

Ensuring 24x7 Operations of IRAM 30m Telescope

 IRAM Granada · AJ Vázquez (avazquez@iram.es) · 2026.10

Index

- Introduction
- IRAM 30m radio-telescope
- Astronomical facilities
- Architecture and Philosophy
- Deployment overview
- Use cases
- Conclusions

Introduction

- **Who are we?** *Institut de radioastronomie millimétrique*: Europe's leading research center on millimeter radio-astronomy
- **What is the 30m radio-telescope?** One of the world's largest and most sensitive single-dish millimeter radio-telescopes
- **Who am I?** Head of the 30m Computing Group
- **What do we monitor?** Hardware, software, operations metadata
- **Why do we monitor?** Reliability: maximizing observing time means doing more science



IRAM 30m Radio-Telescope (Millimetric Radio-Astronomy)

- 24-7 operations since 1984 except weather -atmosphere opacity-, contingencies, and planned stops:
 - Short window of maintenance on Tuesdays (stop ~5 hours / week)
 - Heavy maintenance during summer (stop ~1 week / year)
 - Antenna Motors and Surface maintenance recently (stop ~few months / 40 years)
- Teams: Astronomy, Operation, Frontend, Telescope, Computing, Administration



Astronomical facilities: overview and challenges



Observatories

- Industrial+IT+scientific environment
- Heterogeneous and complex systems
- Certain systems receive little updates in decades
- Zabbix used in some observatories but focused on IT
- SCADA-based solutions more typical
- Grafana widely used



Objectives

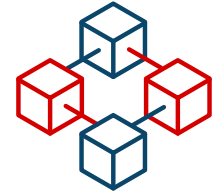
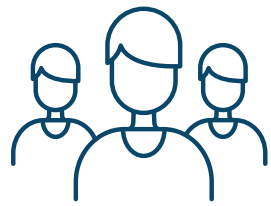
- Minimize downtime
- Maximize observing time
- Improve reliability of systems
- Minimize effects of human errors



Challenges

- Team size and workload
- Interwoven legacy heterogeneous systems
- Heterogeneous data sources and data policies
- Time is precious, small time windows for interventions

