DevOps: the perfect ally for Science Operations for a large and distributed astronomy project.

Rocio Guerra,[1] Neil Cheek,[2] Eduardo Anglada,[3] Pilar Esquej,[4] Emilio Fraile,[4] Enrique Pozo[5] and Uwe Lammers[6]
[1]*European Space Astronomy Centre, ESA, Madrid, Spain;*
rguerra@sciops.esa.int

[2]*European Space Astronomy Centre, ESA, Madrid, Spain*

[2]*Serco Gestion de Negocios S.L. for ESA, ESAC, Madrid, Spain*

[3]*ATG Europe for ESA, ESAC, Madrid, Spain*

[4]*RHEA for ESA, ESAC, Madrid, Spain*

[5]*Aurora Technology B.V. for ESA, ESAC, Madrid, Spain*

[6]*European Space Astronomy Centre, ESA, Madrid, Spain*

Abstract.
The Gaia Science Operations Centre (SOC) is an integral part of a large consortium responsible for Gaia data processing. Serving terabytes of processed data on a daily basis to other Processing Centres across Europe makes unique demands on the processes, procedures, as well as the team itself. This paper describes how we have embraced the DevOps principles to achieve our goals on performance, reliability and teamwork.

1. Introduction

Gaia was launched on December 19th 2013 with an astonishing objective: make the largest, most precise three-dimensional map of the Milky Way in order to reveal its content, dynamics, current state and formation history. The second data release (Gaia DR2) delivered on April 25th 2018 is the most recent proof of the success of the mission and the revolution this largest catalogue ever made is (already) bringing to all the Astronomy fields.
The Data Processing and Analysis Consortium (DPAC) was entitled in 2006 to carry out the processing of Gaia's data with the final objective of producing the Gaia Catalogue. That is an extremely complex responsibility that encompasses many different tasks: developing the data processing algorithms and corresponding software, providing the IT infrastructure and executing those during the mission. The processing is end-to-end as it starts with the raw data received from Gaia and culminates with the generation of the final scientific data products that will be released to the scientific community.
The DPAC consortium is a large team of more than 160 institutes and ESA involving around 450 scientists and experts in the fields of software development and engineer-

ing. DPAC is organized in specialized units (for astrometry, photometry, variability, etc.) known as Coordination Units (CUs) in charge of the design of the scientific algorithms and their software implementation and the Data Processing Centres (DPCs) where those processing software systems are integrated and executed within a proper IT infrastructure. Fig. 1 shows the DPAC organigram depicting the Coordination Units

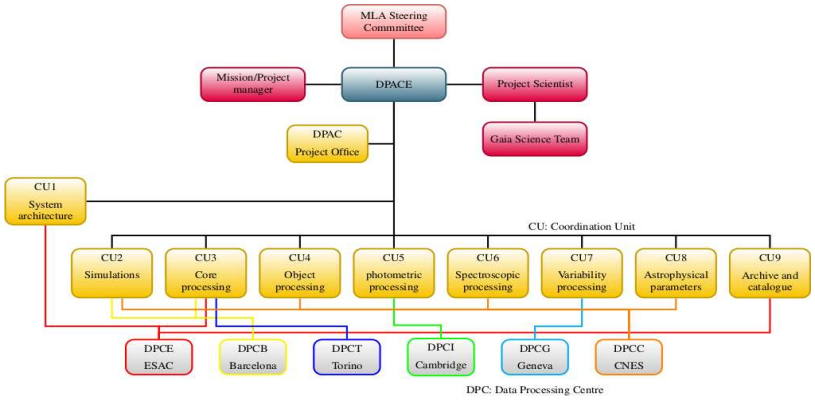

Figure 1. DPAC organigram

and Data Processing Centres (plus the DPAC management bodies at the top). Coloured lines connect each CU with the DPC that runs its software. The Gaia Science Operations Centre is an integral part of the DPAC consortium. It does not only perform the roles generally assigned to a "classical" ESAC SOC, mainly:

- be the Mission Operations Centre (MOC) primary contact for all the payload and science-related mission aspects,
- produce the mission planning products for MOC (Scan Law, Science Schedule, etc.),
- perform regular payload health monitoring,
- processing first-level products and disseminate them downstream and
- archive development and operations.

but also, under the context of DPAC, it holds a Data Processing Centre (the so-called DPCE) and is home to Coordination Unit teams: CU1 (defining system architecture and developing and running the DPAC central repository - Main Database), CU3 (producing daily intermediate products and developing and operating the astrometric core system - AGIS) and CU9 (supporting the catalogue creation). Fig. 2 highlights (in red circles) the DPAC activities performed at ESA. Excluding the operations of the archive, the SOC data processing is divided in cyclic and daily.
The cyclic processing (AGIS, the Astrometric Global Iterative Solution) achieves the ultimate accuracy through iterations, therefore it is run several times during the mission incorporating new improved results from other DPAC systems.
On the other hand, there is a processing pipeline continuously up and running as per a data driven approach, i.e. any time MOC pushes new acquired science telemetry from Gaia (˜35GB/day up to 70GB) it enters into the pipeline that unpackages and

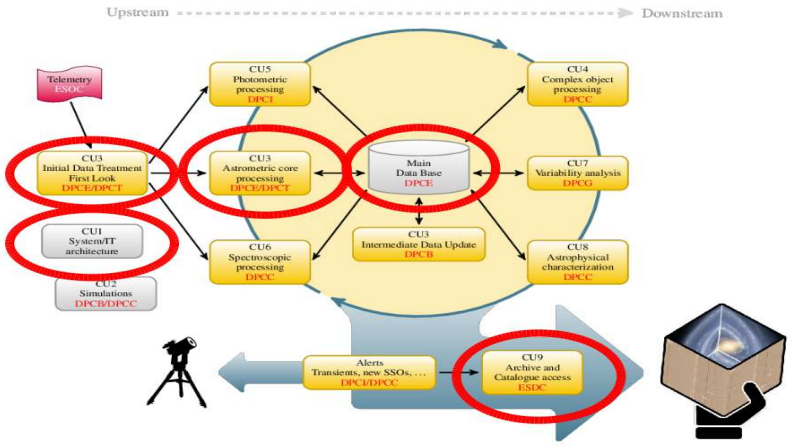

Figure 2. ESA contribution to DPAC

decompresses the raw inputs, determines basic image parameters, cross matches the observations taken during the day with entries in the catalogue and monitors the health of the payload. The main results (300-500GB/day) are stored in the central repository - the Main Database - and distributed to the other Data Processing Centres (Fig. 3). This

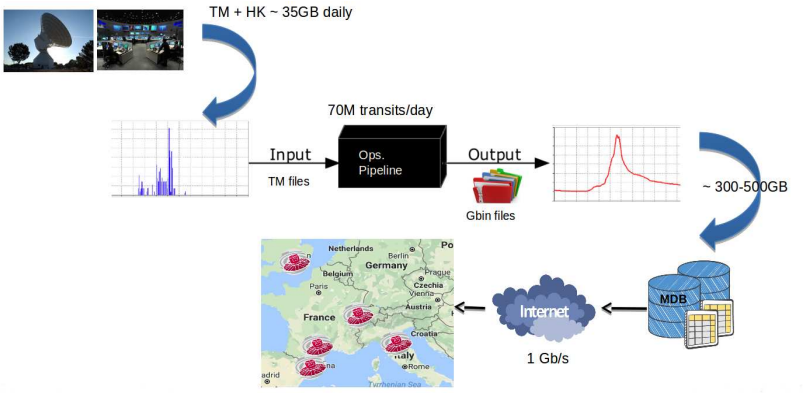

Figure 3. SOC Daily Pipeline

paper describes the challenges that this pipeline imposes to the Operations team and the solutions adopted in order to achieve a smooth sustained processing.

