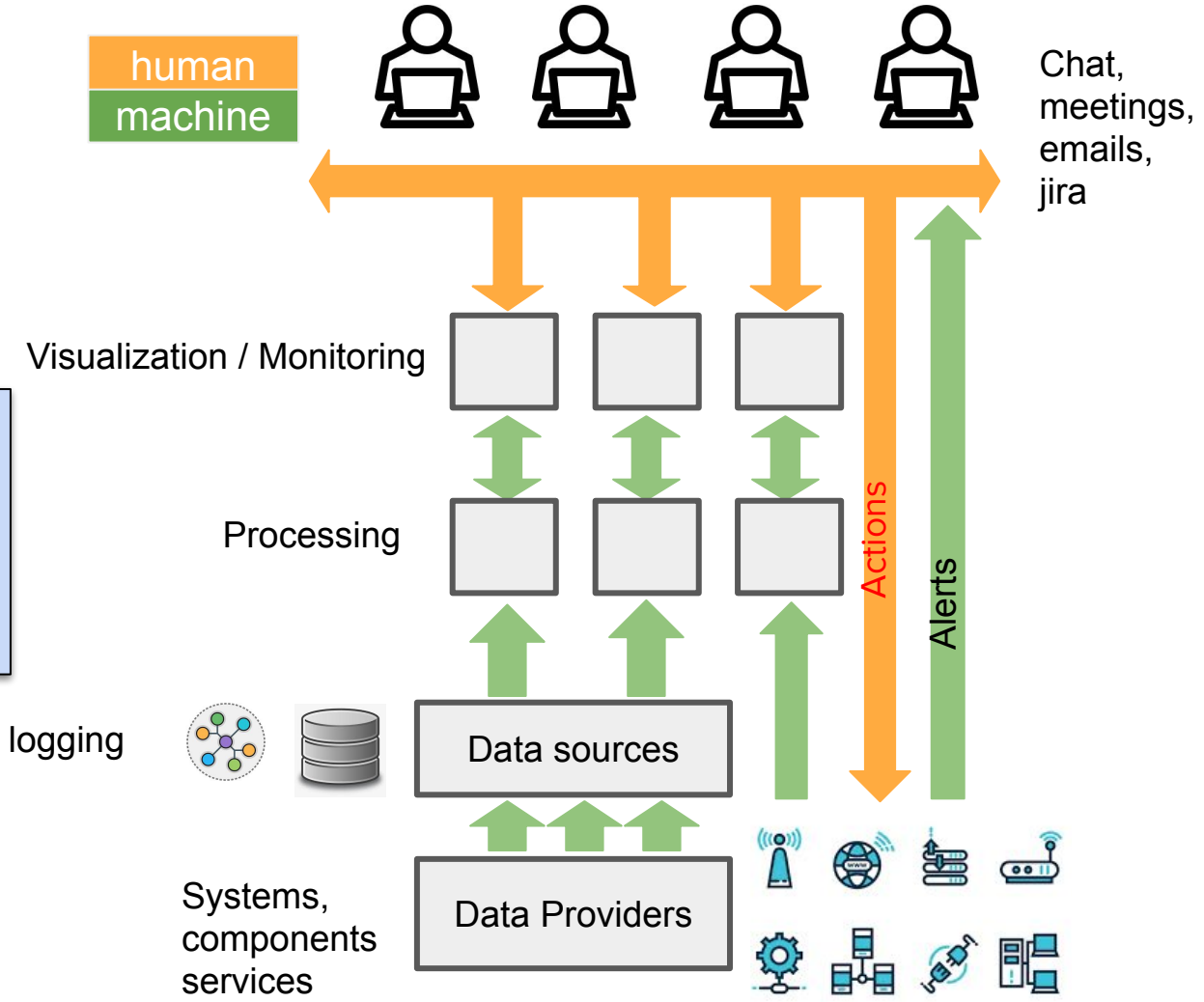
Operational intelligence (**OI**) is a category of real-time dynamic, business analytics that delivers visibility and insight into data, streaming events and business operations. OI solutions run queries against streaming data feeds and event data to deliver analytic results as operational instructions. OI provides organizations the ability to make decisions and immediately act on these analytic insights, through manual or automated actions.

Optimizing Computing Operations (and not only)

Alessandro Di Girolamo
on behalf of the Operational-Intelligence@cern.ch community

Operations today

ATLAS/CMS: 100+ people involved in Computing Operations (**50+ FTEs/experiment**)!
In 1 year:
> 1k GGUS tickets for ATLAS, > 2k for CMS



Operations today: common points

- We reviewed operations in several experiments (ATLAS, CMS, LHCb, DUNE)
  - Several teams responsible for own services + central team
  - Multiple ways to interact (mails, meetings, chats, ...)
  - Several monitoring components (many custom)
  - Trivial tasks already automated (via custom scripts)
  - Non-trivial tasks hard because typically require collection and correlation of information from several sources
  - Documentation scattered in several places+expert knowledge
- As a result, operations require:
  - a lot of **human effort**
  - learning a **lot of tools**
  - **Long training time** for newcomers

Can we do better?

- **Standardize operation procedures:**
  - Use common technology to store and access monitoring information
  - Use common visualization tools, alerts system
  - **Automatize common operational workflows**
  - **Keep flexibility to fit all/new use cases**

- **Add intelligence to operations:**
  - LHC experiments built a successful computing ecosystem for LHC Run1-2, but at which depth do we fully "**understand**" it?
    - Can we perform precise modelling of the system?
    - Can we use this modelling to make predictions?

