



A Scalable and Open Data Platform for Data Exploitation

GSAW 2020
Loic Coulet, Kratos

© 2020 by . Published by The Aerospace
Corporation with permission.



Agenda

- Introduction
- What is a Data Exploitation Platform?
- Architecture
- Security
- Openness
- Use Cases
- Conclusion

Who am I?

- Working at Kratos since 2002
- Toulouse office (France)
- Data architect & solution manager

loic.coulet@kratosdefense.com



The image features a dense, intricate network of glowing blue lines and nodes, resembling a complex data structure or a global communication network. This network is superimposed over a blurred, high-angle view of a city skyline at night, with various buildings and lights visible in the background. The overall aesthetic is futuristic and technological, emphasizing the theme of data exploitation.

On Data Exploitation...

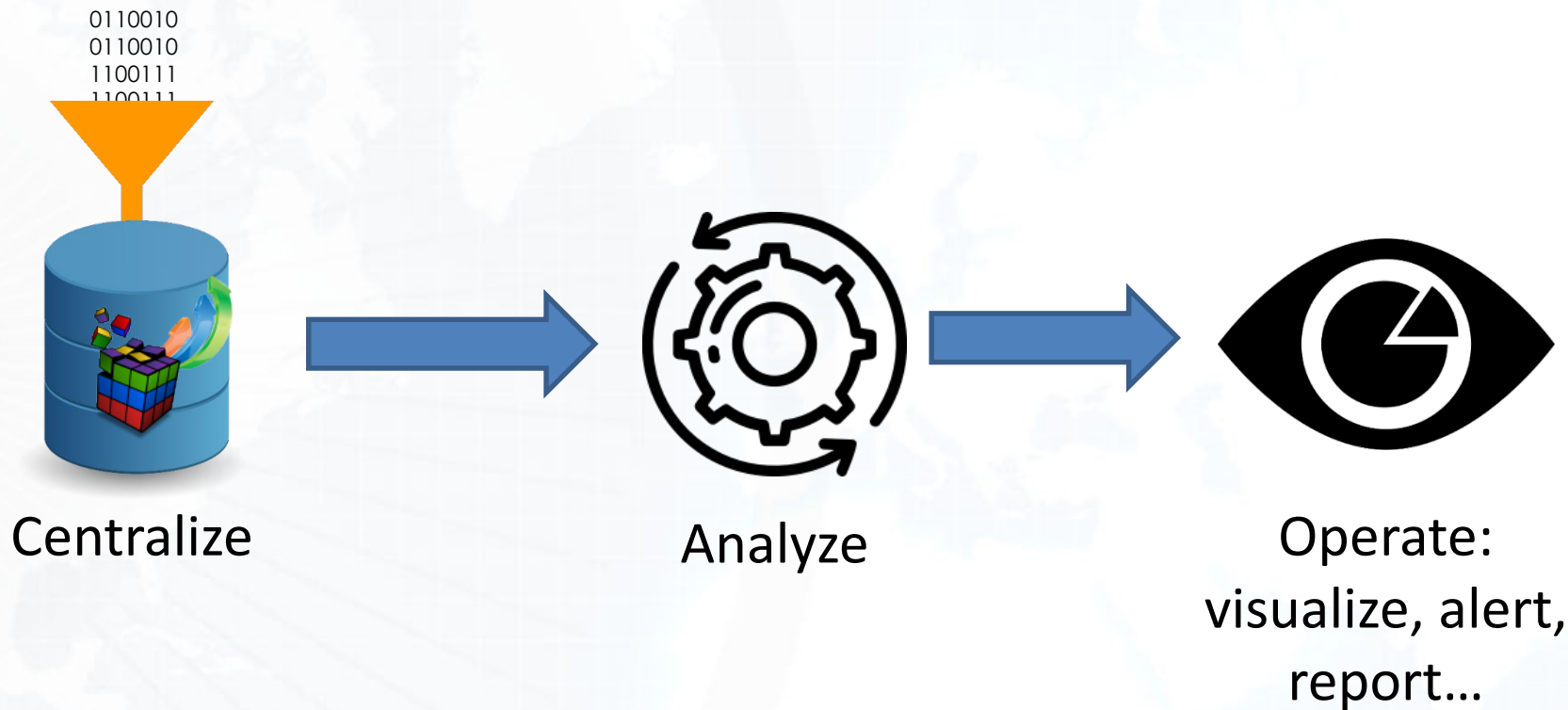
Data Exploitation

4 steps

1. What data should be collected ?
2. How this data can be leveraged ?
3. Collect the data
4. **Leverage the data**

Structuring data & having metrics on the data are keys to success.

Data exploitation platform principle



What is a Data Exploitation Platform ?



A Data Platform

A **data platform** is an integrated technology solution that allows **data** located in database(s) to be governed, accessed, and delivered to users, **data** applications, or other technologies for strategic business purposes.

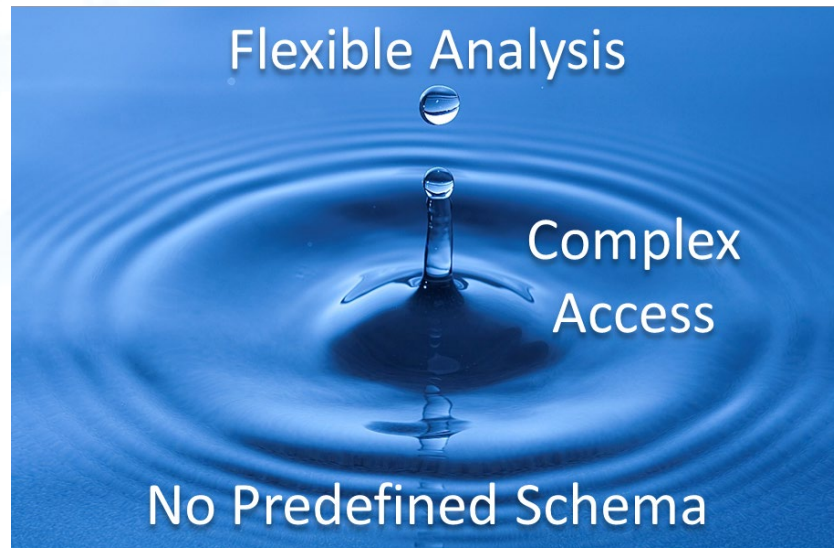


How does it differ from a Data Lake



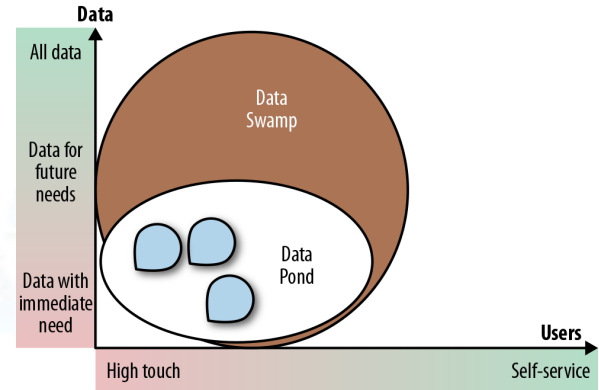
What is a Data Lake?