2. Challenges

The daily pipeline running at the Gaia SOC is very complex as it was designed to fulfil very strict requirements especially in terms of performance. The reason is twofold. Firstly, DPAC produces science alerts (photometry and solar system alerts) in other Data Processing Centres that need of inputs from the SOC as soon as possible to be traceable. Secondly, the payload health monitoring requires very often meticulous checks and therefore it is also necessary to have the first-level products available

promptly.
Both needs requires the processing is done in near real-time. Because the telemetry does not arrive time-ordered (it follows a priority downlink scheme - plus there may be issues that mess up the incoming data) the integrated systems were designed to be very tightly coupled with many dependencies among them in order to produce small time ordered chunks of data to be distributed downstream.
On the other hand, the volume of data to handle is massive. The average size of telemetry (science and housekeeping) received from MOC daily is 35GB (70TB in total since launch). And for denser areas scanned by Gaia (like the Galactic plane) it increases up to 80-90GB. The outputs stored in the Main Database are about 300-500GB/day (and most of them are disseminated to the rest of the DPAC DPCs). The average size of the working database (IDTFLDB) is 30TB. It is so large because it has to keep data from a long period of time. Fig. 4 is a more realistic representation of the previous Fig. 3. The

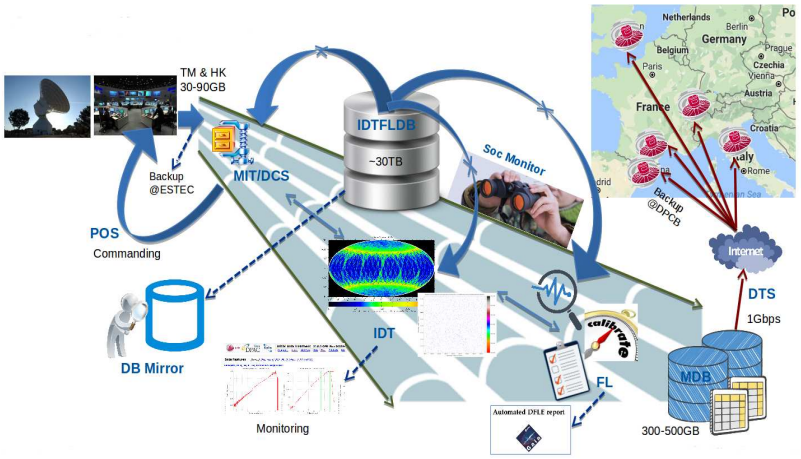

Figure 4. Complex SOC Daily Pipeline

names of the systems are not relevant. It intends to show the amount of dependencies and tasks involved: there is not a sequential flow but many tasks tangled.

3. The search for Robustness

Well before the Gaia launch the SOC team started the integration and validation of the software systems within the IT infrastructure. The focus was set on achieving a robust pipeline (highly reliable, scalable and available) capable of coping with errors and protecting the processing from all kind of unexpected malicious event.
In order to achieve that goal a large suite of validation tests were conducted (system, integration, end-to-end tests) including Operational Rehearsals.
A dedicated DPAC Coordination Unit was in charge of providing simulated data at different levels (crucial to get realistic tests and meaningful outcomes) and through formal Test Readiness Reviews and Test Review Boards the quality of the test campaigns were guaranteed and the outputs traced.
In parallel additional software engineering and preparatory operational activities were done e.g. organizing Configuration Control Boards, defining workflows and interfaces,

identifying stakeholders, creating detailed procedures, etc. resulting in a robust integrated pipeline.
However,the hectic operations needed to keep up with the intense processing and the unlimited scenarios surfacing bugs, unexpected loops and hidden dependencies demonstrated that robustness, although certainly positive, is not enough for the requirements imposed for new missions like Gaia. Fig. 5 is a processing flow diagram. While the