IRAM 30m Radio-Telescope: resources

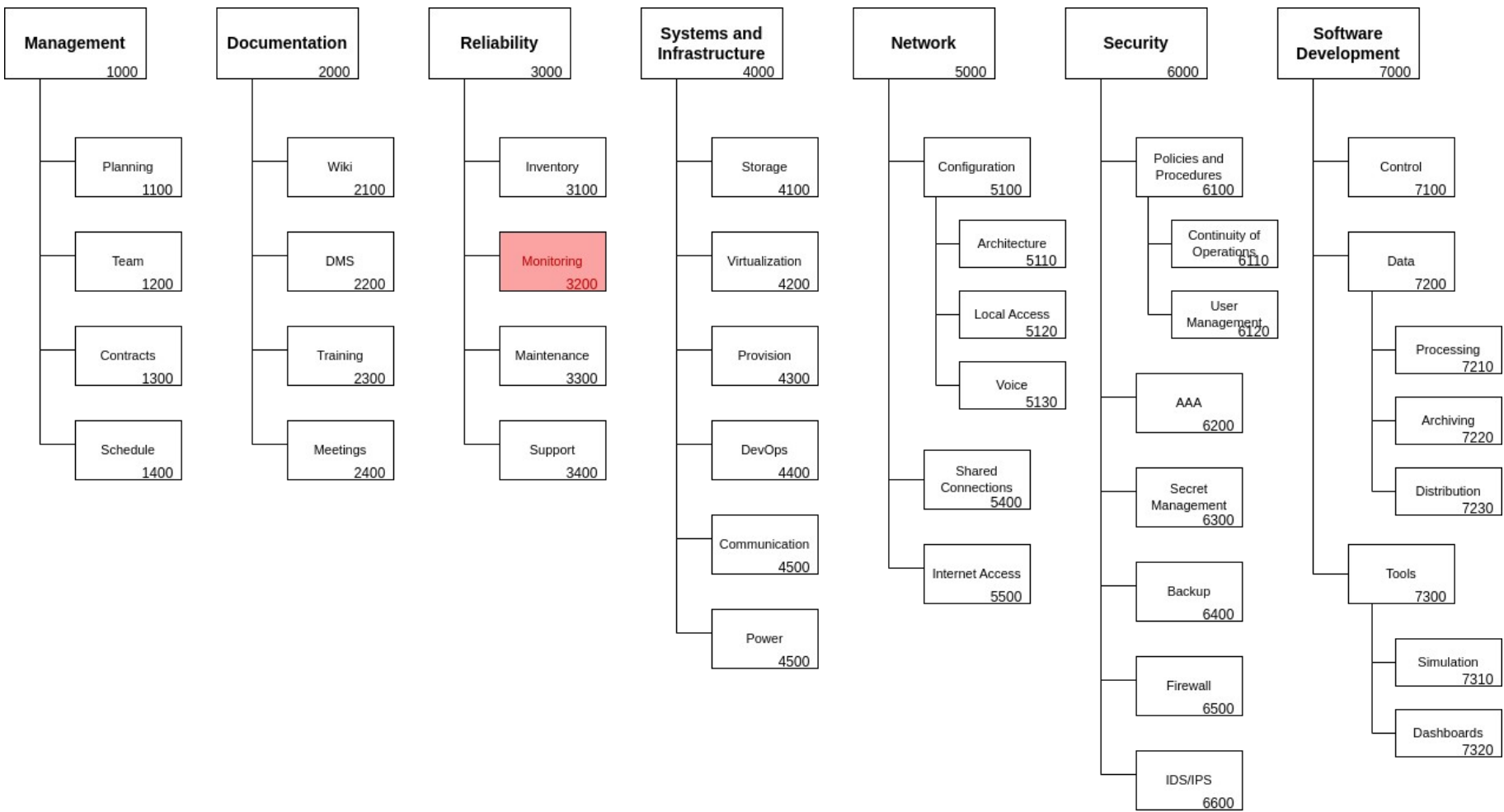


Computing Group

- 1x Head
- 1x SysAdmin+
- 2x Developer+

Areas

- See WPS below

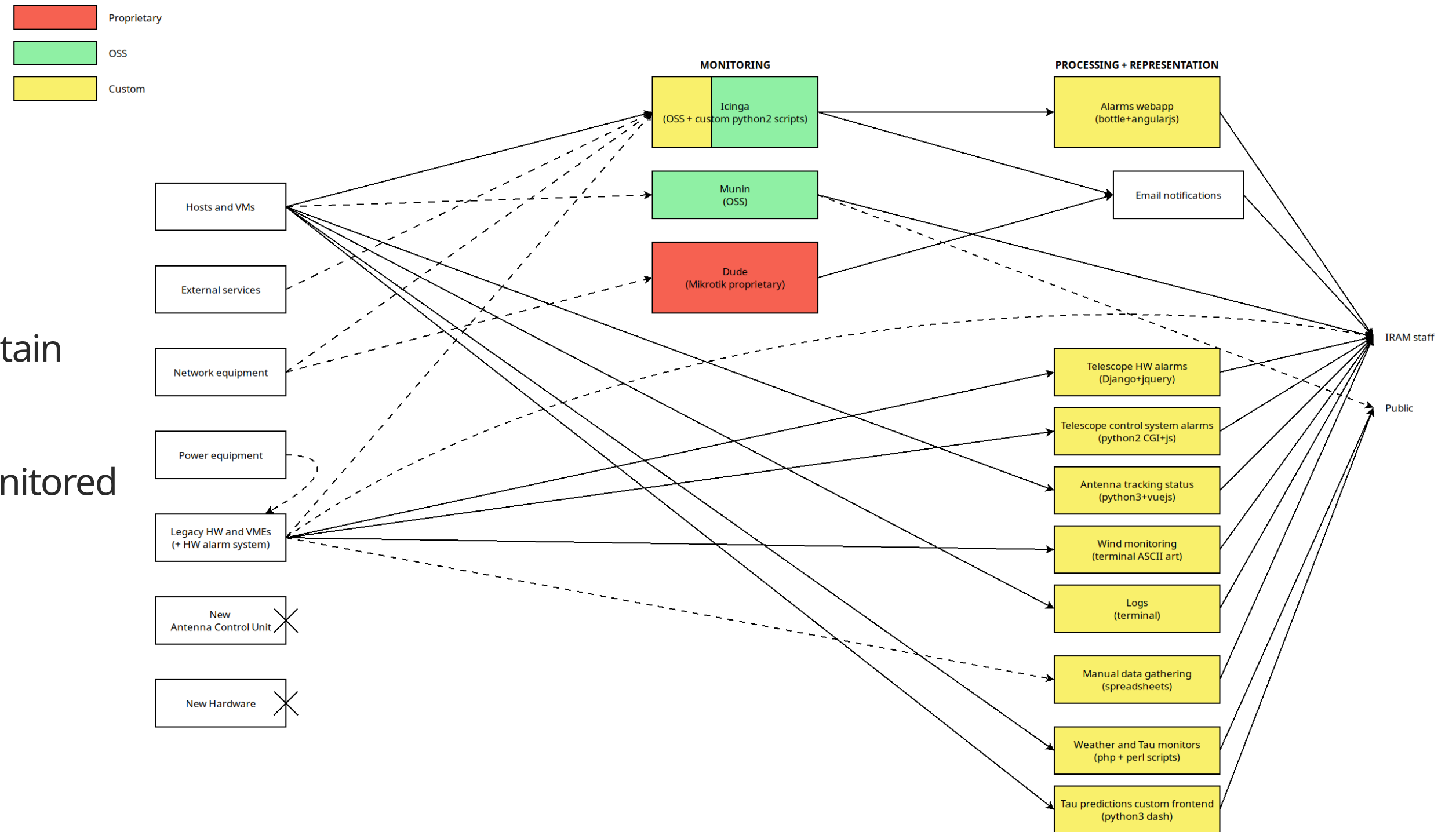


Facilities

- Telescope:
 - Telescope systems
 - Data Center
 - Operation room
 - Offices
- Granada:
 - Data Center
 - Offices