- A **data lake** is data repository stored in its natural formats
- Includes different kinds of data
 - Structured
 - Semi-structured
 - Unstructured
 - Binary



When Data Lakes are Inefficient

- Required skills and efforts to use the data
 - typically for data scientists
- Exploring the data set is very difficult
- Poor query & analytics performances

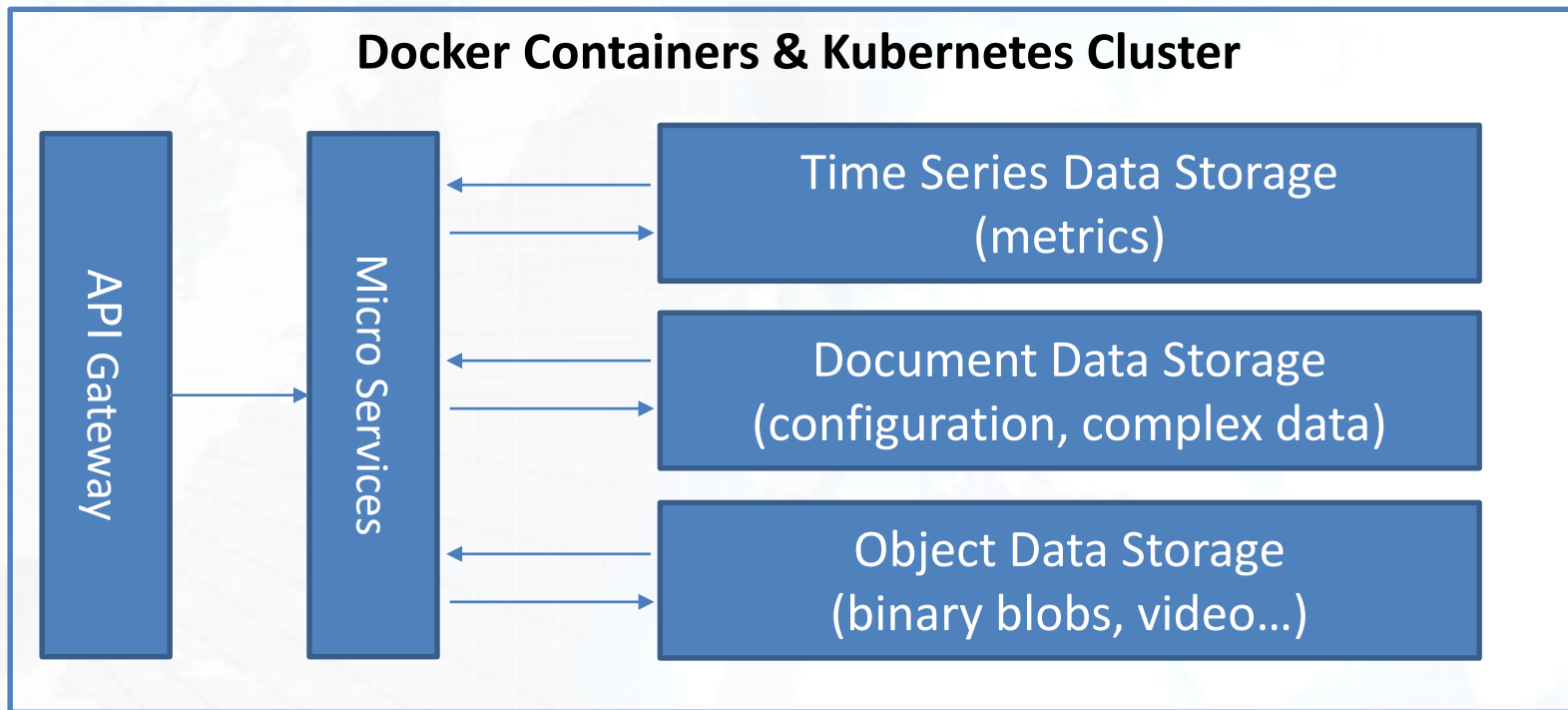


Source: [*The Enterprise Big Data Lake*](#),
Alex Gorelik, O'Reilly

How a Data Platform differs from a Data Lake

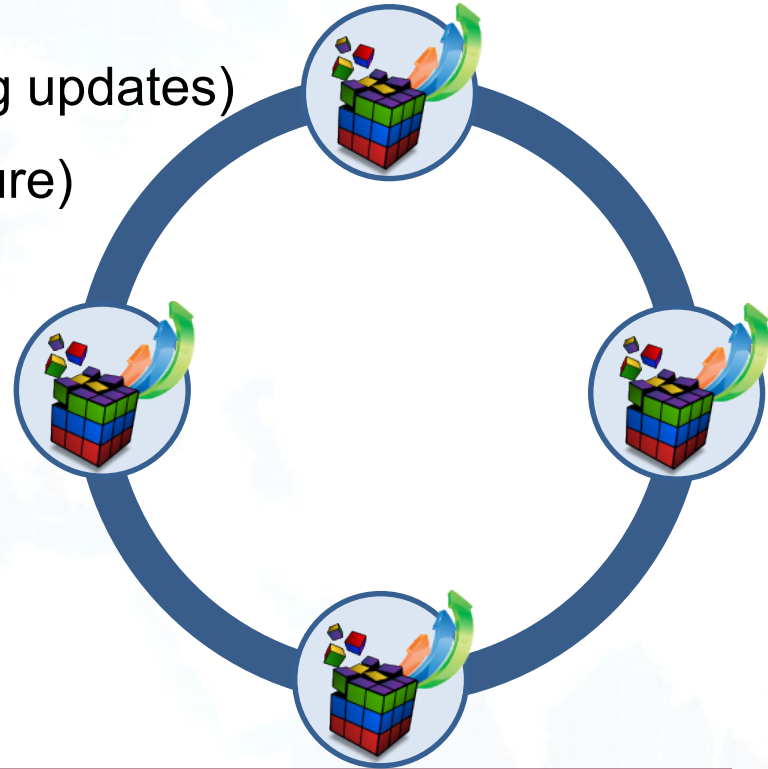
	Data Lake	Data Platform
Data structure	Semi-structured / Unstructured	<u>Simply Structured</u>
Processing	Schema-on-read	<u>Schema-on-write</u>
Storage	Designed for low-cost storage	Designed for low-cost storage
Agility	Highly agile, configure and reconfigure as needed	Highly agile
Security	Maturing	<u>Manageable</u>
Users	Data scientists et. al.	<u>Business professionals</u>
Data retrieval	Software code	<u>REST API</u>

