Big-data for satellite control and testing

Workshop on Simulation and EGSE for Space Programmes (SESP)
26 - 28 March 2019

ESA-ESTEC, Noordwijk, The Netherlands

Irina Alles[(1)], Nicolas Hennion[(1)]

[(1)]Thales Alenia Space, France
5 Allée des Gabians
06150 Cannes

Email:firstname.lastname@thalesaleniaspace.com

INTRODUCTION

Most of the stages of a satellite's lifecycle produce large amounts of data. Each module of a satellite has to undergo an extensive testing phase before it can be integrated onto the main structure. An important part of these tests is the capturing of sensor or telemetry data of the unit under test. The captured data then needs to be screened for anomalies and compared to the expected behaviour. The amount of acquired data during a test case can vary from some Megabytes up to several Gigabytes, for the case of tests of high throughput buses like 1553. A further stage is the "Assembly and Integration" (AIT) phase during which the components are mounted onto the structure and undergo another set of functional and stress tests. A crucial part of this phase is again the monitoring and analysis of sensor and telemetry data.
The test phases are not the only data generating entities. All following stages, from the setup at the launch platform up to the In-Orbit Support phase, produce a huge amount of data. Moreover, the evolution of satellites makes the units more complex and talkative. For example the TM bit rate will be multiplied by 80 in the next 5 years. An appropriate solution for the monitoring and analysis is crucial for the success of a mission.
In order to ease reporting, analysis and monitoring of data for the here described activities, Thales Alenia Space has developed a big data technology based software product called Big Data for SCC/CCS.

BIG DATA CHALLENGES IN THE SPACE INDUSTRY

The classical big data definition introduced by D. Laney in [4] and [5] consists of 3Vs, the high volume, velocity and variety of the information assets. The data produced and collected in the space industry largely fulfills those characteristics. We have seen that the high volume is generated during the different phases of the satellites life-cycle, high velocity needs to be supported for the testing of high throughput sensors or buses, like the 1553. Considering the high variety criteria, a big data solution in the space sector needs not only to support telemetry data but also log and imagery data from sensors and observations. These characteristics require cost-effective, innovative forms of information processing for enhanced insight and decision making, but there are further challenges and constraints to tackle in order to make a big data solution work. One of them is the requirement of using the same or similar testing setup at the white room during the AIT phase and at the launch site. This implies a flexible solution that can run in a minimal setup and scale big if needed. It also adds another V, *variability*, that is often cited in the extended big data definition. In this case it implies the bulk synchronization of the data gathered at the launch site, with the main storage.
A similar constraint is implied by the security requirements of certain white rooms that are not interconnected with the main network and thus require a local big data solution that might synchronize with the main storage periodically.

REFERENCE ARCHITECTURE

Inspired by big data best practices as described in [1],[2],[3],[6] and in order to structure and harmonize the different big data initiatives, a reference architecture has been put in place in order to define the guidelines for a big data solution at Thales Alenia Space.

As shown in Fig. 1. the architecture is divided into four different zones that shall be composed of reusable components, the so called building blocks. The different zones are

- Transit Zone: This is the layer where data gets collected, cleaned, transformed and passed over to the next layer.
- Data Store: This zone is often referred to as serving layer [2], it is where the data gets stored and made accessible for the other parties.
- Processing Zone: This is where the data analysis and machine learning takes place.
- Exploitation Zone: This zone hosts the generic dashboard solutions and custom GUIs.