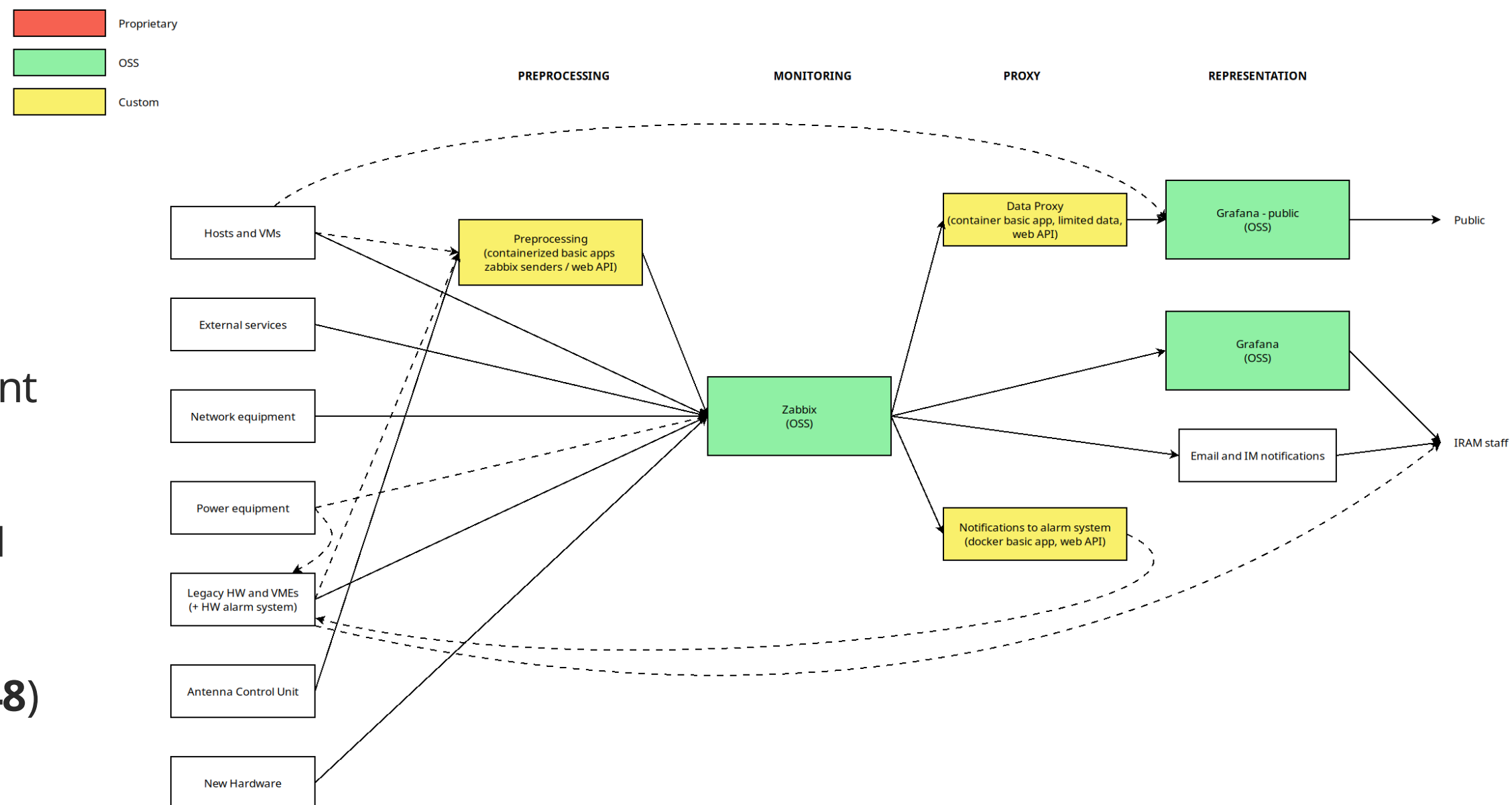
Legacy monitoring architecture

- Heterogeneous and hard to maintain
- Monitoring systems not updated
- Recently added hardware not monitored
- History only for certain items



Current monitoring architecture

- Combine data from multiple sources
- Single-pane-of-glass for operators
- Leverage OSS and reduce development
- All systems with periodic updates
- Integration with alarm system and IM
- More **hosts**: from ~50 to ~250 (**x5**)
- More **items**: from ~500 to ~24000 (**x48**)
- History for all items
- Proxy for public data

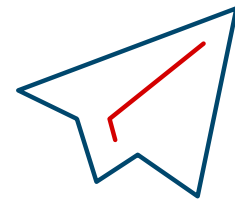


Philosophy



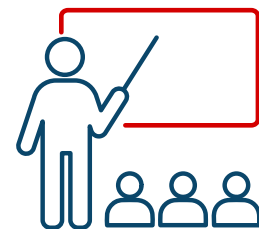
Simplified and centralized information

Zabbix as the system core, integrated with low level alarm system, combining information from multiple sources



Structured alert approach

From chained approach depending on operators towards conditional based type of alert