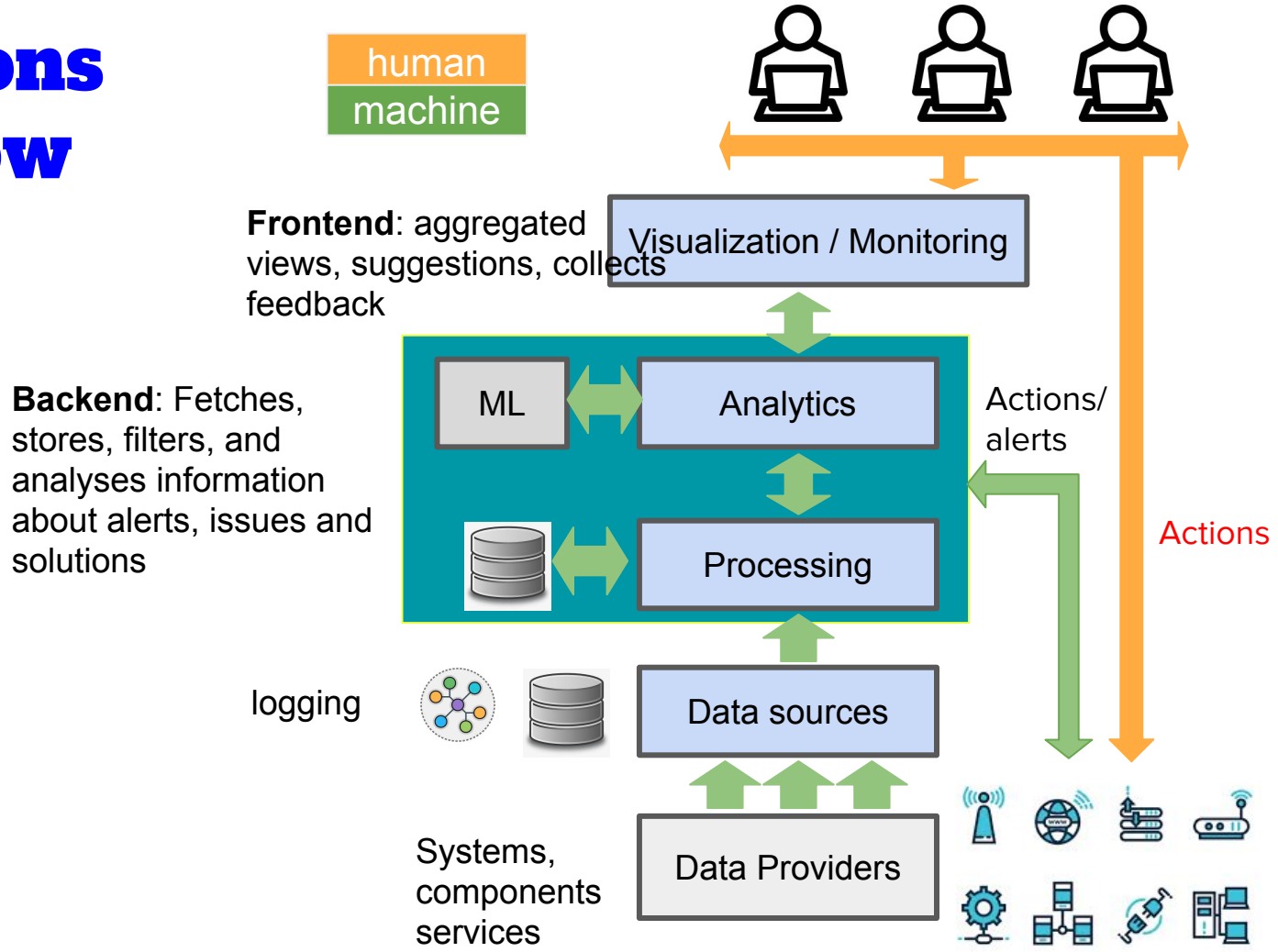
Operational Intelligence - the Mission

- Improve resource utilization
  - by implementing a set of recommendation algorithms learned from the existing dependencies between user actions and operational results
- Minimize human effort for repetitive tasks
  - And in general wherever possible
- Build a community of technical experts: critical mass to have impact on concrete and common issues while setting up sustainable tools.
  - Do not re-invent each time the (same) wheel

