Evolving Argus to meet the GÉANT NOC requirements

Ilona Podliashanyk – Software Engineer, Sikt
Erik Reid – Head of Software Development, GÉANT
Hanne Moa – Senior Software Engineer, Sikt

NORDUnet Conference 2024, Bergen
11 September 2024

Argus Intro, Sikt perspective

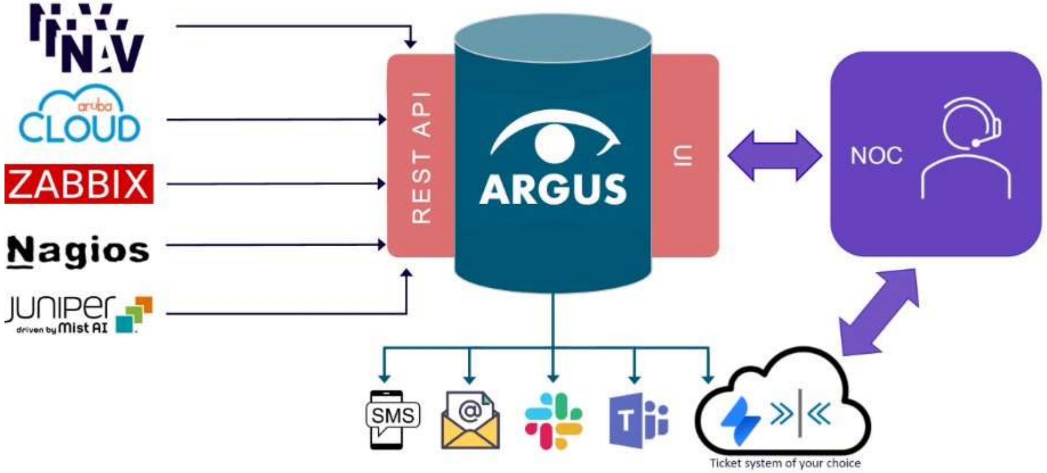
The plan

- Integrate Argus:
  - Source agnostic
  - Minimalistic features
  - Essential workflow
- Into GÉANT NOC workflow:
  - Source specific
  - Rich features
  - Nuanced, intricate workflow
- Given constraints:
  - Resources
  - Deadlines
- Success criteria:
  - No breaking changes in Argus
  - Updated GUI in GÉANT NOC
- Bonus:
  - More flexible and customizable Argus
  - Argus that is easier to develop and maintain
  - Experience a great cooperation within community

- What is Argus?
- The need for change
- The approach
- The results
- Why Does GÉANT Need A New NOC Dashboard?
- What Routes Were Considered?
- Why Argus?
- Experience so far
- Demo
- Q&A

Argus is...

- A tool to aggregate incidents from numerous monitoring systems into a single, unified dashboard
- Both an incident management system and a notification system
- Source agnostic and domain independent



Argus in a nutshell

In

- Any state to monitor:
  - alarms, events, changes, updates, deviations…
- One or more monitoring systems via glue-services:
  - NAV, Nagios, Juniper MIST, Aruba Cloud, Zabbix…
- Notification media plugin:
  - SMS, Email, MS Teams, Slack…
- Ticketing system plugin:
  - Jira, RT…

Out

- One screen to rule them all
- Essential filtering
- Ability to tag alarms with arbitrary metadata
- Ability to classify by severity
- Personalized notifications
- Automated ticket generation

The need for change

- Make Argus more easily fit several different workflows
- Mitigate the frontend bottleneck
- Mitigate the dev bus factor

The approach

- Complete rewrite of React frontend using:
  - HTMX
  - Tailwind CSS
  - DaisyUI
- Sikt dev team provides ability to customize existing components and add custom made elements
- GÉANT dev team develops and integrates their custom components into Argus

The results

Before	After	By
Bus factor: 1 frontend developer	(Potentially) 5 full stack developers	Switching to HTMX+Django, daisyUI
Excessive React complexity	KISS, no framework boilerplate	Switching to HTMX
Dependency hell	Dependency minimalism, simple and lightweight	Dropping npm
Complex deployment	Simpler deployment, one less moving part	Merging frontend and backend
	Backend developers having the option to style the UI components	Using daisyUI (ready-made and little opinionated components) and Tailwind CSS

Costs/Benefits

Costs

- Had to throw half of dev resources overboard
- Lower priority on other systems we work on
- Fundamental data model changes

Benefits

- Lots of development in short time
- Customizable, personalized and themeable Argus dashboard
- Synergy between teams