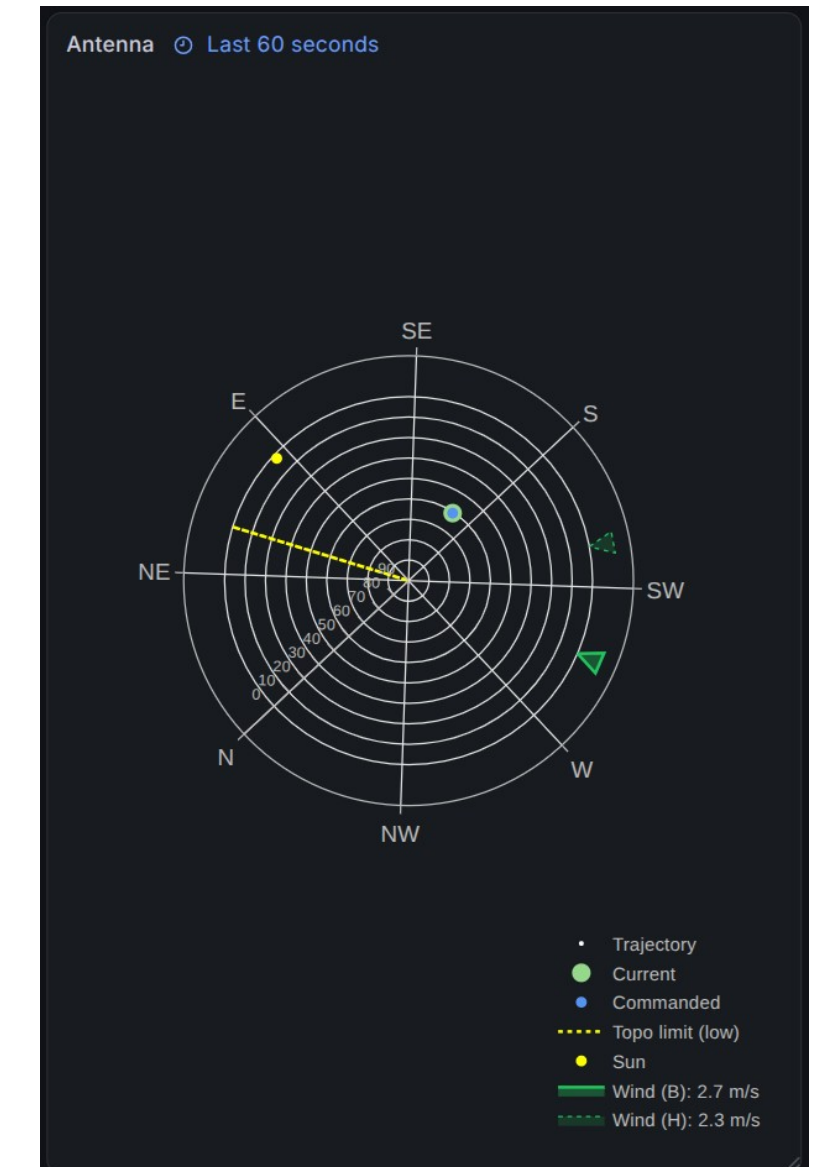
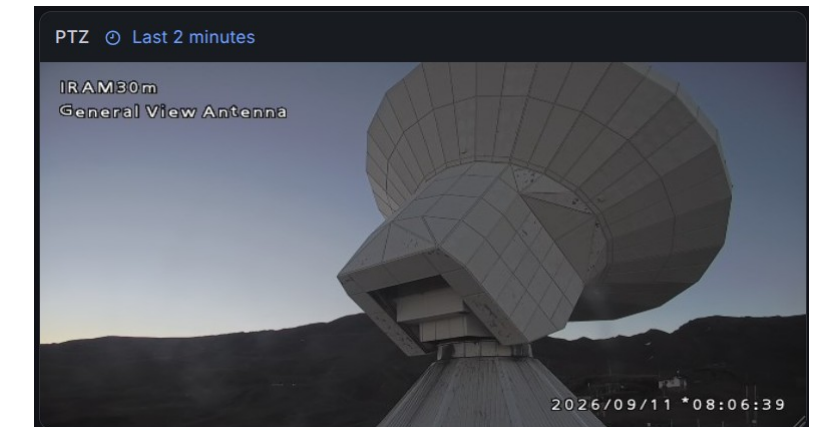


Continuous improvement

From reactive to proactive (less firefighting and more prevention)

Deployment: infrastructure

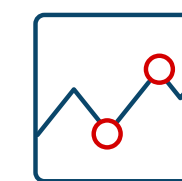
- Infrastructure:
 - Hardware:
 - Servers
 - Firewalls, router, switches, radiolinks
 - NAS
 - **PLCs (instruments, cryogenics controllers, antenna control unit, etc.)**
 - UPS, ATS, PDUs
 - NVRs, cameras
 - **Industrial I/O devices (sensors, actuators, etc.)**
 - VMs and Software:
 - Containers, processes, REST APIs, websockets, etc.
 - Metadata:
 - **Observation metadata**
 - External sources:
 - External websites



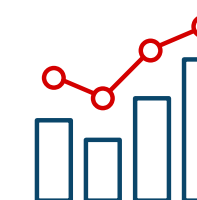
Deployment: challenges

Monitoring periods:

- **IT:** ≥ 1 minute
 - Services running, data pipelines generating data, web scenarios, etc.
- **Operations:** 1 second
 - Critical data (e.g.: antenna vibration), sensible equipment (e.g.: sun avoidance), tracking control (e.g.: tracking errors detection), etc.
- Operations **high precision:** < 1 second
 - Wind (e.g.: wind gusts detection), data from PLCs (e.g.: store all data for a certain time)



ZABBIX



influxdb™

Additional requirements on some items:

- High-density and combination of wide and short windows (delivering high performance)
- Aggregation allows to keep all trigger logic in Zabbix
- Live ephemeral (not persisted) data served via WebSocket (< 1 second)