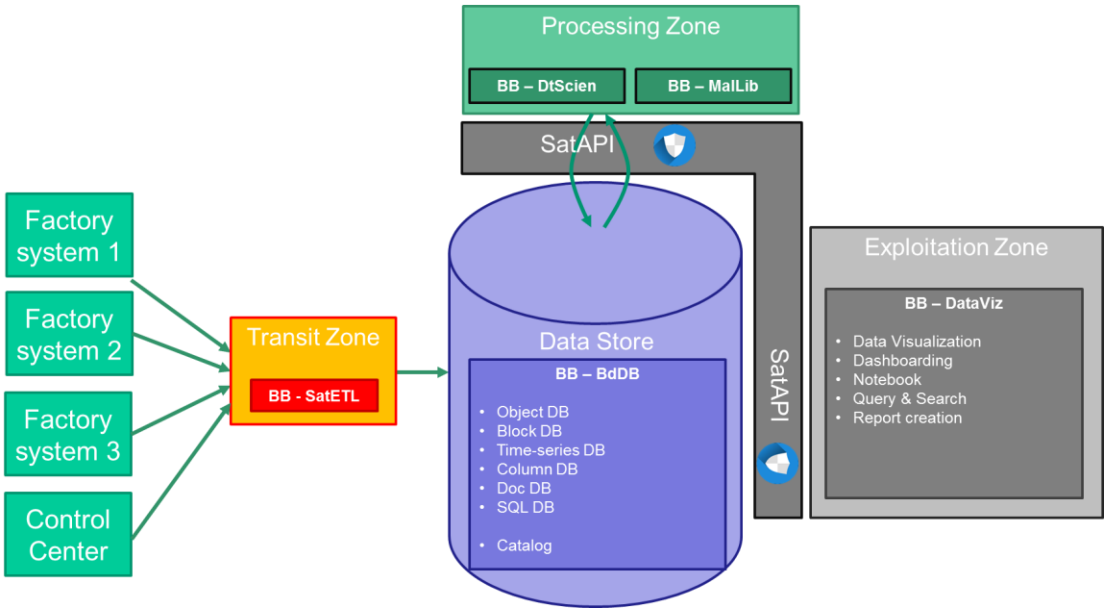

Fig. 1 Reference architecture overview

BIG DATA FOR SCC/ CCS

The Big Data For SCC/CCS product is about applying the reference architecture to a specific domain, that of the Central Checkout System (CCS) and the Satellite Control Center (SCC). The goal is to design and implement a solution that satisfies the needs of the given domain while keeping the components as generic and modular as possible in order to produce reusable building blocks.
In order to take into account a large spectrum of end-user needs during the design process, a user requirement analysis involving the end-users into the process has been conducted following the first steps of the user centered design standard defined by ISO 9241-210 [7].
This study confirmed and structured many expected requirements, such as the support of very variable throughput speeds; besides telemetry and log data, the support of imagery data, the need for correlation between different data types and the need for a flexible solution that can work on a minimal infrastructure setup, just to name a few.