Data Routing Architecture



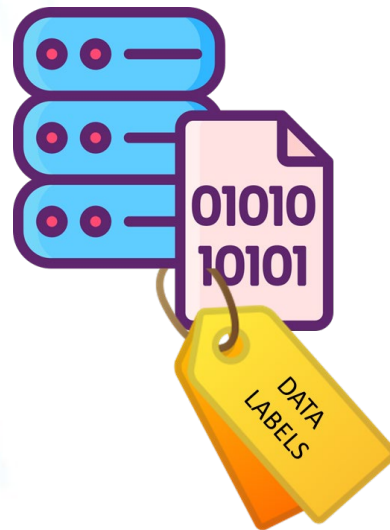
Benefits of the Architecture

- Maintainability (e.g. no downtime rolling updates)
- High-availability (no single point of failure)
- Scalability (can be automated)
- Performances
- High flexibility
- In-control consistency & security
- Simple machine learning pipelines



Data Management

- Data is labelled
- Key-value pairs
 - E.g. satellite, subsystem, antenna, host, domain, site...etc.
- Labels allow
 - Exploring the data set
 - Defining thin access control mechanisms
 - User with role **X** can read all data for *satellite=sat1*, and data with *domain=telemetry* from *subsystem=ADCS* and *fleet=LEO1*...





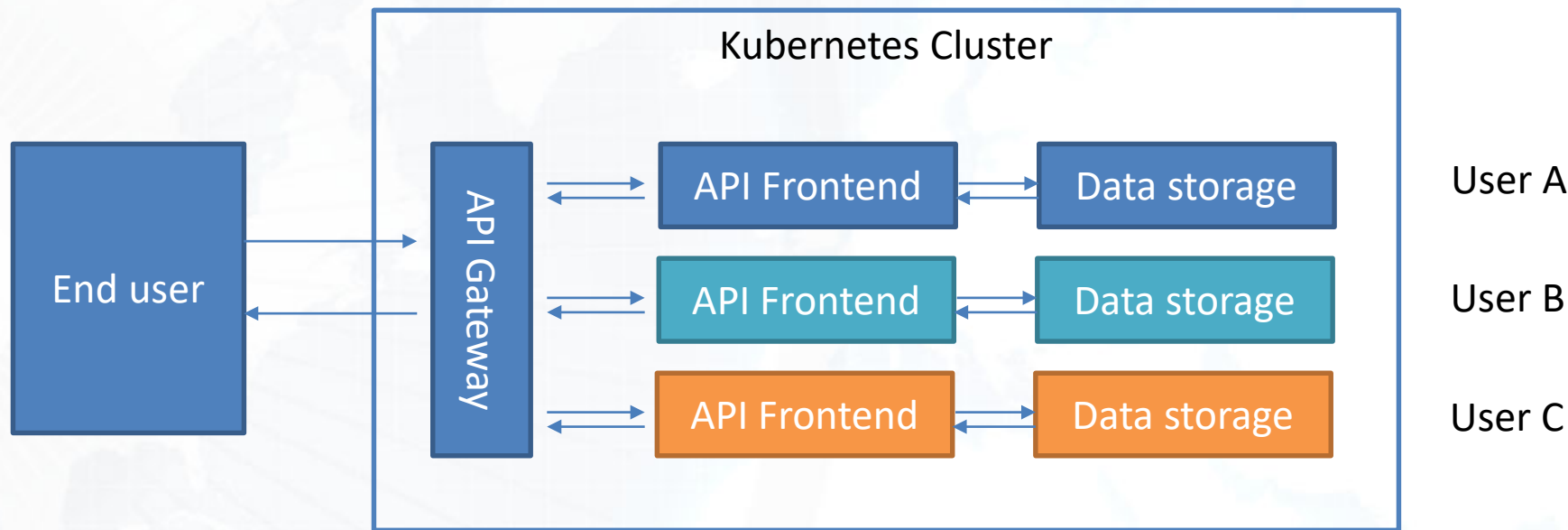
Security & Access Control

Security Architecture

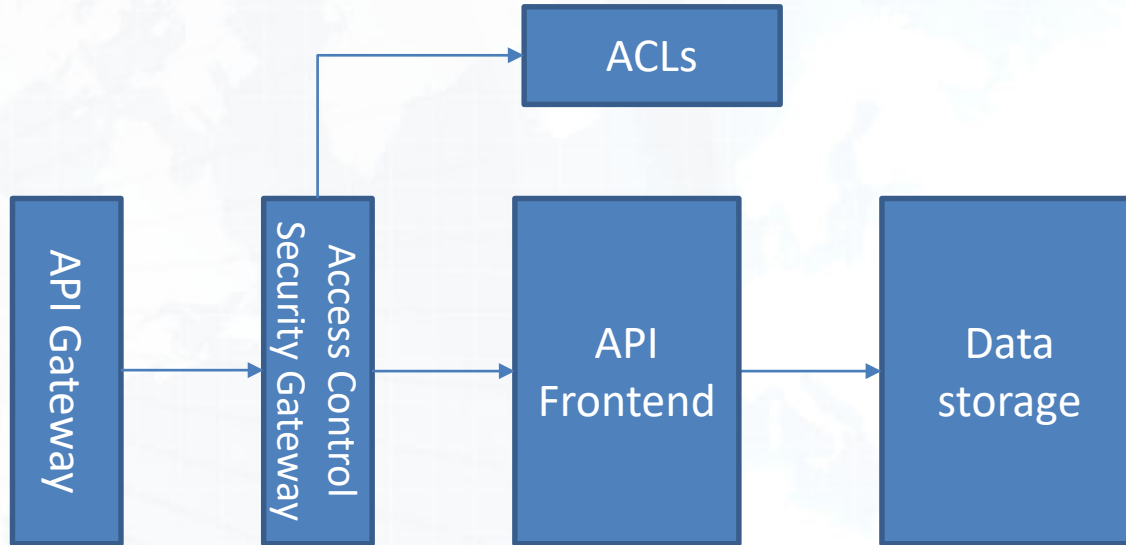
- Single sign-on (OAUTH-2)
- Multi-tenancy
- Ingress access control mechanisms