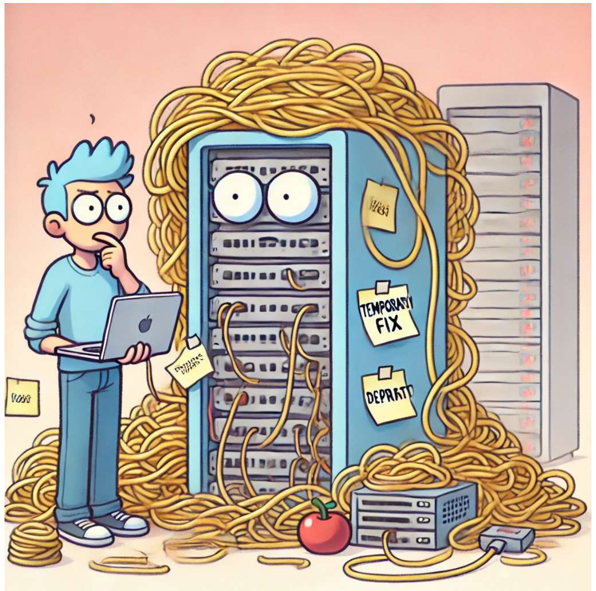
GÉANT NOC Perspective

Bringing Argus into the GÉANT NOC

- Why?
- How?
- Why Argus?
- Experience so far …

Why Does GÉANT NOC Need A New Incident Dashboard UI?

The usual software development suspects and problems …



- Current GUI architecture hasn’t aged well
  - Overcomplicated, bloated abstractions
  - Hard to maintain: changes take a long time/hacks vs. big refactoring work
- Legacy software stack
  - old Spring Framework, old Tomcat, (too-)complex deployment
  - developers prefer different tech stacks, e.g. Python/etc.

How? (i.e. What Alternatives Did We Consider?)

- Integrate with some popular product?
  - E.g. Zabbix, LibreNMS, whatever …?
    - Silver bullet mentality? …there would still be lots of integration/glue work to get back to where we are now
- Full DIY rewrite?
  - Pro: We know what & how. And would be fun.
  - Con: Really? Are we so special there's nothing we can leverage? Really?
- Argus?
  - Pro: Buzz. Community-developed.
  - Con: Still lots of integration. *Gaps* between the Argus use case and ours.

Why Argus?

A quick look at some of the gaps we found …

- Argus
    - Alarm aggregator …
        - Source-agnostic: lower-level details are mainly in the source NMS
        - … therefore correlation between sources is mostly manual
    - TTS integration: straightforward but nominal

- GÉANT NOC Dashboard
    - Tightly integrated with NE's, config, services
    - Automated upstream correlation
    - TTS integration: business rules are encoded in the correlation logic
    - Different columns & views (e.g. NE/POP details, multi-level Ack)
    - Incidents have complex lifecycles, evolve over time
    - Extra functionality and "widgets" specific to the GÉANT stack

Why Argus?

Safest path was DIY …

- Maintain full control over architecture and changes
- Maybe not more work that integrating with another framework …?

Ultimately collaborated with the Argus team because …

- Nice abstraction between incident generation vs presentation
- Maybe the multi-source behavior would be useful someday
- Community-developed, leverage resources
- Tolerance for risk: GUI is a small part of the overall NMS
- ***… and mainly: the GÉANT and Sikt teams clicked***

