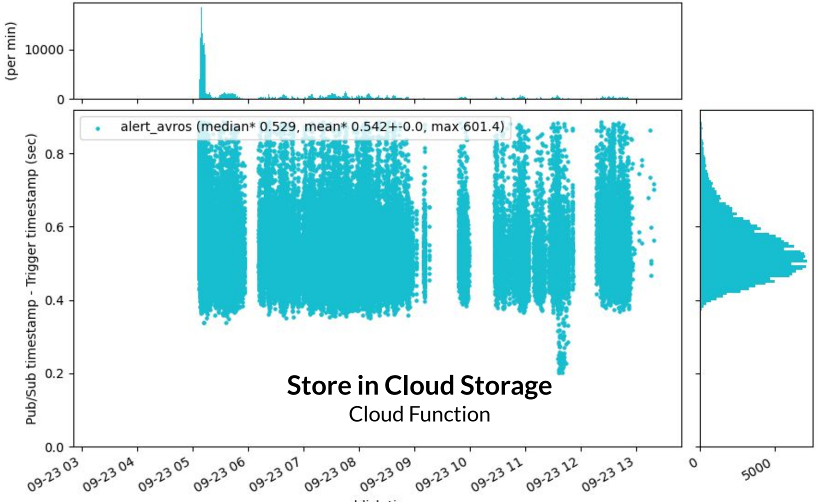
Module processing times on ZTF stream



Module processing times at LSST rates (briefly)







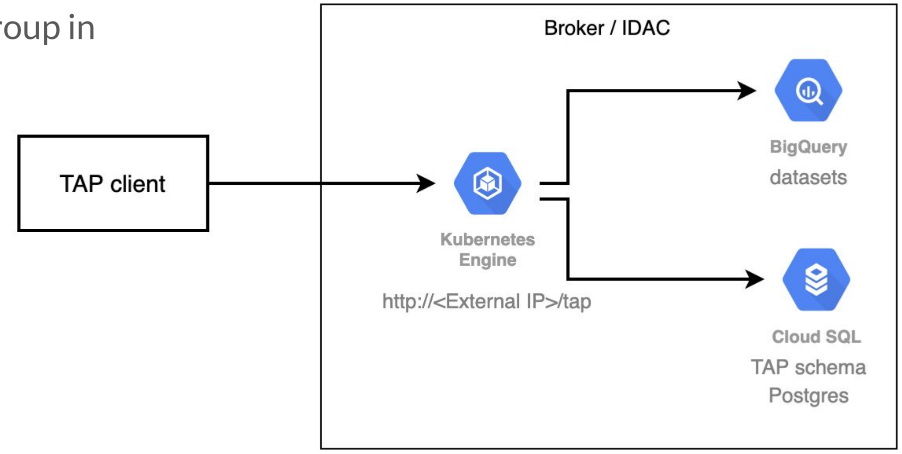
Other Google Cloud Tools

Things I’d like to explore.
Things that may be of use to the LIneA IDAC.

BigQuery + TAP and ADQL

TAP and ADQL for BigQuery was announced in the 235th AAS Meeting by [J. Thomson et al. (2020)]

Currently under development by a private group in collaboration with Google.



International Virtual Observatory Alliance
Table Access Protocol
Astronomical Data Query Language
Used in the Rubin Science Platform

BigQuery ML

https://cloud.google.com/bigquery-ml/docs/introduction

“create and execute machine learning models in BigQuery using standard SQL queries”

Compute at the data.

BigQuery Public Dataset
London bicycle hires k-means example.
https://cloud.google.com/bigquery-ml/docs/kmeans-tutorial