Security Architecture – Multi Tenancy



Security Role-Based ACL



Openness



Openness

- Open APIs ensure
 - Availability of the data
 - Integration with other software
 - Strong & flexible security



Openness

- Integrates Open Source software for storage and exploitation - with permissive licenses
 - Cassandra / KairosDB for time series
 - ElasticSearch for structured document data
 - Ceph for object storage & file storage
 - Grafana for time series dashboards
 - Python & Jupyter notebooks for analytics
 - Docker containers & Kubernetes for operations



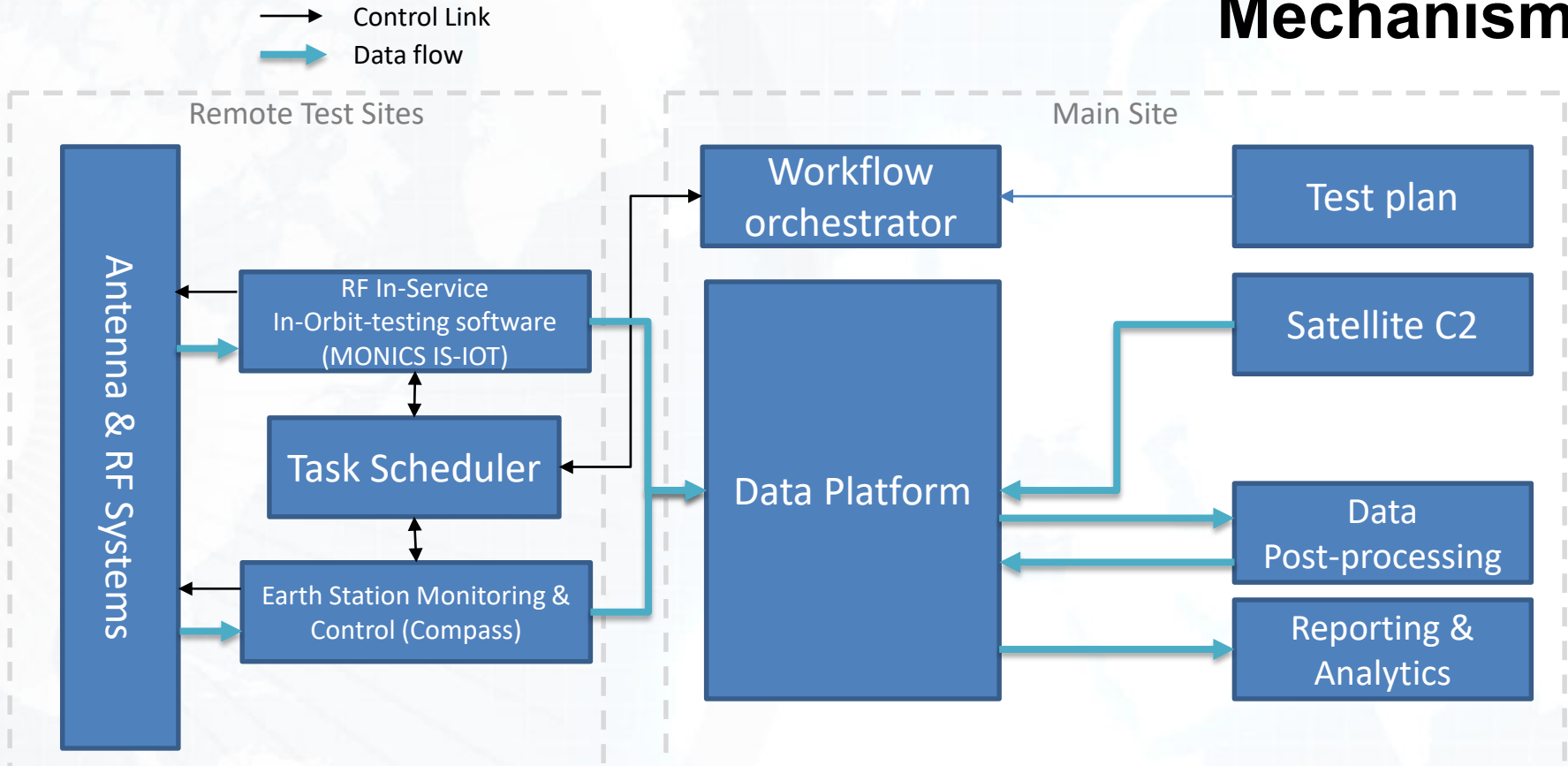
Use Cases...