Single Pane of Glass

- Unify and simplify data display for operators and other users



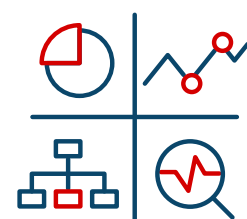
Grafana

Deployment overview: metrics



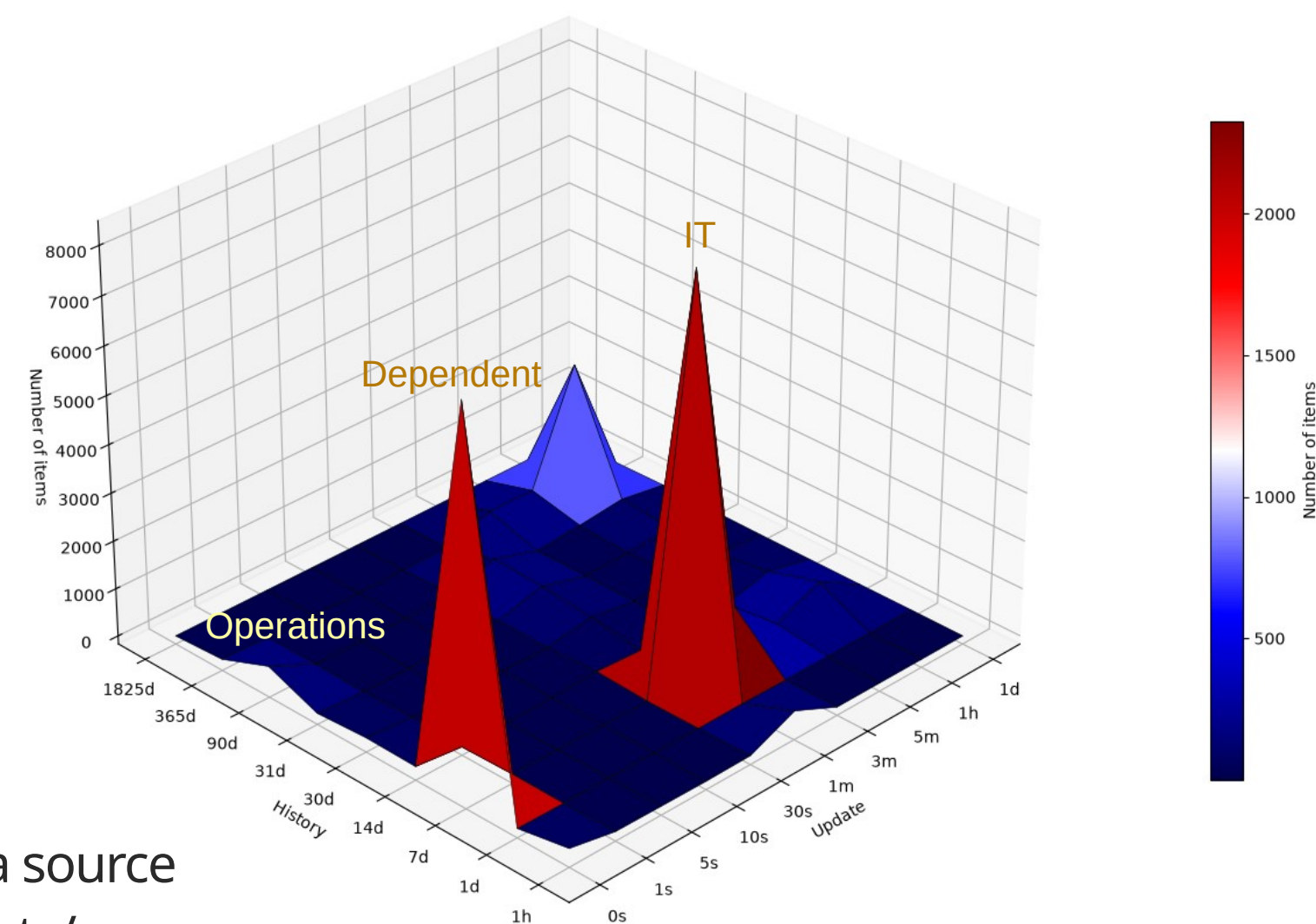
Zabbix 7.0 LTS

- Docker
- Periodic updates
- Setup running for ~3 years
- Integrated with telescope alarm system, IM (Zulip) and email notifications



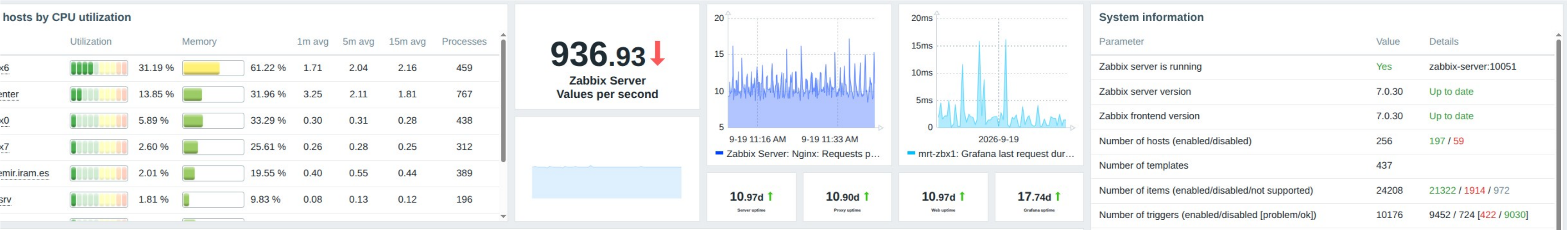
Some metrics

- Two sites
- 1 server and 1 proxy
- Around 250 hosts
- 24k items
- 9k triggers
- 1k values/second
- Serving Grafana ZBX data source typically at 10~250 requests/s