Where We Are Now …

Lower-level

Port

Baseline

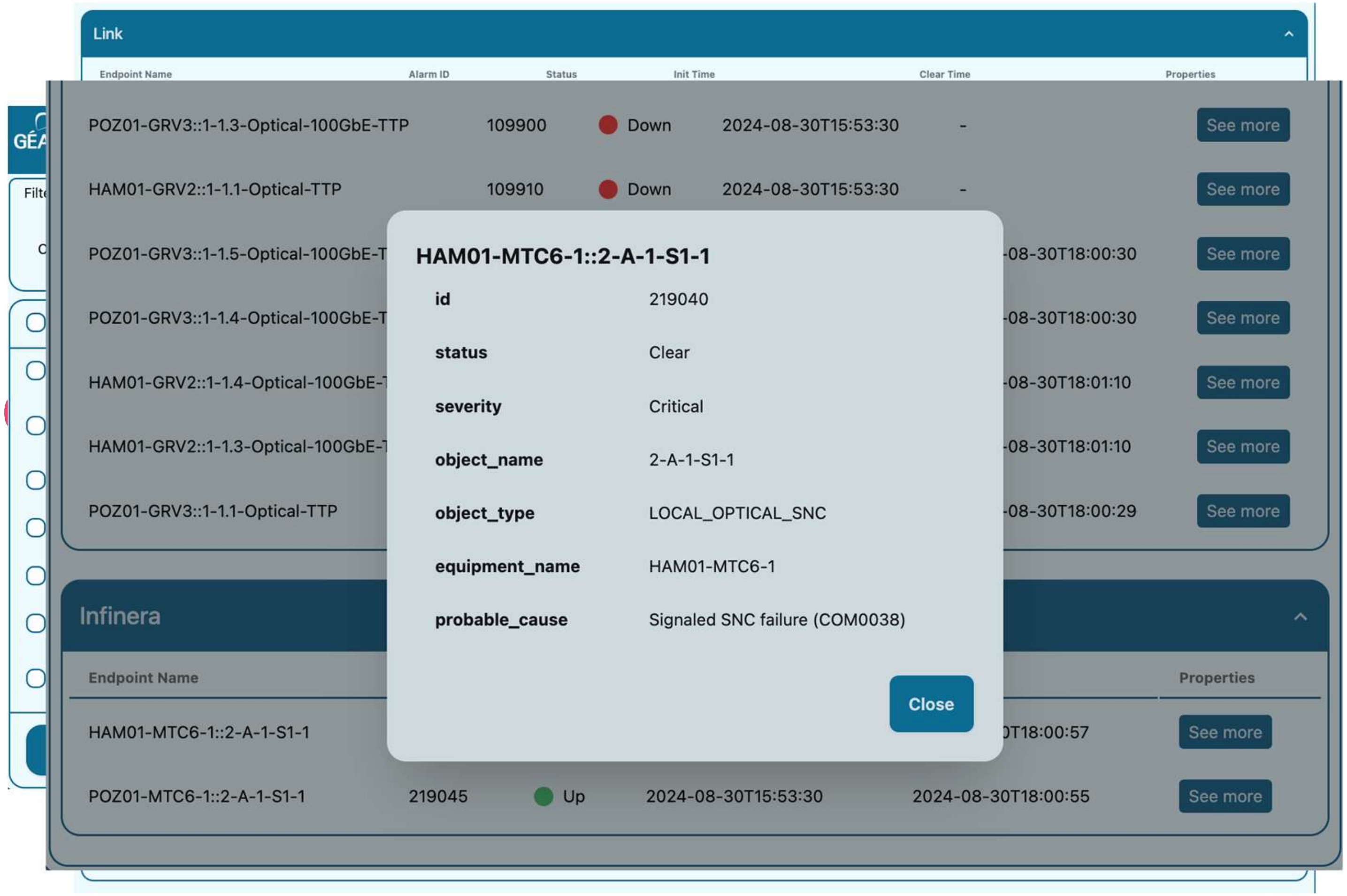
Incident

Details

Incident View

Details

C



Link

Endpoint Name	Alarm ID	Status	Init Time	Clear Time	Properties
POZ01-GRV3::1-1.3-Optical-100GbE-TTP	109900	● Down	2024-08-30T15:53:30	-	See more
HAM01-GRV2::1-1.1-Optical-TTP	109910	● Down	2024-08-30T15:53:30	-	See more
POZ01-GRV3::1-1.5-Optical-100GbE-T				-08-30T18:00:30	See more
POZ01-GRV3::1-1.4-Optical-100GbE-T				-08-30T18:00:30	See more
HAM01-GRV2::1-1.4-Optical-100GbE-T				-08-30T18:01:10	See more
HAM01-GRV2::1-1.3-Optical-100GbE-T				-08-30T18:01:10	See more
POZ01-GRV3::1-1.1-Optical-TTP				-08-30T18:00:29	See more

Infinera

Endpoint Name					Properties
HAM01-MTC6-1::2-A-1-S1-1				T18:00:57	See more
POZ01-MTC6-1::2-A-1-S1-1	219045	● Up	2024-08-30T15:53:30	2024-08-30T18:00:55	See more

HAM01-MTC6-1::2-A-1-S1-1

id	219040
status	Clear
severity	Critical
object_name	2-A-1-S1-1
object_type	LOCAL_OPTICAL_SNC
equipment_name	HAM01-MTC6-1
probable_cause	Signaled SNC failure (COM0038)

Experience so far…

- Lots of enthusiasm about making a platform that works for multiple organizations
  - Teams: right people, right skills and knowledge
  - Same vision: no wall between “ours” vs. “yours”
- Found a lot of common work to share
  - Filtering framework, sorting, etc.
  - Baseline Incident table is mostly shared
  - Collaborate on styling
  - … expect this to continue: notification abstractions, common TTS things, etc.
- Django is nice, flexible, easy to use, extend - everybody likes it
- Actual custom rendering code is quite small
  - *Be careful with org-specific modules: important to keep the big picture in mind*
- Abstract separation between Argus and GÉANT Correlator is not 100%
  - Not completely solved – some synchronization is still necessary
  - Some non-abstracted direct access is still present

DEMO!

www.geant.org

Co-funded by
the European Union

Resources

- Argus overview: https://network.geant.org/argus/
- Argus documentation: https://argus-server.readthedocs.io/en/latest/
- Argus repo: https://github.com/Uninett/Argus
- HTMX Argus repo: https://github.com/Uninett/argus-htmx-frontend
- GÉANT Argus repo: https://github.com/GEANT/geant-argus

Thank You

Any questions?

Contact us at:

- **Ilona Podliashanyk: ilona.podliashanyk@sikt.no**
- **Erik Reid: erik.reid@geant.org**
- **Hanne Moa: hanne.moa@sikt.no**

www.geant.org

Co-funded by
the European Union