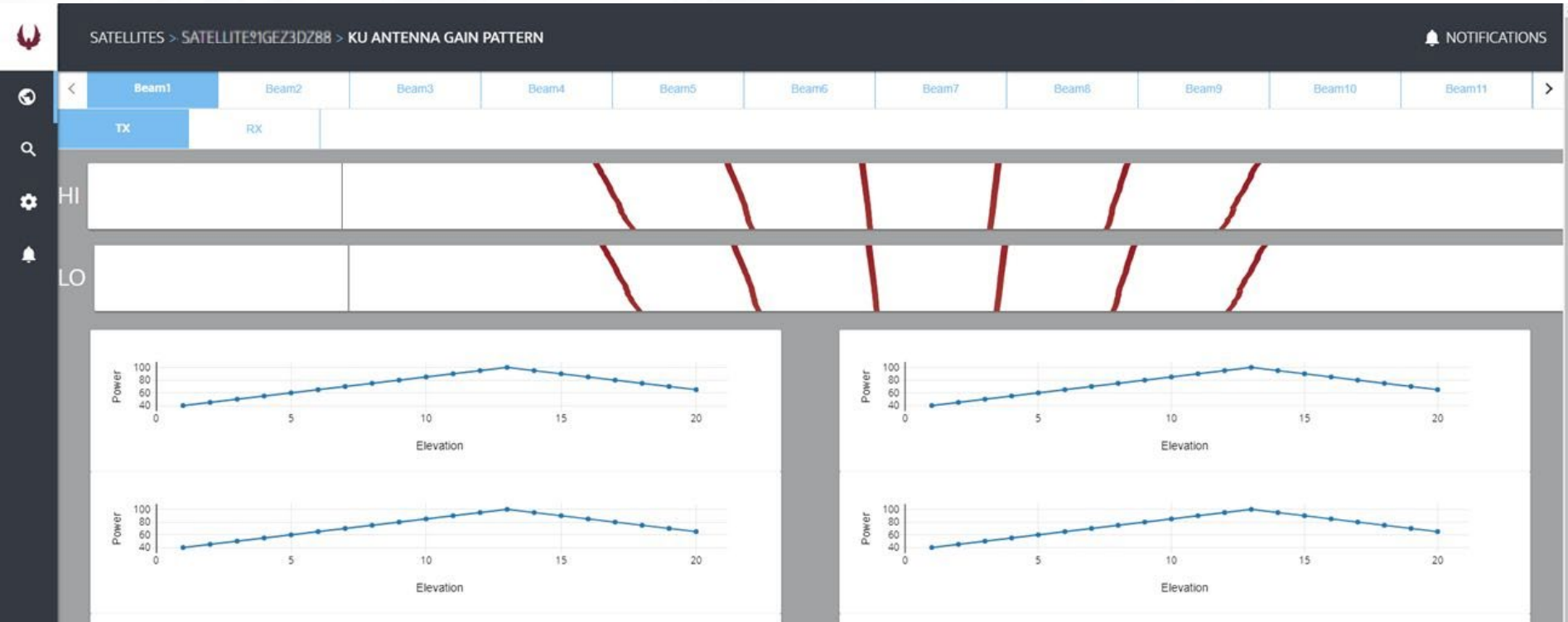
Automated In-orbit Payload Testing of a Satellite Constellation

- Used with Kratos software :
 - Monics IS-IOT (payload testing)
 - Compass (Monitor&Control)
 - A Kratos-provided scheduler and a workflow orchestrator
- Our platform collects, post-processes and presents the data
 - Post-processing algorithms transform the data
 - Bespoke Web-UI for results visualizations