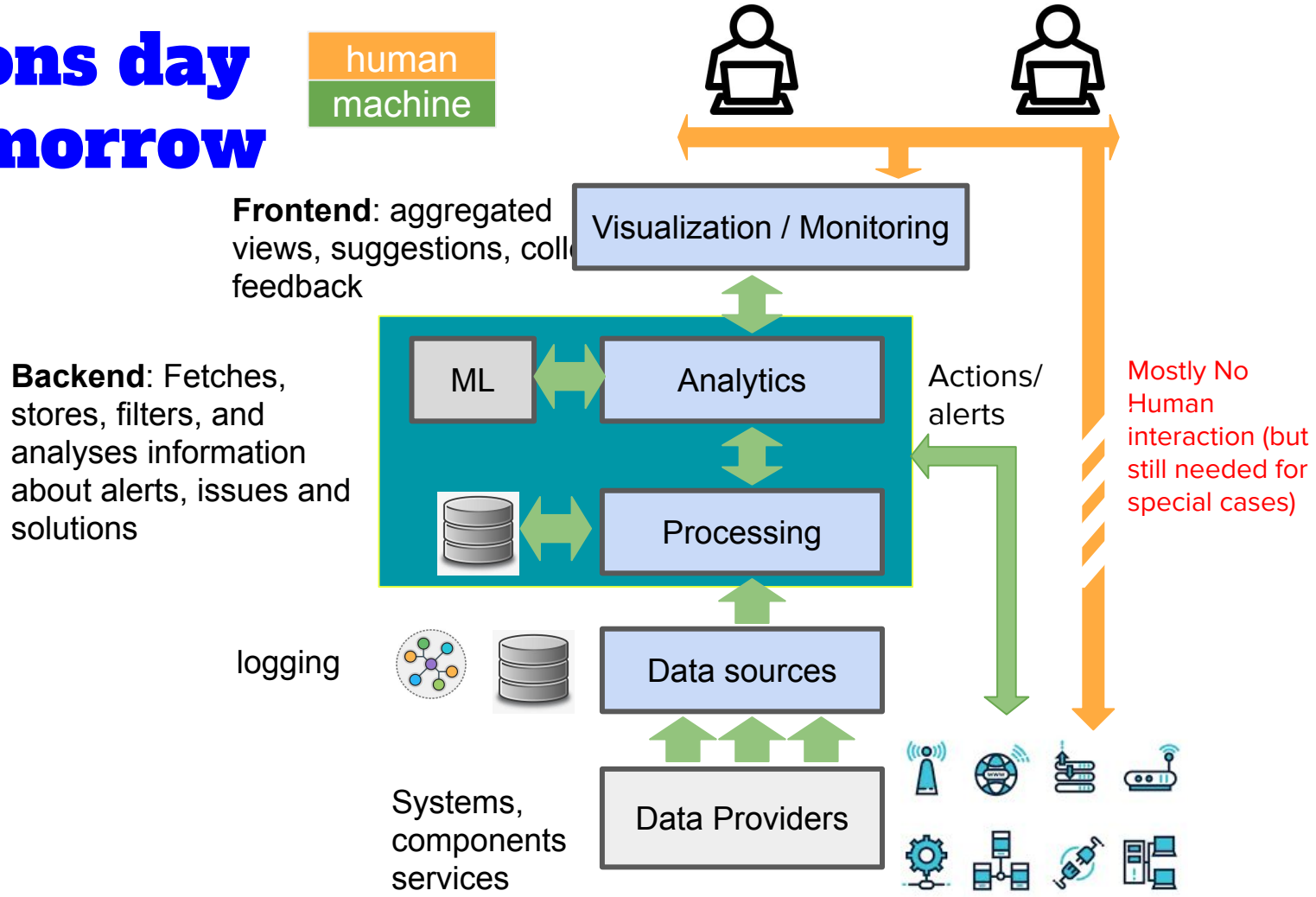
What we are doing to succeed:

- develop tools to automate computing operations exploiting state-of-the-art technology and tools
- run a technical forum, experiment-agnostic to:
  - bring people together
  - discuss ideas, brainstorm together, share experience and code

Operations tomorrow



Operations day after tomorrow



What we are doing

- Bring people onboard: successfully started Operational Intelligent activities
  - Kick-off meeting at HOW19
  - Regular bi-weekly meetings
  - White Paper Draft
  - Experiment-agnostic: the community is the driving force
  - Data-formats, schema, tools/layers
  - For now ATLAS, CMS, LHCb, HammerCloud, CERN Monit, FNAL and more...
- Working in parallel on different areas
  - data standardization: achieve global schema among data (difficult)
  - (More realistic) use data as is and apply analytics/ML methods to understand the data
  - Close the loop and provide feedback to upstream tools (for example, to improve error reporting)
  - Study the literature, in WLCG but not only.
  - Do not rush on implementations, but architect system(s) which can cover multiple experiments needs

… a lot of already ongoing activities

- Infrastructure/tools:
  - CERN, FNAL, INFN-CNAF et al. Analytics Infrastructure - Site resources usage optimization
  - ATLAS ML Chicago platform
  - MLaaS for HEP
  - Visual Analytics
- Projects:
  - Jobs: CMS Operator Console and Alert Triage, Jobs Buster in Atlas
  - Data management:
    - ATLAS, CMS, and LHCb Data Popularity
    - ATLAS Data Transfers alerts
  - Sites:
    - FNAL black node detection
    - INFN Predictive Site Maintenance and Site operations anomaly detection

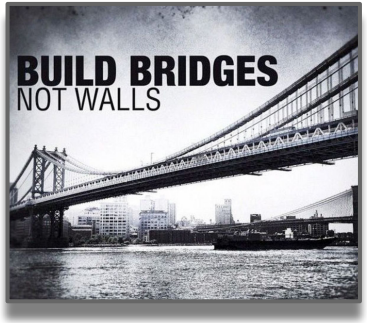
OpInt: forum where people can share and think together how to build sustainable & reusable tools, exploiting already hardly gained experiences

Challenges

- Streamline operations across teams and experiments:
  - One framework to rule them all?
  - Lightweight approach based on messaging services?

- Add **intelligence**:
  - For all supervised approaches: Lack of well annotated data
    - Need experiment-agnostic event annotation tool
    - Currently we only have tickets as a history of things that happened.
    - Not classified in any way that can be used to train any model.
  - Natural language Processing (NLP) for log/text analysis
    - No experts in our field, we are learning!
  - Anomaly detection in time series (Data quality, Network issues, Sites performance)
    - despite the importance, not available off-the-shelf tools
  - For unsupervised: how to validate models without burdening shifters?