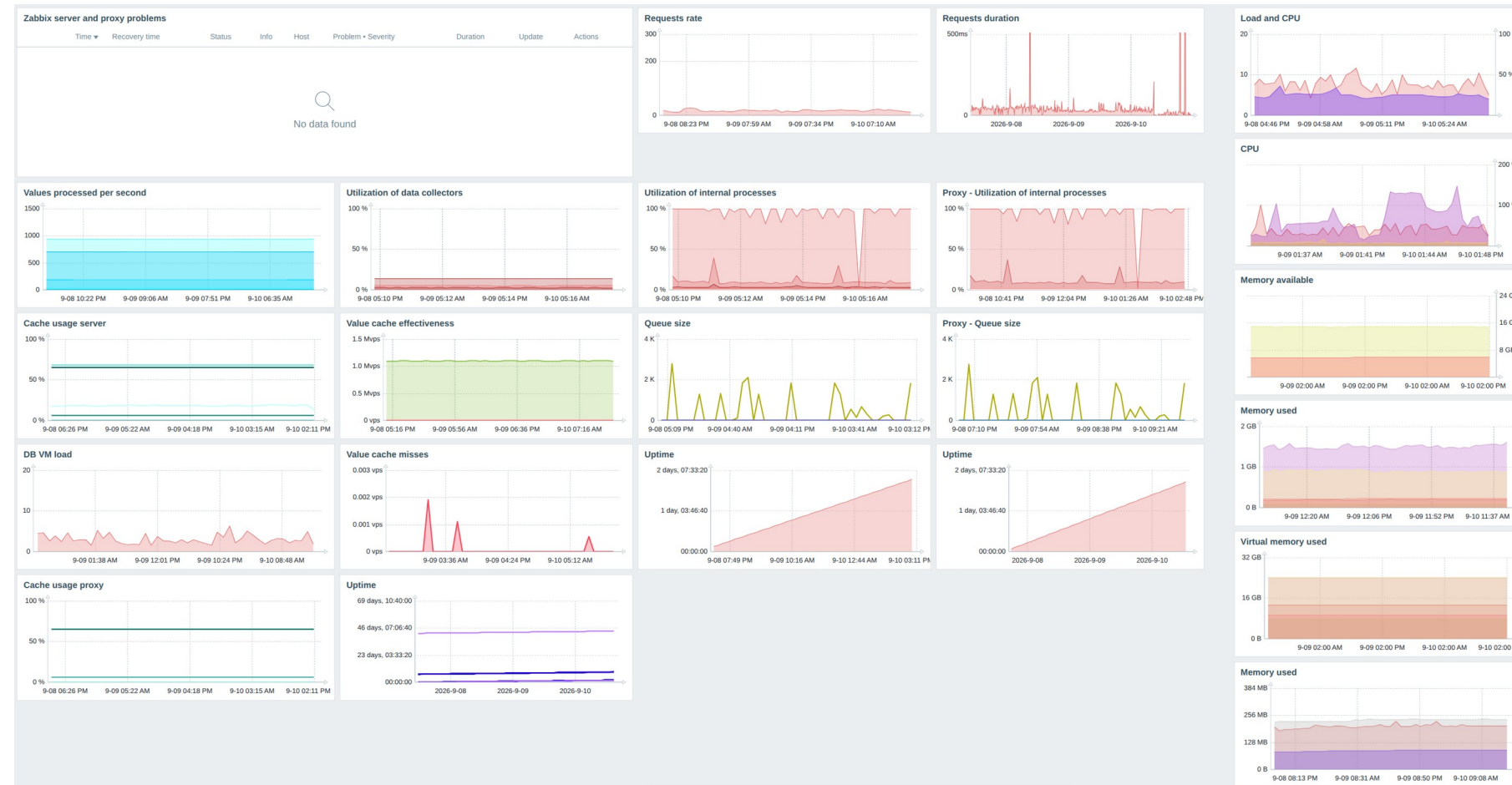


Deployment overview: health (I)

- Main dashboard extra info:
 - Nginx requests per second
 - Grafana last request duration
 - Active problems:
 - operations-related
 - general
 - software
 - hardware
 - network



Deployment overview: health (II)



Performance tuning/debug

- Basic Zabbix health dashboard
- Cache usage + DB load
- Internal processes server and proxy
- Queues sizes
- Requests

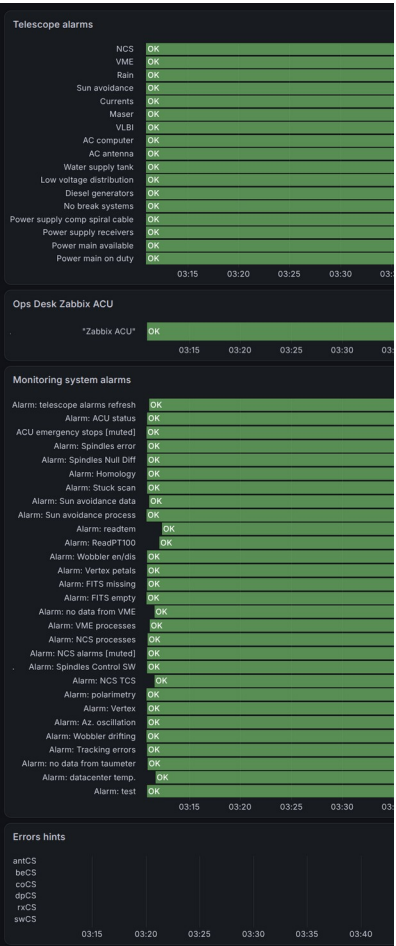
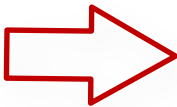
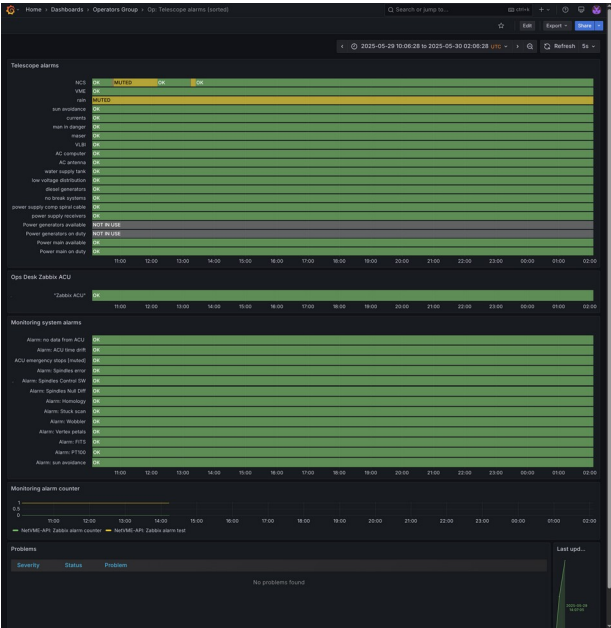
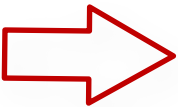
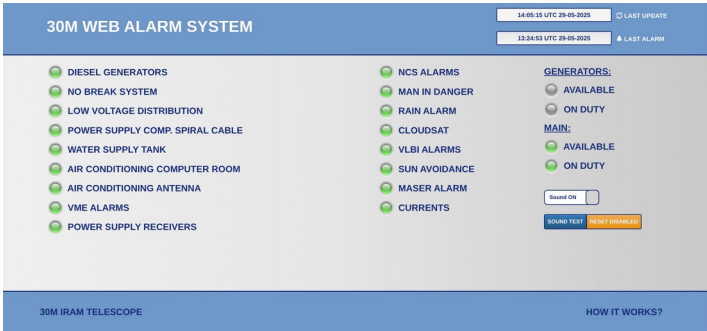
Zabbix upgrades

- Most upgrades are fine
- Few occasions: internal processes causing false alarms and high load over time, revert / configuration tuning / mitigation / upgrade
- Performance tuning tricky

More dashboards

- Grafana

Use cases (I): Telescope alarms



Snapshot

- Focus on current status

History + combined alarms

- New alarms integrated in existing system combining data from multiple sources

History + combined + hints

- Debugging that used to take hours hours now takes seconds
- Detect quickly if data processing pipeline stops
- Detect quickly issues related with tracking
- Etc.