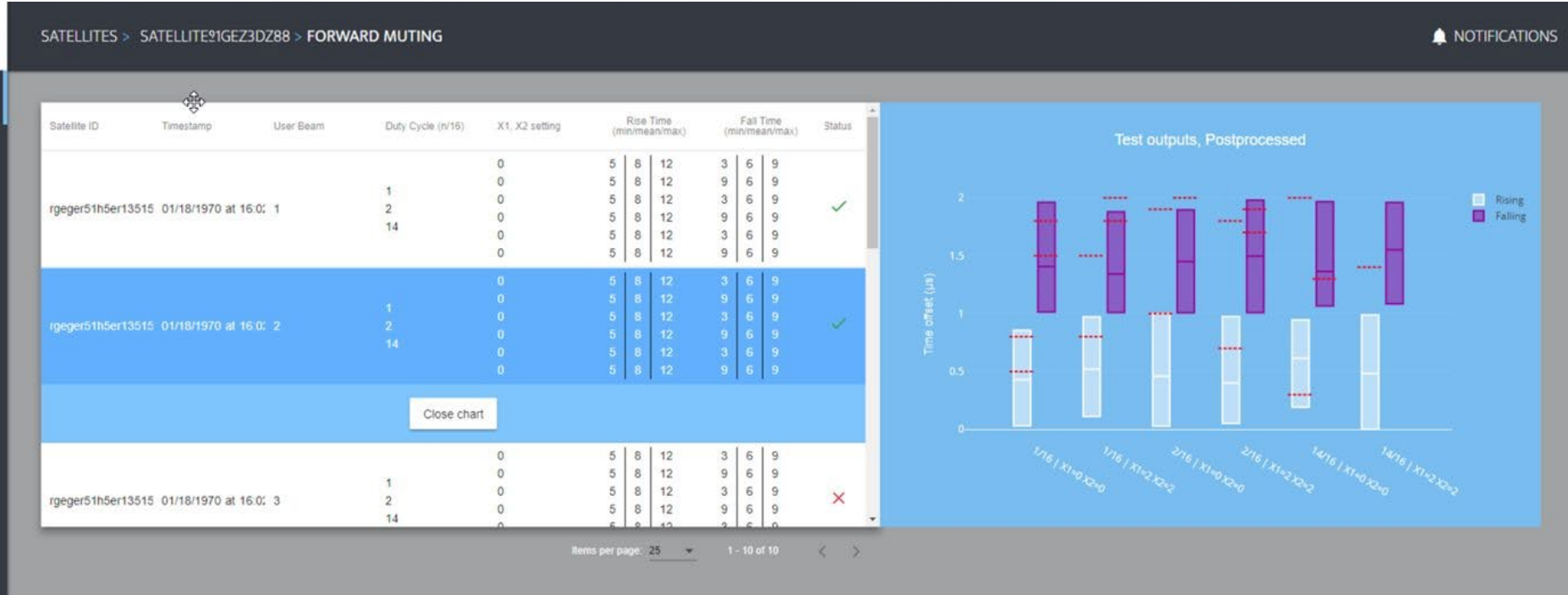
Mechanism



Use Case: Automated In-orbit Testing of a Constellation



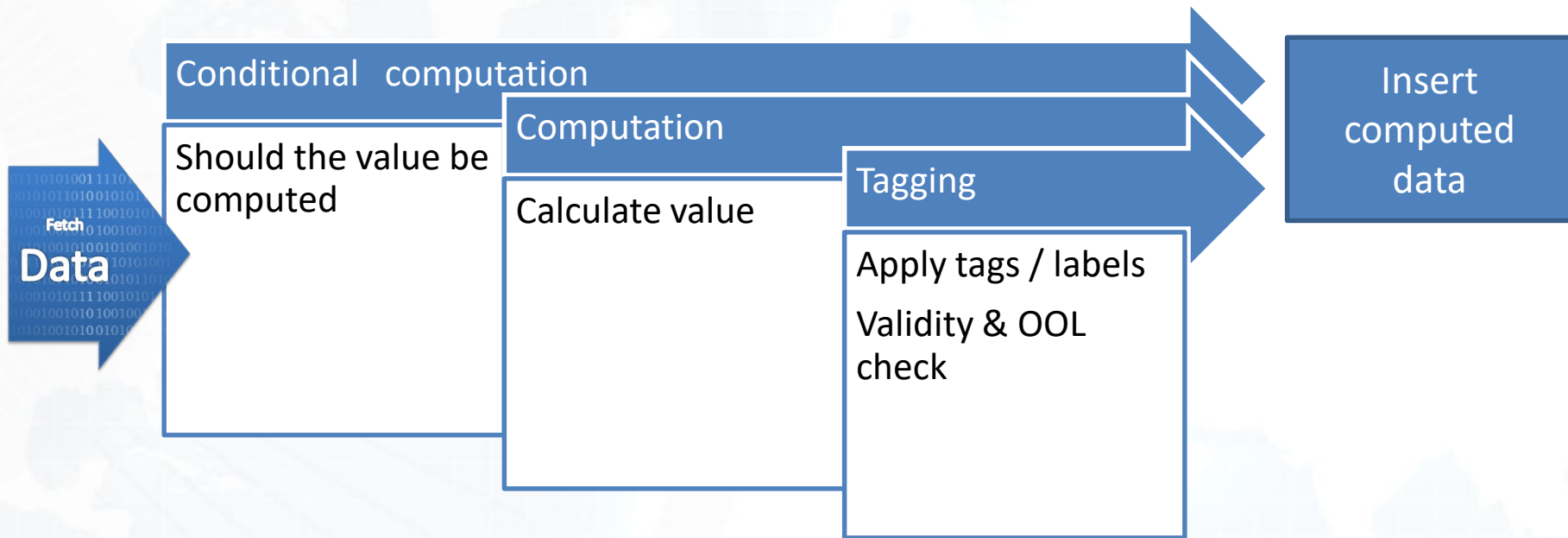
Use Case: Automated In-orbit Testing of a Constellation



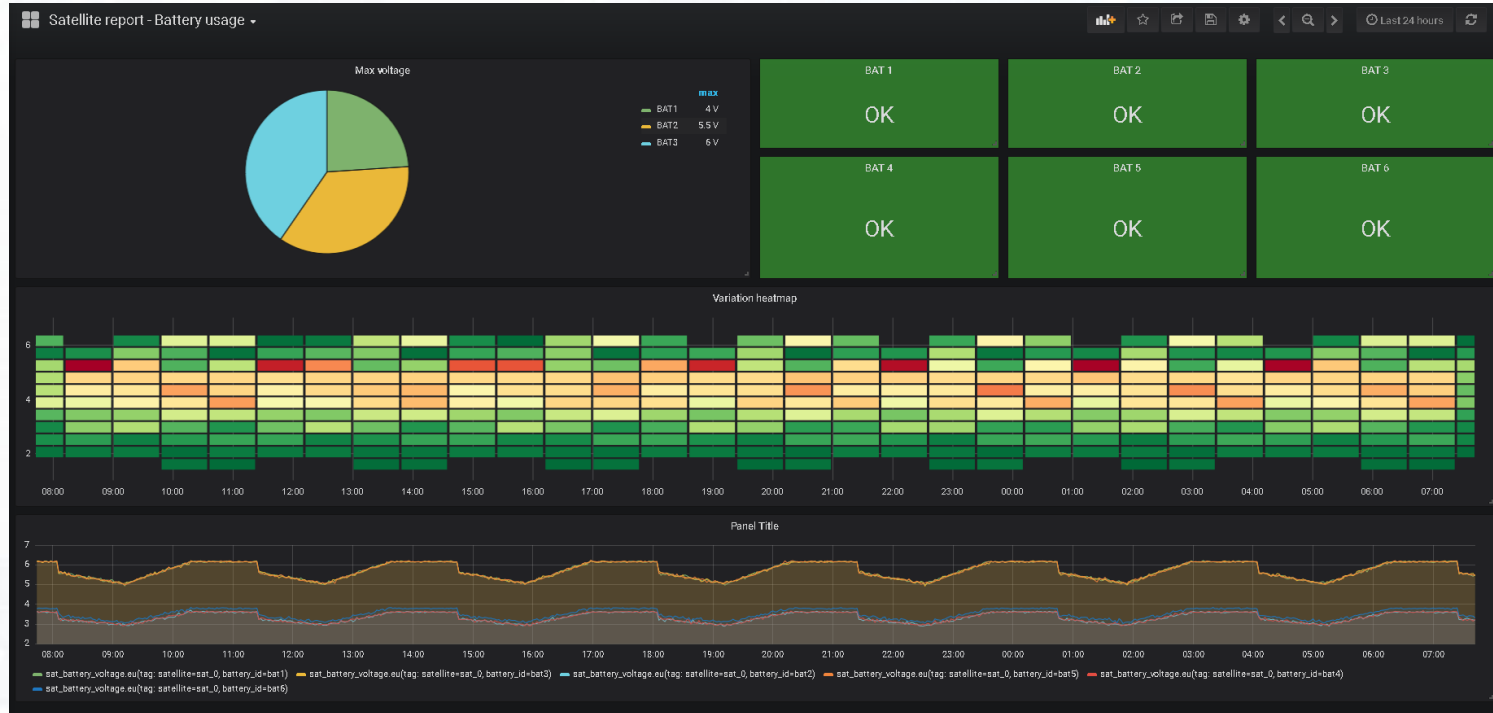
Use Case: Satellite C2 Fleet Data Analysis

- Leveraging value of SCC data
 - 3rd party data ingest using our APIs (C2 not provided by Kratos)
- Faster queries
- Use multiple-satellite in a single query
- Aggregation and correlation between spacecrafts
- Unlimited & re-computable derived parameters from beginning of life
- Automated reporting