Think BIG - Start Simple



- Identify precise projects
  - E.g. Data Transfers performance
  - Experiments are moving between sites several millions files/day, O(50+GB/s) . Many of these are through FTS (for ATLAS, CMS and LHCb), and for ATLAS and CMS there is also the Rucio common ground in addition.
  - This is from where we have decided to start
- Setup a prototype
- Evaluate the impact

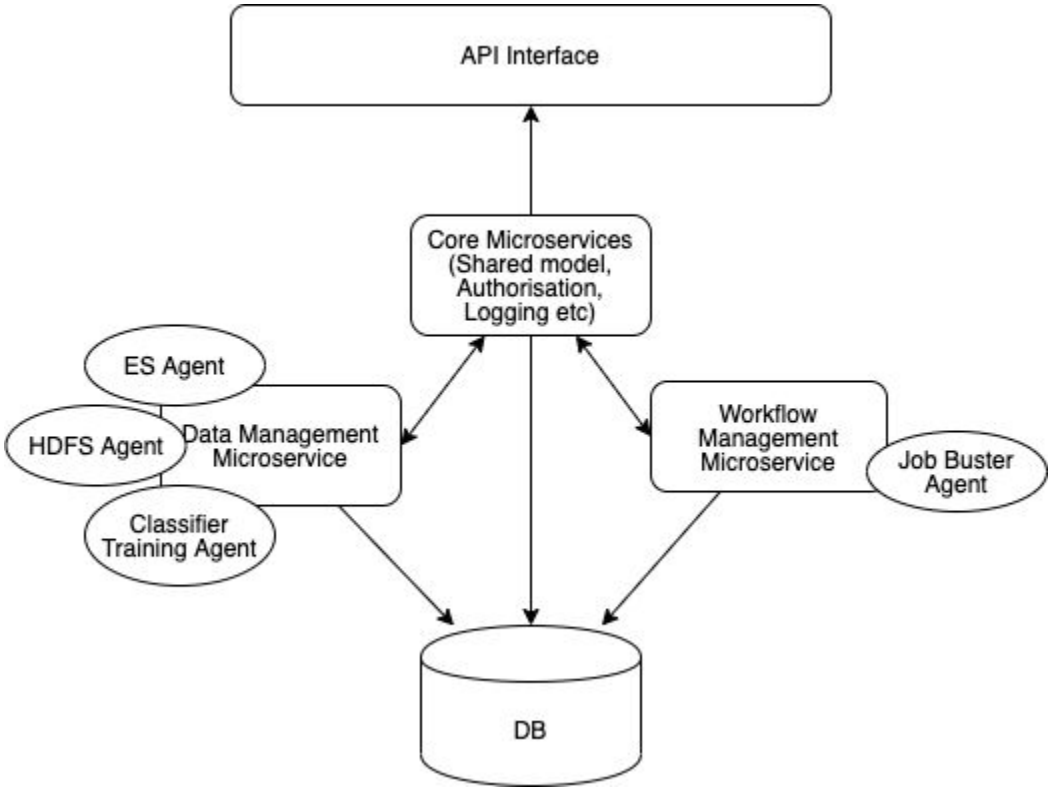
OpInt - The problem

- Shifters are often reporting similar issues and taking "repetitive" actions;
- We need to start dealing with operational issues, e.g. data transfer and workflow management errors more intelligently.
- We need a framework, abstracted from all the services, where we gather the errors, analyse them, offer (suggest) solutions, and get confirmation (or not) on the tool proposed action to take.
    - Shifters/experts feedback is vital to learn, improve and finally manage to build automated tools.
- This is a collaborative cross-experiment effort.
    - We are pretty close today to have a working prototype of OpInt framework which shifters/experts can start using

OpInt - The framework

- We will follow the microservices architecture.
  - Developing a tool for data management errors and incorporating “Jobs buster” project (more later) into the framework
  - Every service (DM, WM etc) will be developed as a different microservice.
  - These parts are loosely coupled. They live under the same project/server/framework but they are as independent as possible.
- We will provide a single deployment and we will (try to) hide the VO specific parts.
- When it’s unavoidable we can incorporate VO specific information either by offering different views/api-presets or by just exposing the information in the existing interfaces.
- Framework will support different information delivery channels to incorporate into existing users environment