Use cases (II): Operations debugging

Visual debugging

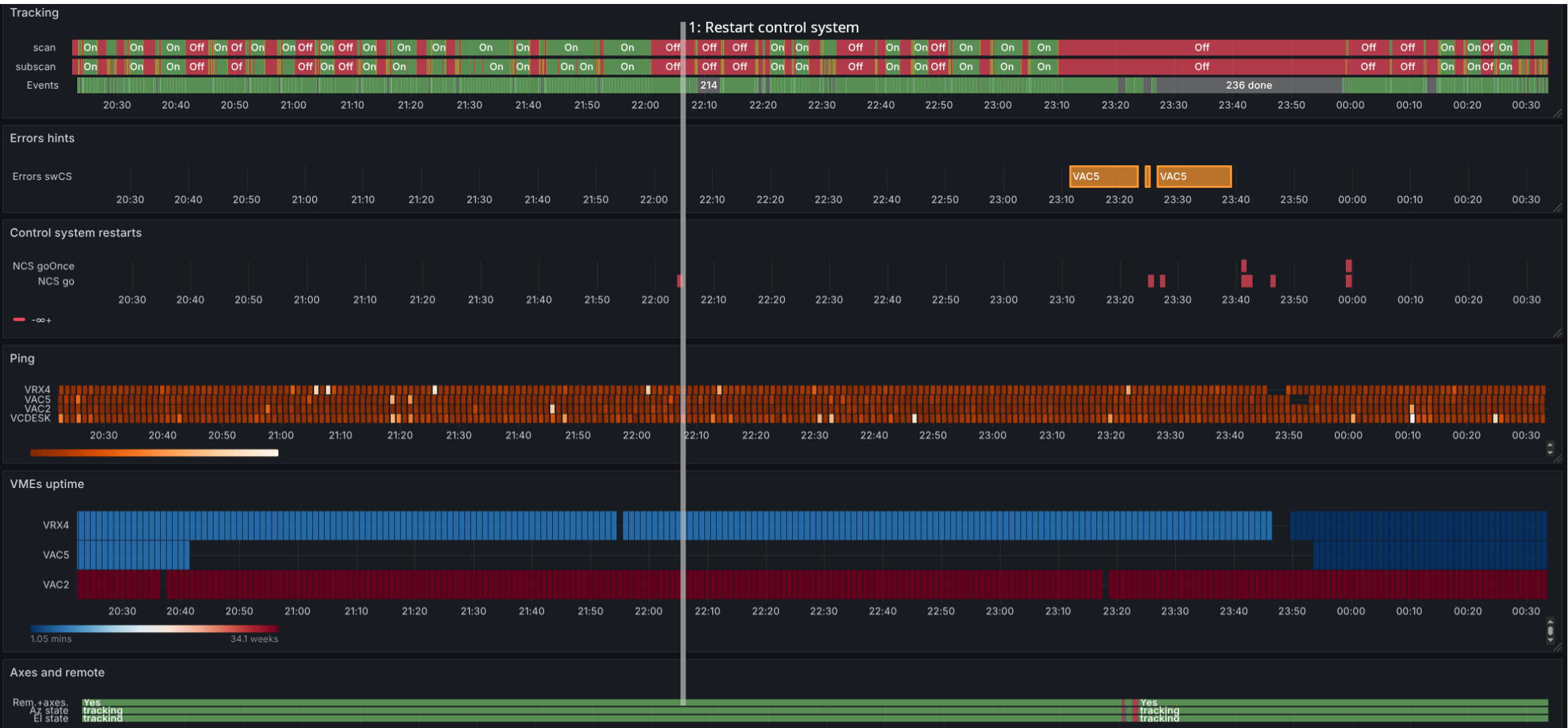
- Debug in minutes what would take ~1 hour of extensive testing
- Operations-oriented alerts

Overall increase of MTBF

- Some issues are now detected before they affect observation and can be fixed seamlessly