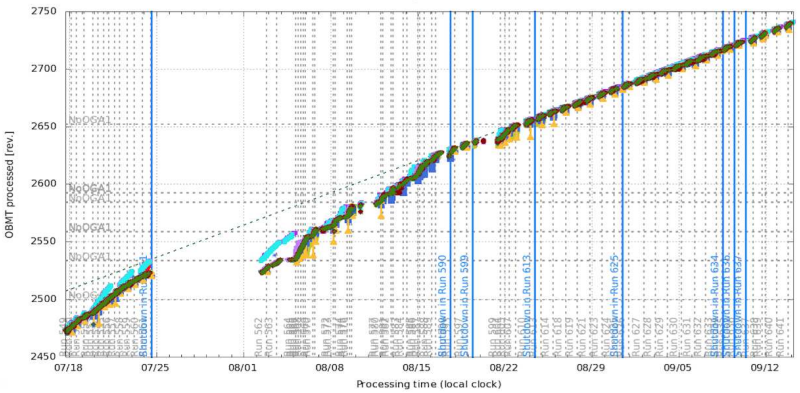

Figure 5. Procesing flow diagram showing constant delays and gaps

colour code is irrelevant, it clearly shows gaps and delays (when the coloured symbols are below the increasing dash line) consequences of the big number of unexpected and unknown contingencies impossible to anticipate.

4. A step further: the search for Antifragility through DevOps

While a robust system resembles a castle, designed to protect rigidly the inside and repel any attack from the outside it became clear that in order to fulfil the demanding requirements of the daily pipeline it was needed to get an integrated system capable of being flexible and adaptable: antifragile.
The concept of "Antifragility" involves being able to stress the system in a way that it provides continuous feedback that can be incorporated fast. In this way, the effort is not on repelling changes and problems (as it is not possible to foreseen all of them) but assimilating them to be stronger. And velocity is the key factor to achieve it.
DevOps appears in this context as the way to get antifragile operations. By embracing DevOps practices the pipeline processing became smooth even in cases of contingencies and the operations team increased their effectiveness:

- The processing of the daily pipeline became more stable over longer periods of time. Fig. 6 is a typical processing flow nowadays. The system is better able to keep up with the requirements on performance and data completeness (they are fulfilled practically at any time).
- With faster procedures lightened of "waste" (= what does not add any value) and more automated tools the downtimes were reduced (in number and duration).
- More realistic tests allowed us to constantly challenge the ability to detect and recover from failures.

- By having more time to spend on improving the system rather than "fire fighting" new ways to do the same with less are being found (implying saving costs, e.g. reducing the number of HW resources).
- Automatically gathered metrics guarantees the quality of the data produced.
- The team remains motivated, avoiding burnouts by a good priorization of the tasks and facing new challenges even though the system is consolidated.

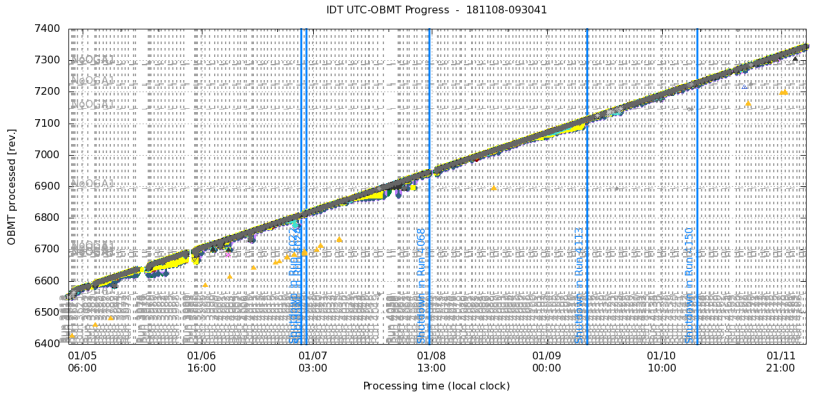

Figure 6. Typical current processing flow of the Daily Pipeline.

DevOps is often associated only with the usage of new tools and technologies as e.g. containers or microservices. But DevOps is not just a set of tools than mixed together does the magic. It requires a cultural change where the concept of automation, gathering metrics, lean (reducing waste) and collaboration are prioritized and always pursued. Obviously, without proper tools the collection of metrics or the processes automation become complicated tasks. Fig. 7 shows the technology stack used in the development and operations of the SOC Daily Pipeline. Some of them are very consolidated and key like Jenkins and some others (as e.g. Splunk) are currently being introduced.