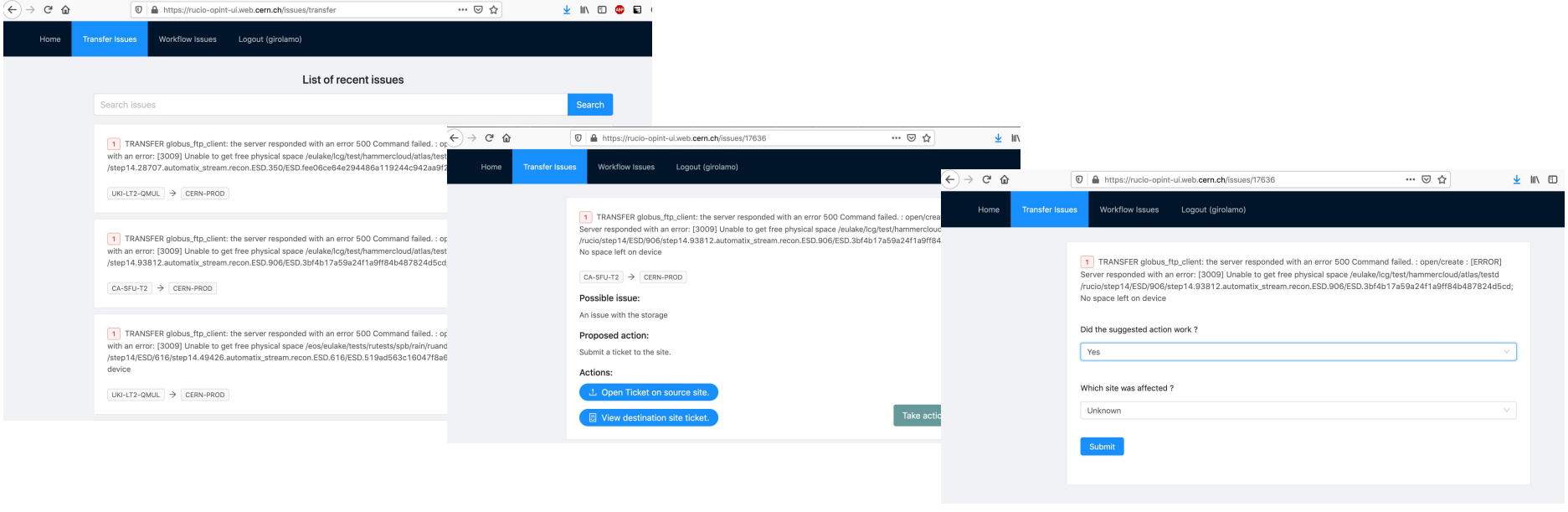
OpInt - The framework



OpInt - The framework

Panos Paparrigopoulos et al.

- The framework is deployed in its current state at the following links:
  - Rucio Frontend: http://rucio-opint-ui.web.cern.ch
  - Backend: http://rucio-opint.web.cern.ch/api/



Rucio OpInt

What do we want to do ?

- New generic tool (expert system) being developed :
  - Will identify recurrent errors in transfers, deletions
  - Suggest to the shifters the actions to take. The shifter will feed back to the service if the suggested action was appropriate, helping the system to become more efficient with time
  - In the medium term, for well identified errors, can even act for the shifters (e.g. submit tickets)
- Main idea is to start simple to address immediate needs from operation team
  - I.e. no fancy feature like ML for the time being
  - Should be operational quickly (not in 2 years)

Cedric Serfon et al.

Workflows

- Several activities ongoing also on the "optimizing compute resources"
- Some examples of WF failures needs to be timely spotted and addressed:
  - We might have very short jobs failing needing to transfer tons of data (not much wallclock lost but a lot of data movements),
  - We might have jobs failing on registration, i.e. a lot of DB operations
  - Jobs failing at their end (lot of wallclock)
  - Tasks failing at particular site should bring to the eye site specific infrastructure issues
  - ...
- Will briefly report about:
  - Jobs buster
  - Real time task monitor with NATS

Job Buster

S. Padolski et al.

System continuously exposes huge number of convoluted failures caused by different reasons. Bigger descriptors shadows smaller ones.