Run the **CREATE MODEL** query

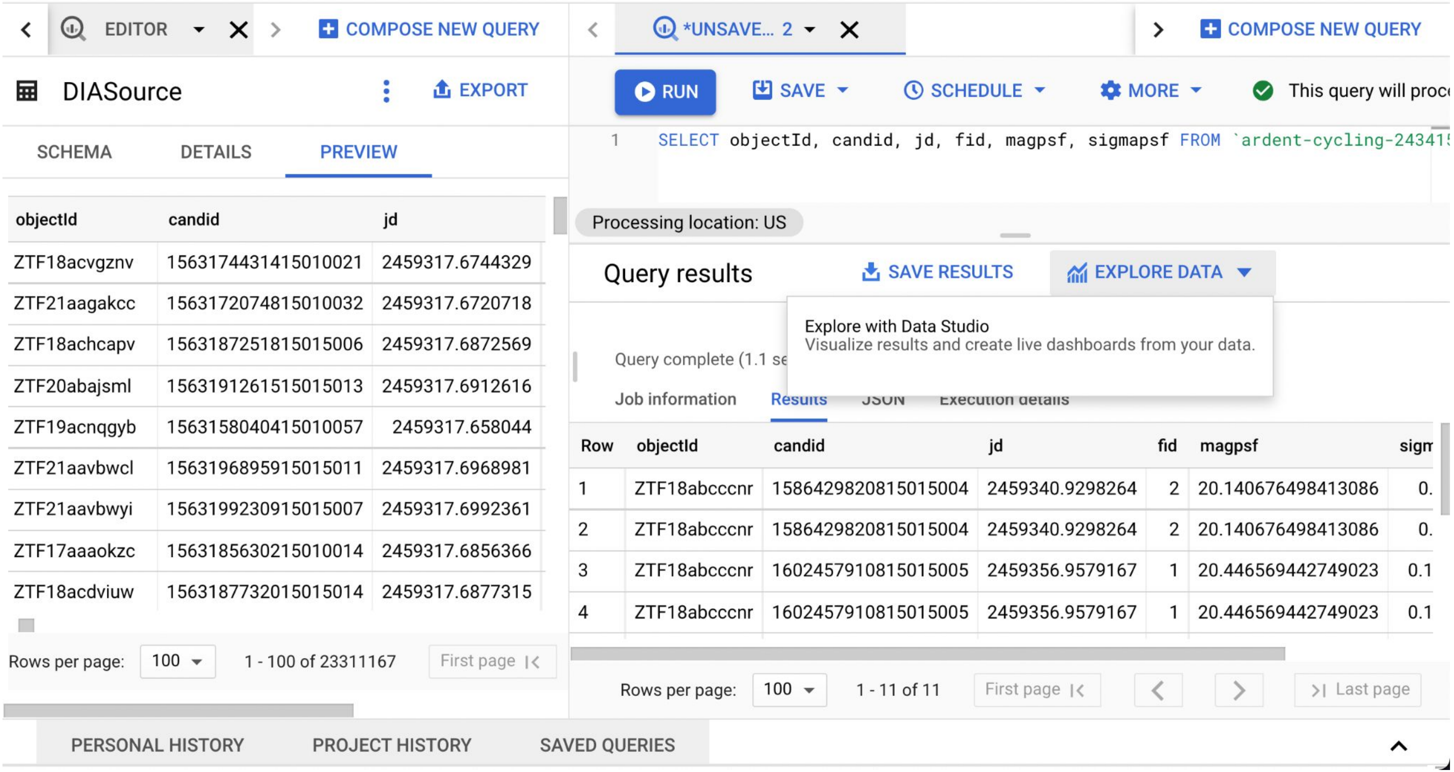
```
CREATE OR REPLACE MODEL
  bqml_tutorial.london_station_clusters OPTIONS(model_type='kmeans',
    num_clusters=4) AS
WITH
  hs AS (
  SELECT
    h.start_station_name AS station_name,
  IF
    (EXTRACT(DAYOFWEEK
      FROM
        h.start_date) = 1
      OR EXTRACT(DAYOFWEEK
      FROM
        h.start_date) = 7,
      "weekend",
      "weekday") AS isweekday,
    h.duration,
    ST_DISTANCE(ST_GEOGPOINT(s.longitude,
        s.latitude),
      ST_GEOGPOINT(-0.1,
        51.5))/1000 AS distance_from_city_center
  FROM
    `bigquery-public-data.london_bicycles.cycle_hire` AS h
  JOIN
```

BigQuery + Data Studio

https://cloud.google.com/bigquery/docs/visualize-data-studio

Pitt-Google dataset.

Dwarf nova lightcurve example.