Extended Derived Parameters Engine - Pipeline

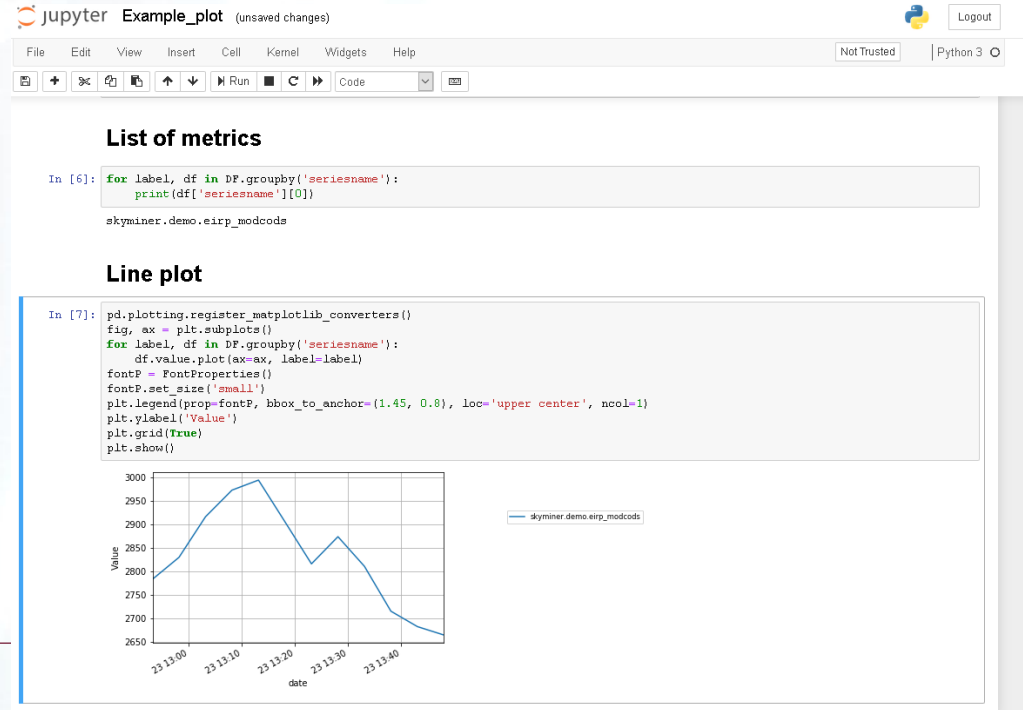
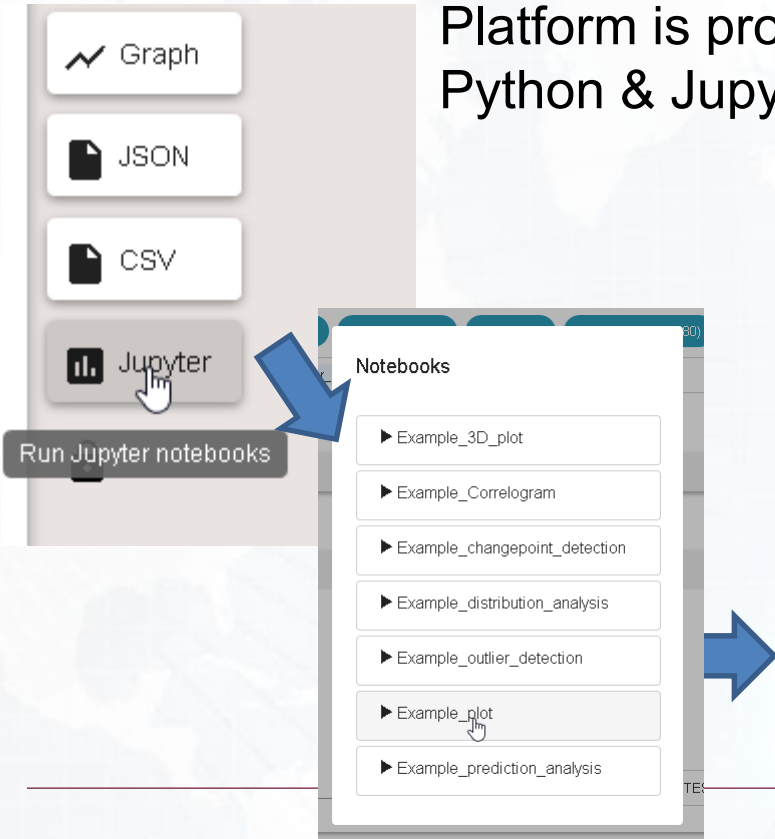


Use Case: Satellite Control Center



Data Science - Python Integration

Platform is provided with analytics integration using Python & Jupyter



Data science – Automated Report Generation

Example_changepoint_detection

January 13, 2020

1 Change point detection

In this notebook the data groups are analyzed for changepoints detection, which means detecting changes in behaviour. Data when processed is considered as a signal and the time dimension is ignored, hence if the sampling rate is inconsistent or if there are gaps in the data, it will be required to pre-process the data before doing this analysis to ensure a consistent sampling rate.

The library used is called **ruptures**. The Changepoint detection is applied using Pelt Model.

cf. the following online resources:

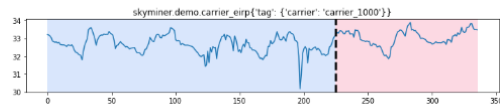
- <https://arxiv.org/abs/1801.00826>
- <http://ctruong.perso.math.cnrs.fr/ruptures-docs/build/html/detection/pelt.html>
- <https://techcrunch.com/2019/08/14/a-brief-introduction-to-change-point-detection-using-python/>