Job Buster

Issues scattered among tens of jobs characteristics and their combination

actualcorecount (16)	1 (45668) 2 (1) 4 (108) 6 (408) 7 (1) 8 (10695) 12 (85) 14 (4) 16 (1) 32 (9) 36 (62) 48 (6) 72 (4) 96 (7) 128 (6) 129 (7)
atlasrelease (100)	Atlas-19.2.3 (391) Atlas-19.2.4 (228) Atlas-19.2.5 (367) Atlas-20.20.14 (221) Atlas-20.20.7 (49) Atlas-20.7.5 (271) Atlas-20.7.6 (1) Atlas-20.7.8 (573) Atlas-20.7.9 (505) Atlas-21.0 (31) Atlas-21.0.100 (3) Atlas-21.0.102 (17) Atlas-21.0.104 (344) Atlas-21.0.14 (3) Atlas-21.0.15 (1100) ... more
attemptnr (83)	1 (25311) 2 (10432) 3 (4883) 4 (3654) 5 (2014) 6 (1818) 7 (1382) 8 (951) 9 (641) 10 (474) 11 (704) 12 (318) 13 (257) 14 (154) 15 (125) 16 (70) ... more
cloud (13)	CA (1637) CERN (1731) DE (7286) ES (766) FR (2289) IT (3159) ND (2190) NL (771) RU (700) TW (46) UK (6368) US (4985) WORLD (25144)
computingsite (262)	AGLT2_UCORE (210) ANALY_AGLT2_SL7-condor (288) ANALY_ARNES (258) ANALY_ARNES_DIRECT (131) ANALY_AUSTRALIA (191) ANALY_BNL_INTEL (15) ANALY_BNL_LONG (612) ANALY_BNL_MCORE (4) ANALY_BNL_SHORT (967) ANALY_CERN (254) ANALY_CERN_HI (197) ANALY_CERN_T0_ART (22) ANALY_CPPM_CL7_ARC (302) ANALY_CPPM_TEST (1) ANALY_CSCS-HPC ... more
corecount (14)	1 (44931) 12 (84) 136 (13) 14 (4) 16 (1) 32 (9) 36 (62) 4 (99) 48 (6) 6 (408) 64 (4) 7 (1) 8 (11443) 96 (7)
durationmin (21)	0-0 (3667) 1-505 (33623) 505-1009 (13304) 1009-1513 (2133) 1513-2017 (755) 2017-2521 (439) 2521-3025 (1865) 3025-3529 (132) 3529-4033 (147) 4033-4537 (574) 4537-5041 (200) 5041-5545 (73) 5545-6049 (24) 6049-6553 (2) 7057-7561 (106) 9577-10081 (28)
eventservice (4)	esmerge (21) eventservice (131) jumbo (13) ordinary (56907)
eventservicestatus (0)	
gshare (15)	Data Derivations (56) Express (483) Group Analysis (1455) Group Exotics (1091) Group Higgs (2638) MC 16 (2727) MC 16 evgen (11946) MC 16 simul (2069) MC Derivations (6916) MC merge (551) Special (24) Test (991) Upgrade (163) User Analysis (25753) Validation (209)
harvesterinstance (9)	CERN-dev (100) CERN_central_0 (9) CERN_central_1 (174) CERN_central_A (11903) CERN_central_ACTA (31169) CERN_central_B (13365) cern_cloud (50) harvester_k8s (6) NERSC_test (13)
homepackage (160)	AnalysisTransforms (150) AnalysisTransforms-AnalysisBase_2019-12-28T0347 (48) AnalysisTransforms-AnalysisBase_2020-01-09T0347 (26) AnalysisTransforms-AnalysisBase_21.2.1 (281) AnalysisTransforms-AnalysisBase_21.2.100 (3070) AnalysisTransforms-AnalysisBase_21.2.101 (321) AnalysisTransforms-AnalysisBase_21.2.102 (90) AnalysisTransforms-AnalysisBase_21.2.103 (1688) AnalysisTransforms-AnalysisBase_21.2.39 (3) AnalysisTransforms-AnalysisBase_21.2.56 (65) AnalysisTransforms-AnalysisBase_21.2.62 (32) AnalysisTransforms-AnalysisBase_21.2.66 (10) AnalysisTransforms-An... more
inputfileproject (15)	data15_13TeV (214) data16_13TeV (984) data17_13TeV (2503) data17_5TeV (50) data18_13TeV (2158) data18_hi (149) group (143) hc_test (39) mc15_13TeV (2350) mc15_14TeV (297) mc16_13TeV (33682) mc16_5TeV (47) panda (577) user (628) valid1 (178)
inputfiletype (63)	AOD (8998) DAOD_BPHY1 (15) DAOD_BPHY9 (7) DAOD_EGAM1 (567) DAOD_EGAM3 (4) DAOD_EGAM5 (1) DAOD_EXOT12 (1) DAOD_EXOT15 (367) DAOD_EXOT19 (30) DAOD_EXOT22 (25) DAOD_EXOT23 (66) DAOD_EXOT27 (323) DAOD_EXOT4 (1437) DAOD_EXOT5 (3034) DAOD_EXOT6 (54) ... more
jeditaskid (4104)	20153074 (2939) 20214763 (1906) 20152539 (1191) 20183153 (1008) 20183205 (904) 20183200 (882) 20214765 (705) 20183149 (676) 20138245 (659) 20232075 (644) 20183151 (546) 20216473 (544) 20166522 (527) 20230498 (521) 20166545 (419) 20214801 (379) 20153669 (348) 20183203 (328) 20194356 (315) 20154503 (290) 20140689 (266) 20198468 (266) 20200099 (266) 20204867 (252) 20156988 (245) 19755875 (242) 20197125 (223) 20137336 (219) 20196023 (217) 20232079 (212) 20233233 (186) 20216499 (184) 20131471 (181) 20041793 (177) 20228898 (173) 20152537 (168) ... more
jobstatus (1)	failed (57072)
jobsubstatus (8)	cancelled (85) fetched (12205) merge_failed (373) prepared (3861) preparing (4414) running (1) staged (5000) submitted (5937)

BigPanDA displays failures

How to identify the lowest common denominator for each failure reason?

Job Buster

Principal reason #1

Principal reason next to #1 (#2)

Principal reason next to #2 (#3)

Principal reason next to #3 (#4)

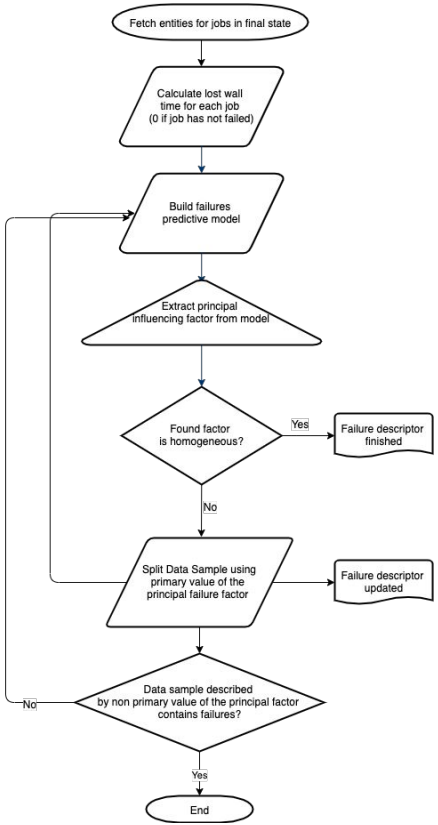
Job Buster

We are scrubbing whole set of problems by principle failure factors

Then we go beyond a principal factor within each subset of jobs and repeat analysis

Divide and Conquer

Job Buster



Step 1: Fetched 20424 failed and 145546 finished jobs

Build failure (wall time loss) predictive model using both successful and failed statistics

Principle factor at this step is Pilot Version.
One value (“Unknown”) is responsible for 150 failures. No successful jobs with this value

Failure spot #1 has found. We can clearly select jobs with homogeneous failure reason