It might be arguable whether the improvements referred above arrived naturally

Figure 7. SOC Daily Pipeline Technology Stack

just because over the time the systems were fixed and cleaned from their bugs and not because a new culture of fostering automation and collaboration were promoted. However, being the former true, embracing new practices have brought tangible benefits: systems have still bugs but there are mechanisms to find them fast causing minimal disruption. Manual intervention is less and less needed and the procedures are better tested so that operations are more reliable and the risks decreased.
The following sections describe the main achievements over the last years.

5. Continuous Integration, Delivery, Artifact Repository and Infrastructure

Jenkins (Fig. 8) is the main Continuous Integration tool and a major responsible for the system's stability. Jenkins pipelines have been recently introduced for the implementation and integration of complex jobs, encompassing building, testing, quality assurance and deployment, a big step towards automation.

SonarQube is the source analysis tool for continuous inspection of code quality that is

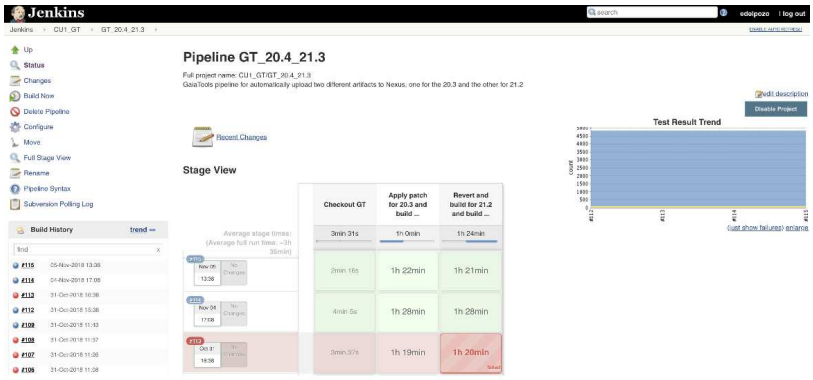

Figure 8. Jenkins usage in Gaia SOC

Figure 9. SonarQube at the Gaia SOC

fully integrated with Jenkins. It provides a centralized place to share these QA metrics.

Fig. 9 is a snapshot showing the type of information provided.
For storage of software artifacts Nexus Professional Repository Manager is the tool used.
The IT infrastructure is managed by the Science IT team at ESAC (SITU) that is also focusing efforts on automating most of their tasks. They use Puppet and Ansible for deploying, patching and configuring the items under their responsibility.

6. Testing Activities

The integration testing activities have always supposed a main challenge due to the difficulties of reproducing realisticly operational scenarios. The main constraint was dealing with the volume of the working Intersystems Cache database (30TB size on average). In the past it was very difficult and extremely time consuming to conduct one test in conditions similar to operations (let alone repeated tests). A major breakthrough was made applying NetApp (our storage vendor) technologies to clone the operational database any time with no impact in operations. It is now possible to do small, simple, fast and completely representative experiments and trials. Moreover, both environments, validation and operations can process the same incoming telemetry at the same time so the comparison between them is straightforward.
Fig. 10 describes the cloning mechanism. This method is not only valuable for testing but also it provides to the SOC with a backup of the database fully reliable (everytime a test is conducted the correctness of the backup is checked).

Future plans are using Jira for testing management and Cucumber for automating

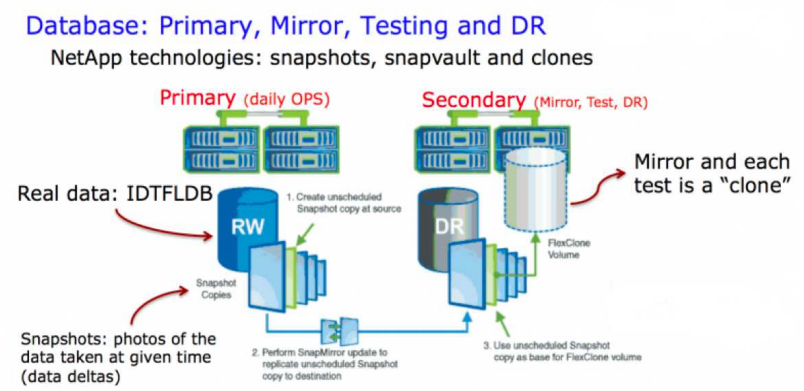

Figure 10. Mechanism to clone the operational database in validation

system tests.