1.1 Import required libraries

1.2 Apply Changepoint detection using Pelt model

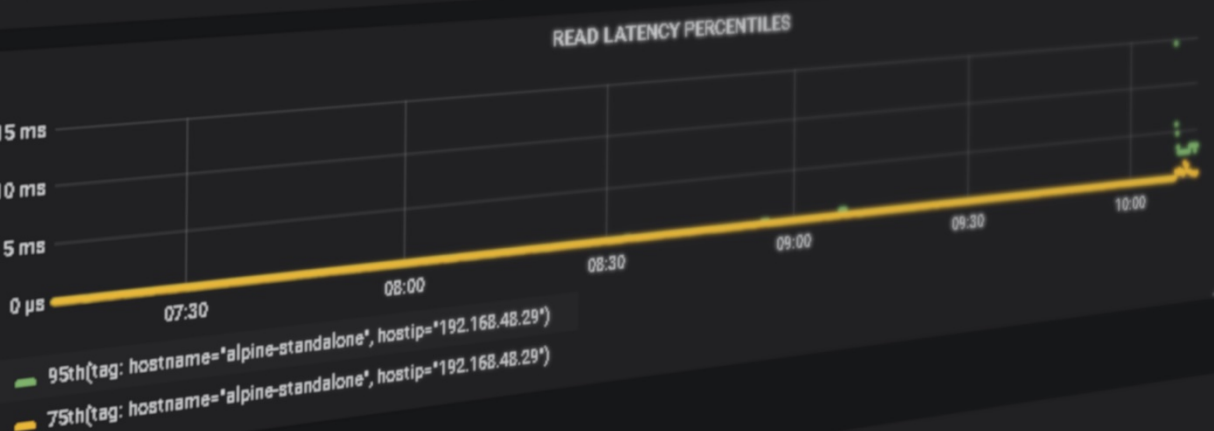
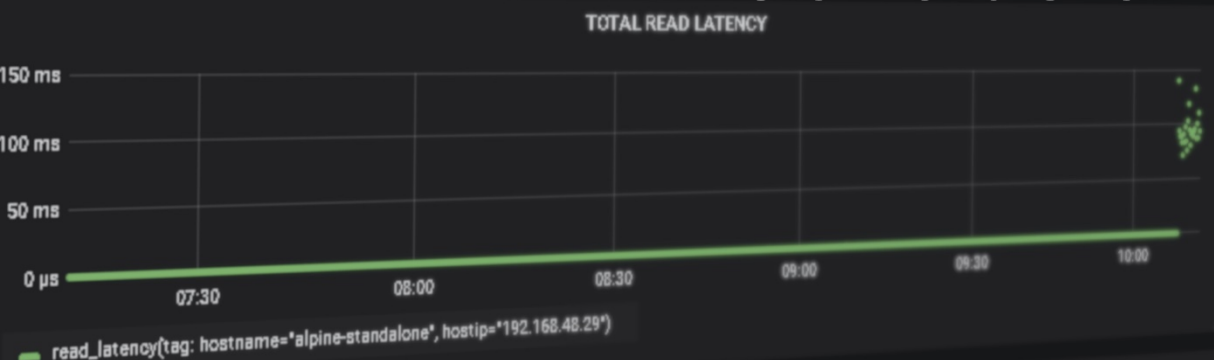
For each of the groups provided by Skyminer, the "signal" is analyzed and then the data is plotted with the change points. There is no time axis since at this stage the time dimension has been ignored (the time series has been analyzed like a signal).

As we know the indice of the change points, we could display data as time series with markers at the specific indices.



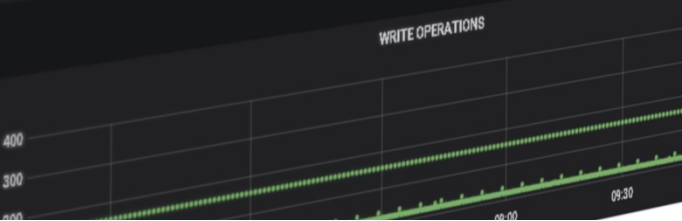
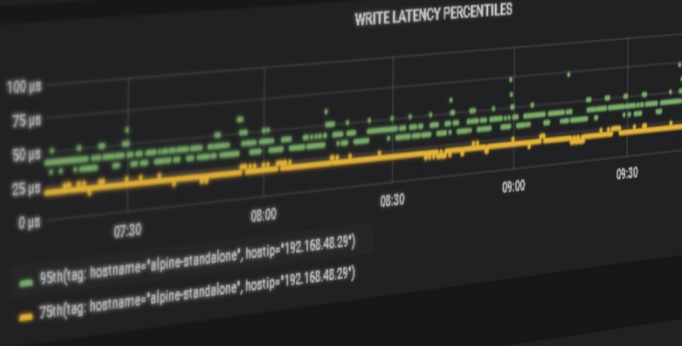
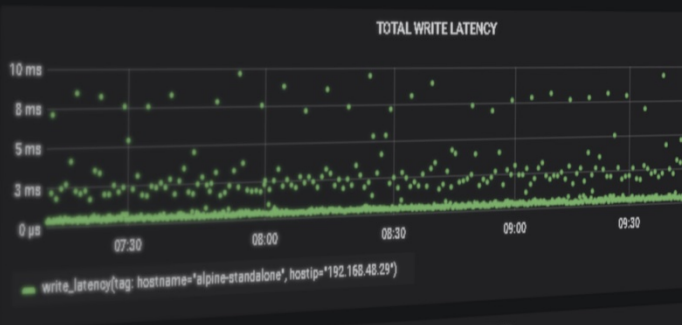
Conclusion

Latency



Operations

READ OPERATIONS



Key Success Factors


- Reliable & proofed in the field
 - Does not require any programming skill
 - Cross-domain data visualization and analytics
 - Users can easily run their own processing
 - Install and connect with compatible products in minutes
 - Integrates quickly with third-party data sources
-
- Built-in high-availability and scalability
 - Unique third-party integrations



Final words

- The platform provides advantages over a data lake
 - Better data accessibility & usability
 - Easier security
 - Smooth machine learning & analytics integration
- Combined with data lake strengths
 - Flexible and modern architecture
 - Built-in High-Availability and Scalability
 - Rely on Open-source products – your data is futureproof





Questions & Answers

References

- <https://www.talend.com/resources/what-is-data-lake/>
- <https://www.unicc.org/in-focus/2019/11/01/the-lake-beneath-the-cloud-icc-works-with-ocha-on-its-first-data-lake/>
- <https://www.collibra.com/blog/data-lake-vs-data-swamp-pushing-the-analogy>,
<https://www.osisoft.com/whitepapers/Data-Lake-or-Data-Swamp.pdf>
- <https://www.morrisopazo.com/diving-deep-into-data-lakes-ebook-aws-partner/>