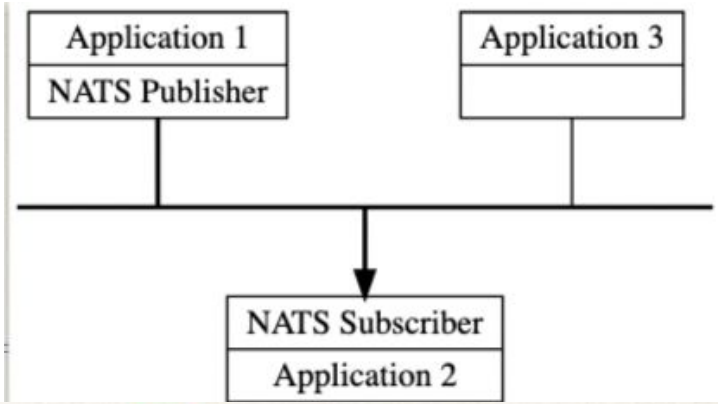
Select jobs with Pilot Version ≠ “Unknown”

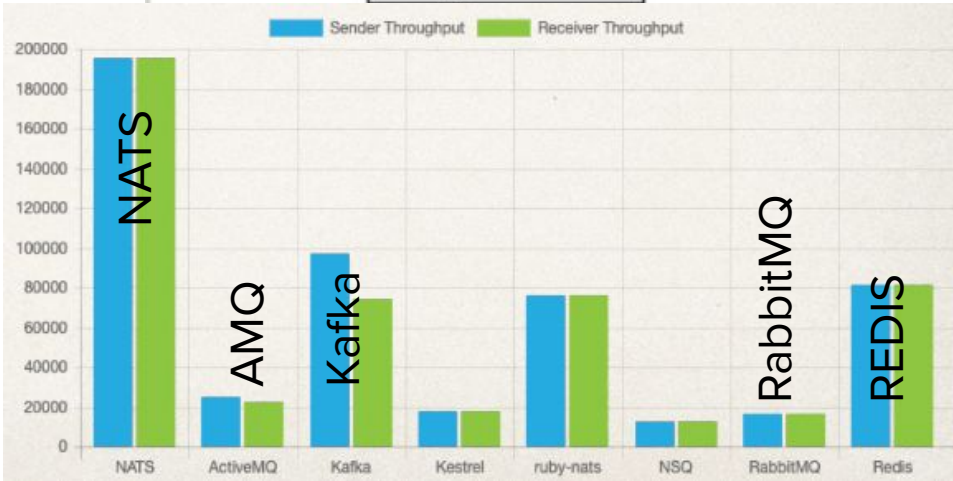
Repeat procedure

Real time monitoring with NATS

V. Kutnetsov et al.

- Must be:
  - easy to use and integrate
  - high throughput
  - minimum maintenance
- **NATS (Neural Autonomic Transport System)** is a simple, secure and high performance open source **messaging** system for cloud native applications
  - Go based: static executable, high throughput, native concurrency, zero configuration
  - support at-least-once, at-most-once deliver, publish-subscribe, streaming operations

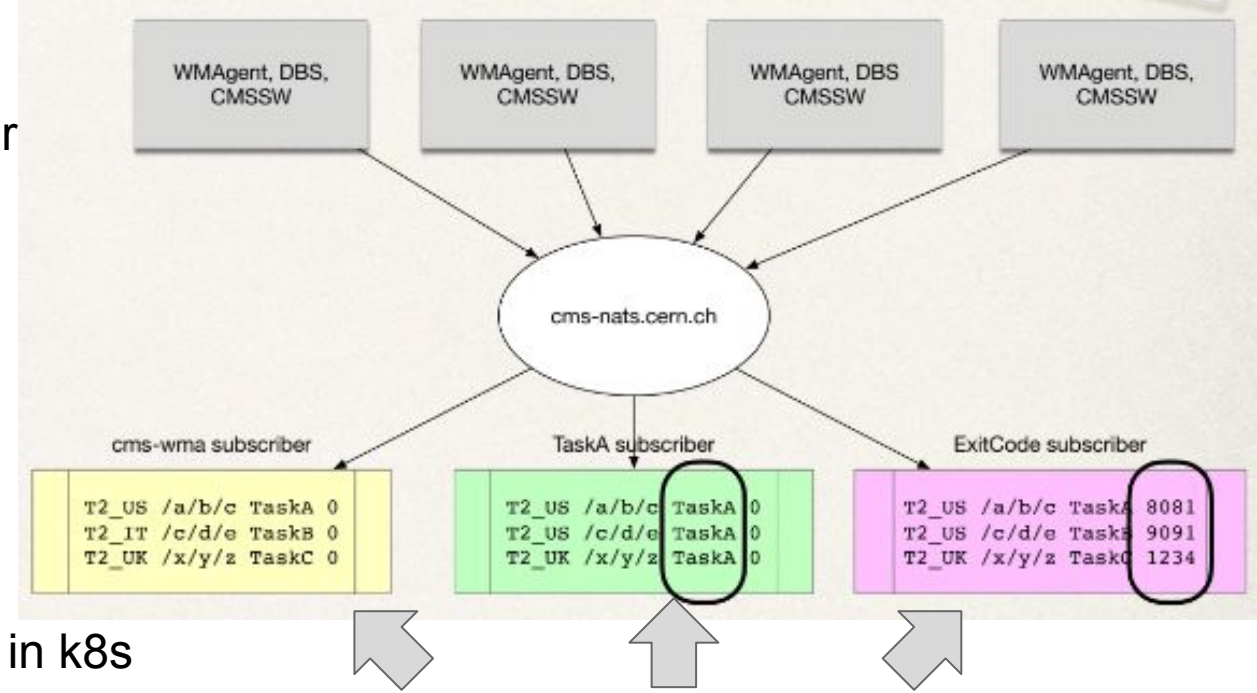




Task monitoring with NATS

V. Kutnetsov et al.

- set of distributed agents which are responsible for data production, reprocessing, job workflows
- need to monitor: campaigns, workflow failures, data accesses, site, ...