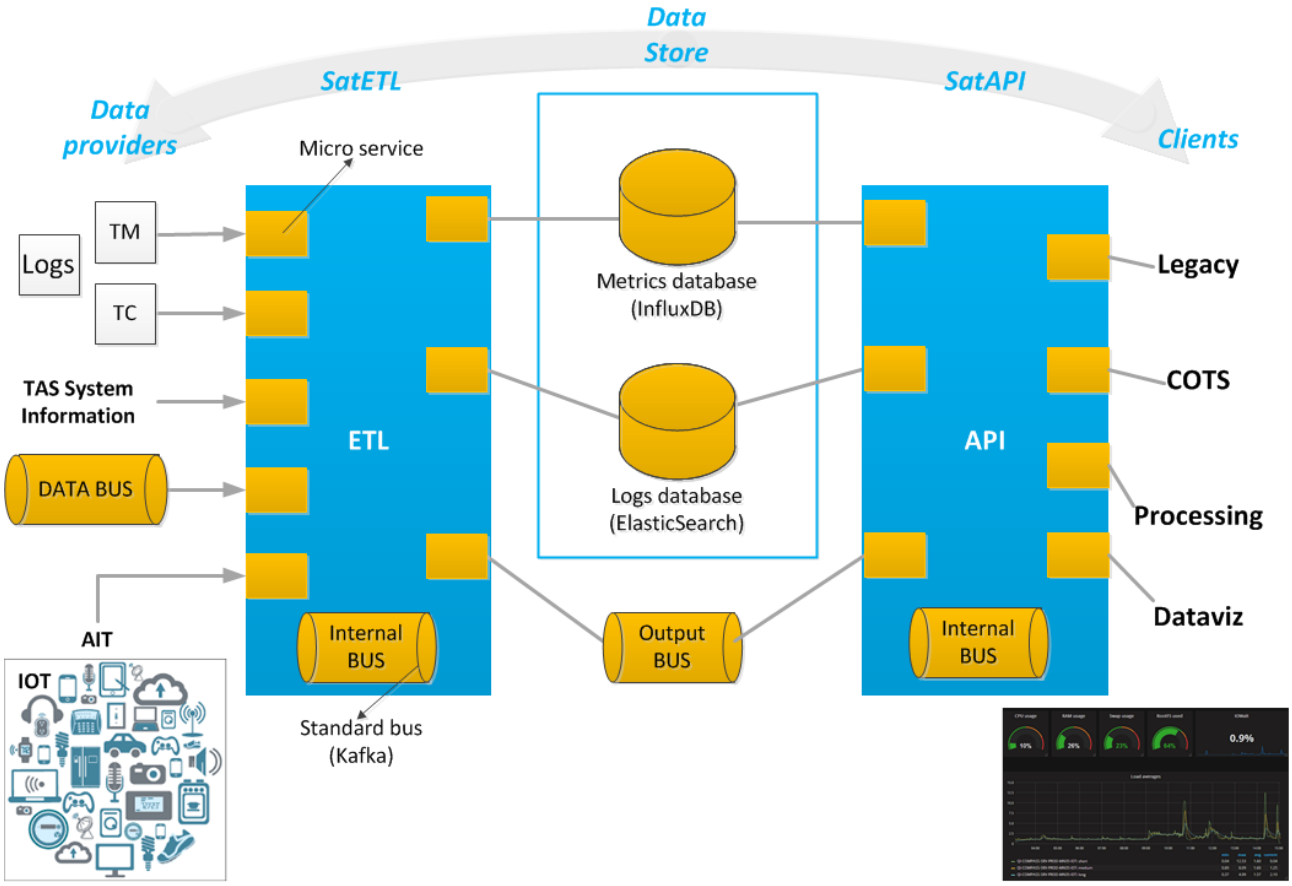

Fig. 2 Big Data for SCC/CCS architecture overview

As already implied by the reference architecture, the retained solution is based on a simplified Lambda architecture [2] without the batch layer, it is often referred to as Kappa architecture [3].
It consists of the following two main layers:

- Speed layer or real-time layer: incoming data is ingested via a messaging system or processed by a stream processing framework.
- Serving layer: The data processed via the speed layer is sent to the serving layer, which is a storage system e.g. a NoSQL database.

In our case this architecture results in three building blocks as depicted by Fig. 2

- SatETL for the speed layer,
- Data Store as the serving layer and
- SatAPI providing a unified data access.

Details for the individual components are provided in the following.

SatETL

SatETL is a data collector, it aims to Extract, Transform and Load data. Based on an open and plugin-ready architecture, it allows to extract data from heterogeneous data sources, like databases, application buses, CSV/XML files, application and system log messages. It allows to apply transformations on the given data in order to obtain a proper format and finally load the data into the target destination (like databases, files, application buses). In case of a high data flow frequency, SatETL acts as a buffer, avoiding data loss whenever the target system gets busy. The ingestion workflows can be executed in parallel and are suggested to be run in distributed mode in order to support high availability requirements.