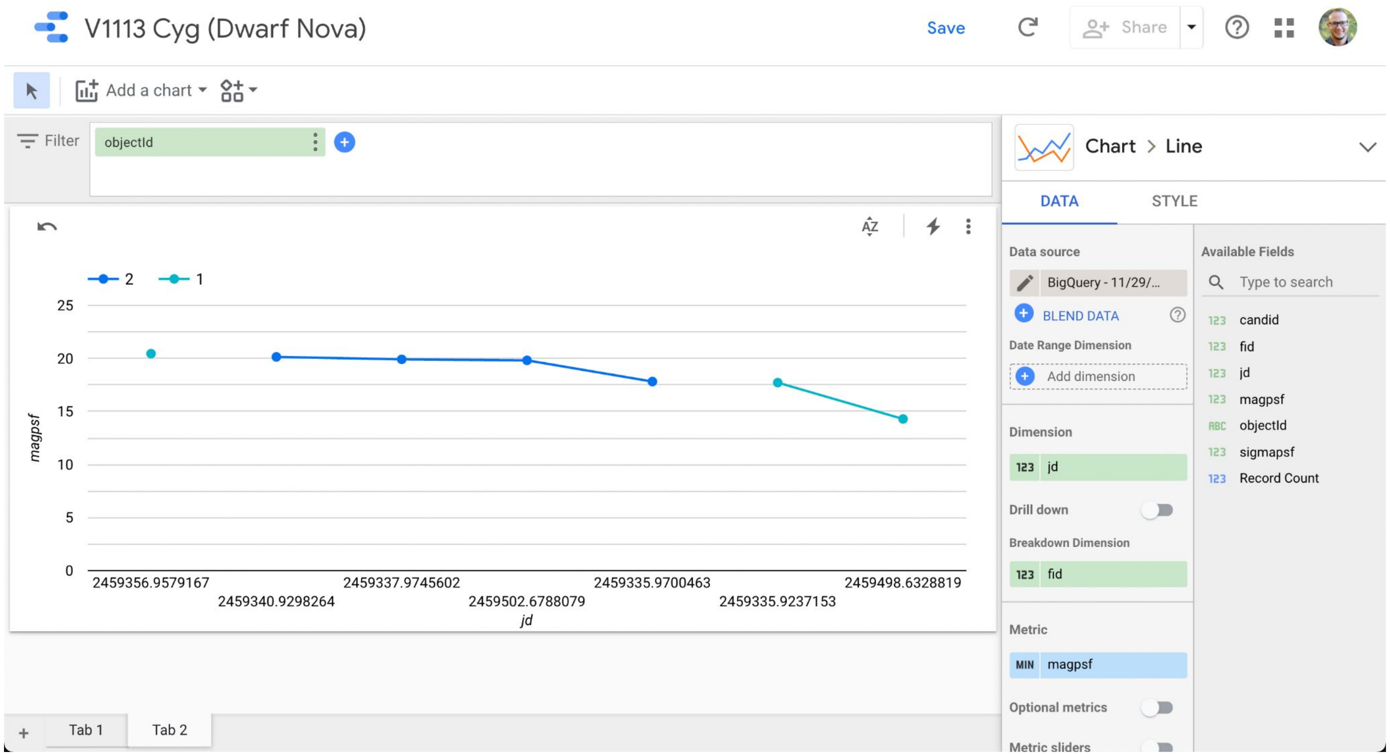


BigQuery + Data Studio

https://cloud.google.com/bigquery/docs/visualize-data-studio

Pitt-Google dataset.

Dwarf nova lightcurve example.



Summary

A selection of takeaways

Google Cloud Model:

- Services
  - Let Google handle hardware, scaling, fault tolerance, data storage & delivery… especially at LSST scales
  - You focus writing your data analysis
- BigQuery: Fully managed data warehouse. SQL access. Optimized for complex analytic queries at petabyte-scale.
- Costs:
  - Everyone pays for what they use.
  - Beware of egress; consider compute next to the data instead.
  - Budget scopes: Paying cloud provider for services, not humans in your organization for their time/effort.

Pitt-Google alert broker:

- Motivation:
  - Enable scientific collaboration by making all data available to everyone via well-developed and stable tools for data analysis/deliver/storage at this scale.
  - Support manageability via light filtering and value added data (cross match and classification).
- Taking advantage of Google Cloud services to simplify broker design and management.
- Event-driven data pipeline using Pub/Sub, Cloud Run/Functions, BigQuery.
  - Fast, fault tolerant, scales automatically, easy to build and manage.