7. Agile methodologies: Kanban

The Operations team adopted Kanban for the purpose of being more effective by visualizing the work. Kanban is based on three fundamental principles:

- Making the policies and procedures explicit. Visualizing does not only mean showing the progress of an activity but also sharing the problems found and the

procedures followed. In this way the team shares the knowledge and is able to e.g. think of better ways to do a task (by e.g. eliminating what is not needed - lean - or by automating specific parts).

- Limit work in progress (WIP). This is the limit for how many items each operator will work on at the same time. It is key for avoiding the time wasted every day in switching from one activity to the other and helps on finishng activities that otherwise would take much longer having many other to address at the same time.

- Manage the work flow. It is the team that prioritize the tasks and not the amount of tasks (planned or unexpected) that swamp the team.

The operations team members gather every morning. The physical whiteboard and the sticky notes are being replaced by a digital board in Jira. The main benefit of using Jira is the metrics provided, very much useful to identify trends that might mean bottlenecks or other problems. Since the adoption of Kanban the throughput of issues resolution has increased considerably.
The team invented a role (that rotates every week) known as "Person of Interest" in charge of resolving all the issues from the other team members. The POI reviews the procedures followed and the information provided, suggesting changes when things are not clear enough. This is an excellent way to keep all the team trained and aware of all the activities ongoing and also an additional way to improve the procedures and processes followed. Fig. 11 shows an example of the reporting obtained by Jira from the Kanban board.

Figure 11. Control Chart of the Kanban board

8. Procedures and Runbooks

The procedures and runbooks (understood as groups of procedures) are being moved from a static documentation to Jupyter Notebooks. Markdown text, executable code and output all inside a single document fit the procedures needs very well. The interactivty avoids they get obsolete as they are often executed and reviewed. In addition, automating the procedure steps decreases the risks of manual interventions.

The procedures report automatically into the operations logbook (eLog) saving unnecessary time to the operator. Fig. 12 is an example of a procedure implemented as a Jupyter document.

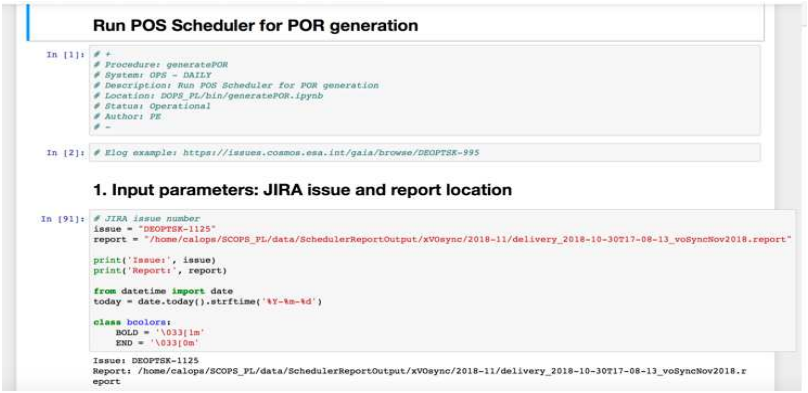

Figure 12. Procedure implemented in Jupyter Notebook - Python 3

9. Conclusions

By embracing some DevOps practices the Gaia SOC enhanced the stability of the pipeline processing and the effectiveness of the team. Automation, avoiding unnecesary tasks (lean), collecting metrics to get knowledge and collaboration are the fundamental drivers of the change. This is a continuous learning process though, that can't finish. The goal is to collect feedback of the behaviours of the processes and the team that can be used for improving them and achieving antifragile systems.