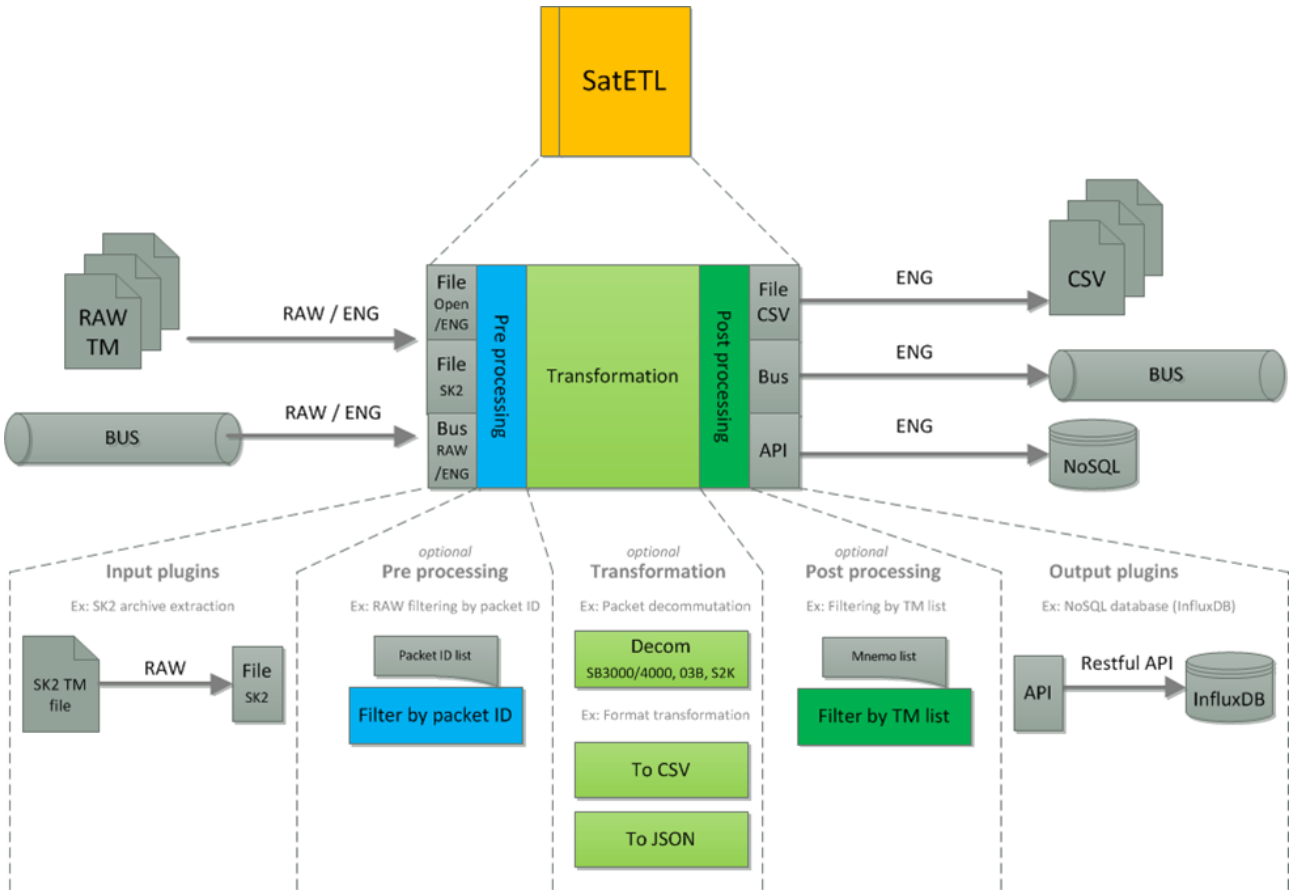

Fig. 3 SatETL applied to telemetry data

Fig. 3 shows the working principle of SatETL on the example of telemetry acquisition. The data source can be a control center or a test bench. The connector receives the data and puts it through a first processing step, where it can be cleansed, filtered, restructured and then, if needed, transformed. Once the data is transformed we can apply another processing step to for example filter on information revealed through the transformation process or enrich the data by calculation or the aid of another data source. These steps form a data ingestion pipeline, they are interchangeable and can be omitted or repeated as often as required before sending the data to the data store.
The underlying technology is the messaging system Kafka. It is broadly used in the industry [8] and satisfies the most important requirements such as high throughput and support for data bursts without data loss, which can arrive during busy testing phases. It assures the modularity and reusability of the solution as also the transformation and processing steps, called Consumers and Producers in the Kafka jargon. They can be reused across pipelines and pipelines can be interconnected to produce complex workflows.
As additional ingestion component we use Logstash in order to facilitate the parsing and ingestion of data present in textual files such as csv, xml etc. Logstash allows to define the parsing of a file and restructuring of the data via a configuration file.