Overall decrease of MTTR

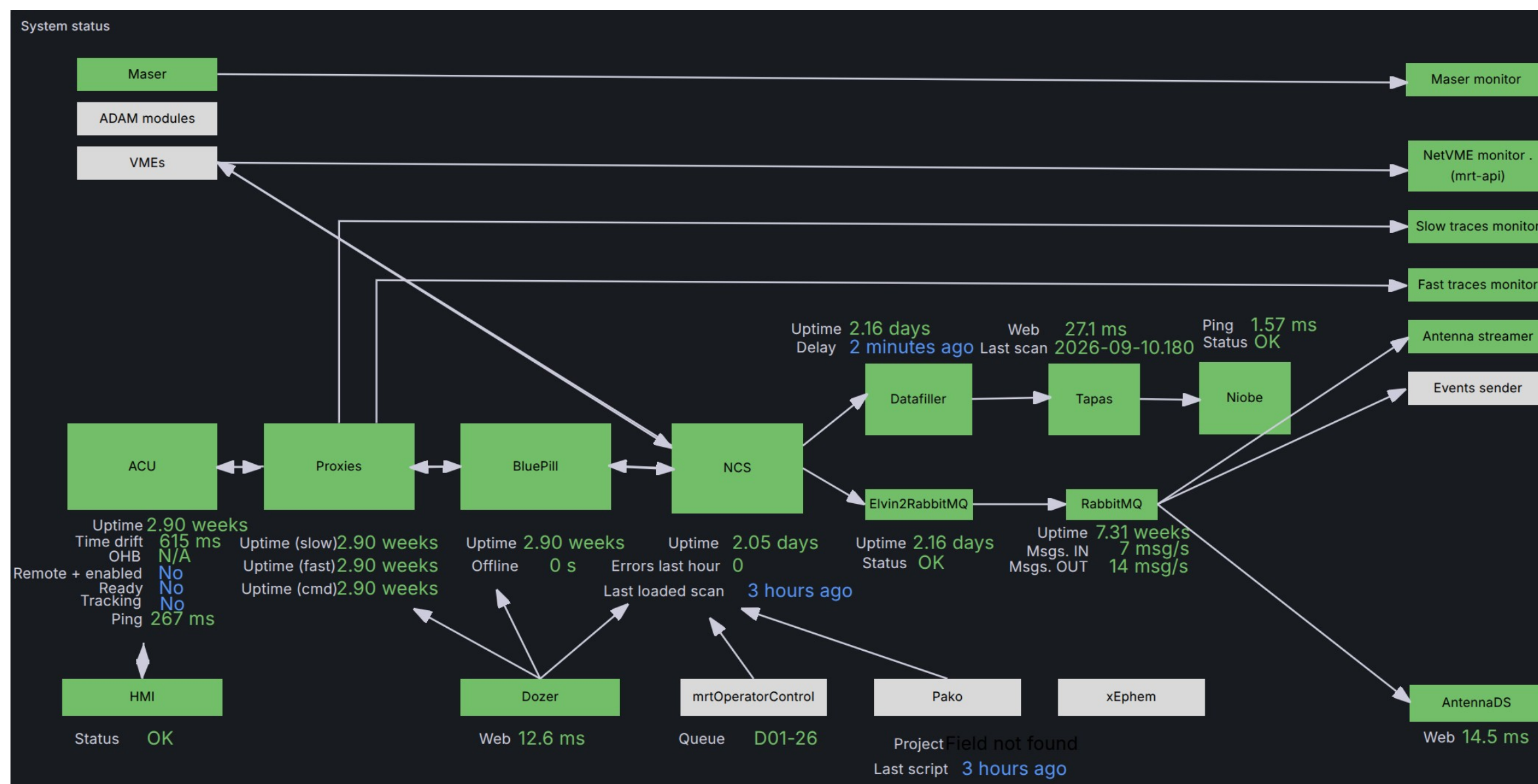
- Some issues are now evident to debug (error hints)



Use cases (III): System status

Visual representation of systems for quick status check / debugging

- Certain checks converted into alarms (minimize downtime)
- Unexpected issues: previously manual debugging (ssh, check logs...), now go through a few dashboards



Use cases (IV): temperature and humidity monitoring

Data centers temperature monitoring

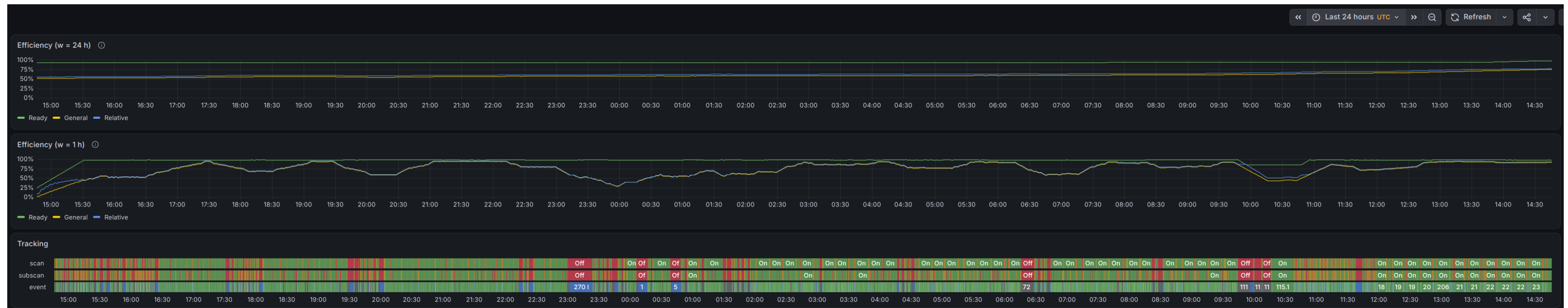
- Four sensors per rack (front/back top/bottom)
- Alarms for ambient absolute thresholds
- Precise monitoring allowed to properly configure A/C air distribution
- Detection of unexpected trends during certain periods



Use cases (V): observing efficiency

Monitoring observing efficiency in real time

- Traditionally calculated based on operators statistics
- Ready (no stops due to weather, maintenance, etc.), general (time taking data), relative (compared to ready)
- Automatic instant, monthly, and yearly measurements of observing efficiencies



Conclusions

- Migration from legacy architecture mostly complete:
 - Improved reliability and operations (UX/OX), not only for alerting, also for debugging
 - Integration with InfluxDB for sub-second monitoring (polling Influx, alerting in Zabbix): alternatives?
 - Not simply a migration: covering many more cases and evolution towards smart operations
 - Continuous improvement
- Gotchas (many along this path, but some worth mentioning):
 - Working at 1 second intervals is not trivial (said softly)
 - Importance of `count(/host/item,10s:now-10s)` vs. `count(/host/item,10s)` (midnight history sync effects?)
- Community:
 - Increase contact with the monitoring community (conferences/training/support) to be up to date
- Future works:
 - High availability: minimize revert/tuning/mitigation after upgrades with issues (e.g.: libcurl memory leak), avoid data loss (some issues do not manifest until hours after the upgrade).
 - SLA measurements for critical systems
 - Model Context Protocol (MCP). Interested in experiences of other users