Data Store

The data store layer applies the approach of polyglot persistence, where different technologies are used to fulfill the varying storage needs. The solution includes a time series database used for telemetry and sensor data. It optimizes storage and query efficiency for time series inherent access patterns and processing requirements. A document store is used for the more verbose log-alike and event data types.
Considering the nature of the data collected from the SCC and CCS we have a large amount of time-stamped data, such as telemetry and sensor data, which is accessed and processed in chronological order. This explains the choice for a time-series database, optimized for this exact use case, in this case InfluxDB. During local tests InfluxDB has shown a remarkable data compression performance (30 000 000 TMs result in 270MB for InfluxDB vs. 500 Mb for Cassandra)

and supports to be run in a cluster as also in a single node setup. This satisfies the need for a fast serving layer and the requirement of a possible minimalistic setup. Of course it needs to be considered that in a minimalistic setup where we deal with a single storage node, high availability (HA) needs to be assured otherwise.
The other part of the collected data is of log-like nature. This kind of data is mostly accessed via key words, thus the choice of the most popular enterprise search engine, ElasticSearch [9]. Hence ElasticSearch supports real-time GET requests on keywords and also on timestamps, it not only serves as a search engine, but is suitable as a NoSQL store.
Despite the current focus on time-series and log data, the data store is easily extendible to accommodate a storage solution optimized for image data.

SatAPI

SatAPI is a micro service oriented Restful API layer written in Go. Besides its main purpose of providing a standard interface to target the data store, it removes the complexity of dealing with the specificities of the underlying data stores at the application layer. Thus the access to multiple storage solutions of the data store, in order to generate a response to a query will be transparent for the caller. Further the SatAPI layer allows to decouple the custom visualization and processing solution from the data store layer. A change in the storage solution will therefore only impact the implementation of the specific SatAPI service.

Visualisation

The Big Data for SCC/CCS product comes with two generic dashboard solutions, Grafana and Kibana. They allow to create dynamic dashboards composed of various types of panels, such as graphs, tables, radar charts as shown in Fig. 4.
These dashboards allow the end-user to get a quick overview of its data as also to configure alerting on thresholds or forecasts.

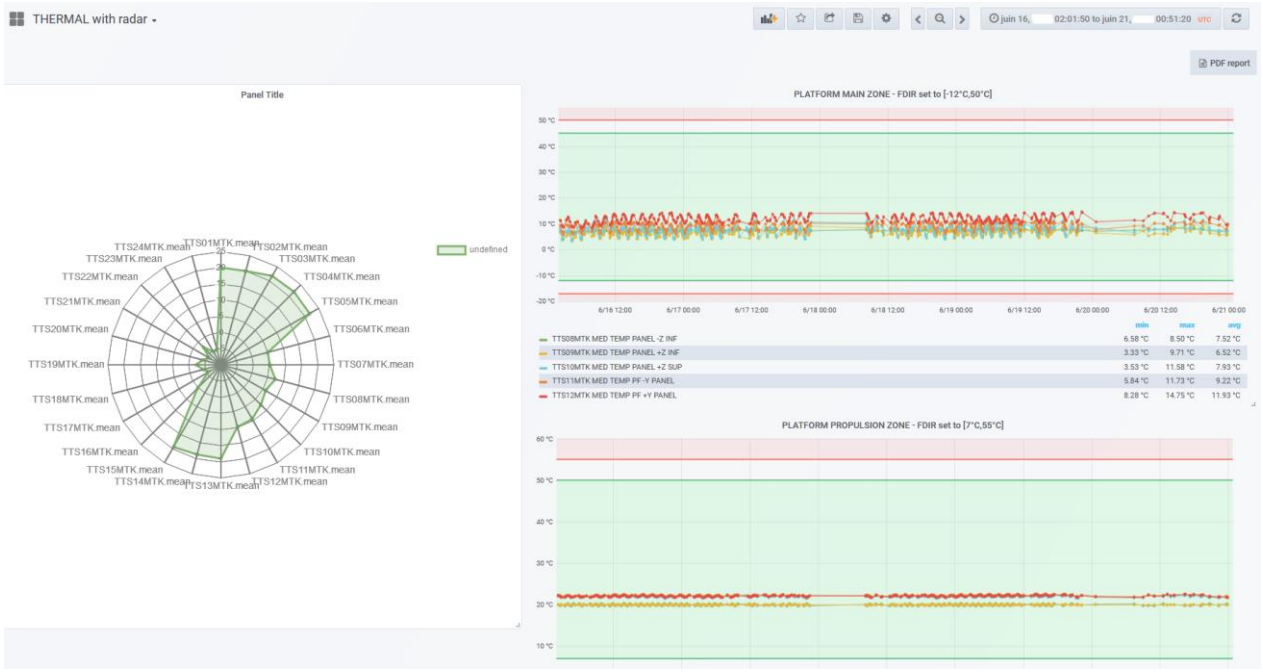

Fig. 4 Example of a satellite health monitoring dashboard using Grafana

CONCLUSION AND FUTURE WORK

The product Big Data for SCC/CCS is integrated into the current version of Thales Alenia Space's control center solution OPEN SCC as also the CCS. The solution is also provided in standalone mode to external clients to allow near real-time satellite health and performance monitoring and the generation of reports.

Furthermore, the generic building block approach shows its first results. SatETL has been successfully extended to incorporate a connector for EGSE for a Minimal Viable Product (MVP) providing near real time test bench health monitoring.
The next step is the addition of the processing layer which consists in consuming data from the data store and using it for long term analytics, fault predictive or deep learning algorithms. Moreover, the automatic analysis of archived data is identified as a strong opportunity for early fault detection as also their anticipation, mainly in the case of in-flight constellations (e.g. Globalstar, O3b, Iridium Next) where all satellites are strictly identical, but also during the AIT phases.
Current efforts focus also on the integration of the here presented building blocks into Thales Alenia Space's global data lake. This step opens the door for data mining and analytics on data coming from completely different sources and provides the traceability of a satellite's components during their whole lifetime.

REFERENCES

[1] A. Verma "Real-time Big Data Pipeline with Hadoop, Spark & Kafka", *https://www.whizlabs.com/blog/real-time-big-data-pipeline/*, August 29, 2018
[2] N. Marz, W. James, "Big Data: Principles and best practices of scalable realtime data systems.", *Manning Publications*, 2013
[3] J. Kreps "Questioning the Lambda Architecture", *https://www.oreilly.com/ideas/questioning-the-lambda-architecture*, July 2, 2014
[4] D. Laney, "3D data management: Controlling data volume, velocity and variety.", *META group research note*, vol 6, p1, February 2001
[5] M.A. Beyer, D Laney, "The Importance of "Big Data": A Definition.", *Gartner Publications*, pp.1–9, 2012
[6] W.L. Chang, "NIST big data interoperability framework: Volume 6, reference architecture." *No. Special Publication (NIST SP)-1500-6r1*. June 2018.
[7] ISO 9241-210, "Ergonomics of human-system interaction. Part 210: Human-centred design process for interactive systems*.", ISO*, 2008
[8] Confluent, "Survey of Apache Kafka Users Highlights Impact of Streaming Data on Global Businesses", https://www.confluent.io/press-release/survey-apache-kafka-users-highlights-impact-streaming-data-global-businesses/, Press Release, 2017
[9] DB-Engines, "DB-Engines Ranking of Search Engines", *https://db-engines.com/en/ranking/search+engine*, accessed February 2019