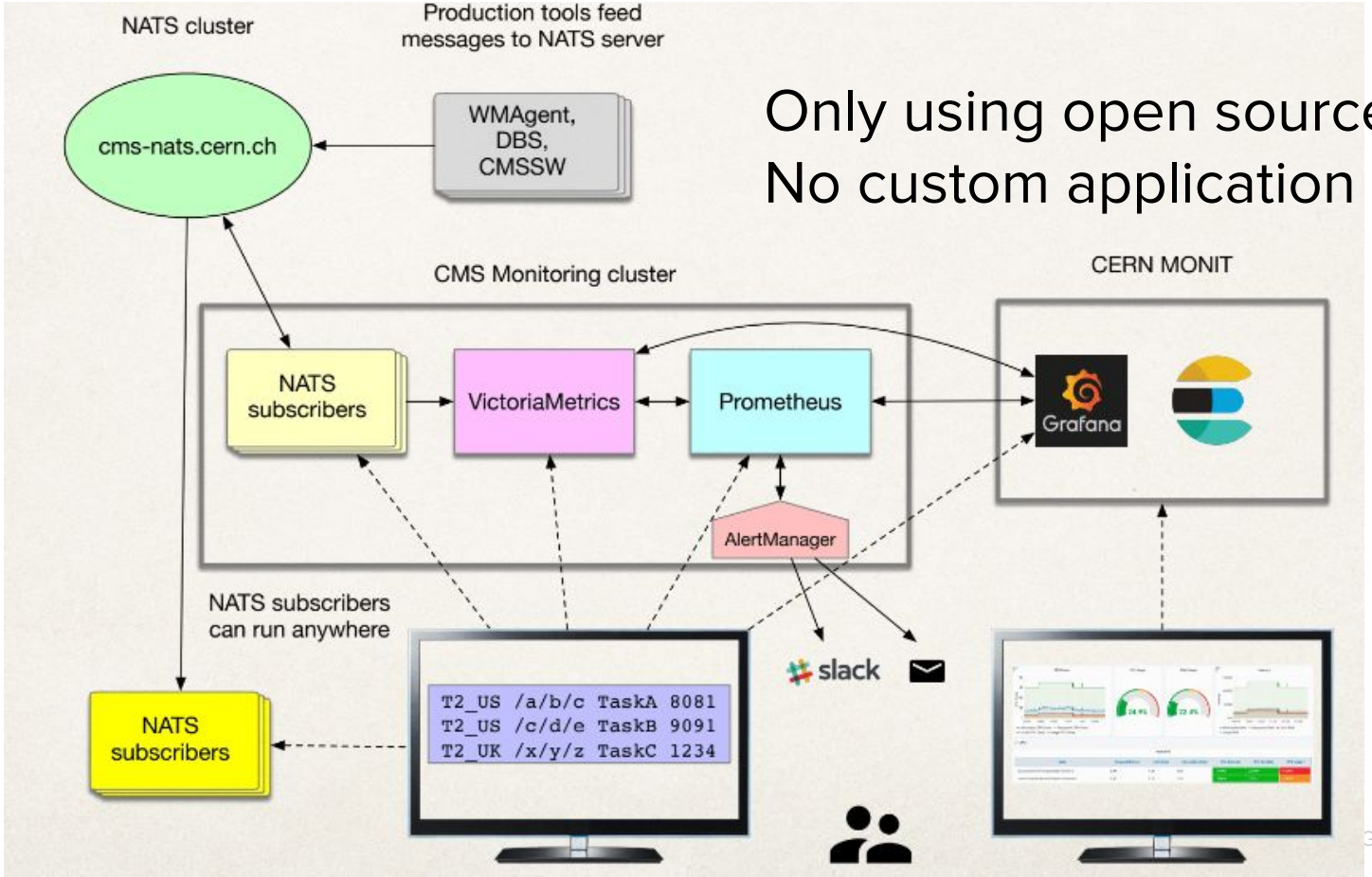


- We setup NATS service in k8s
  - python clients send messages
  - Go-based clients subscribe
  - Persistence with VictoriaMetrics
  - dashboards in Grafana

Different operation teams will subscribe to different messages

Complete monitoring workflow

V. Kutnetsov et al.



Only using open source tools
No custom application

OI: site operations

One focus of OpInt covers the **computing operations at (distributed) computing centers**

Goal: develop (in a collaborative approach) general and shareable frameworks and algorithms to attack a variety of issues preventing an effective use of resources of the computing centers

Expected advantages:

- *Manpower-wise*: reduce latency between spotting problems and taking actions, and increase quality and attention focus of operators' interventions
- *Resource-wise*: address optimisation of resources usage

Soon: diagnostics tools for ML-enforced predictive maintenance at sites

Just one example (similar efforts at other sites):

INFN-CNAF Monitoring and Analytics infrastructure

Focus on log analysis. A Big Data Analytics infrastructure is being deployed at the INFN-CNAF Tier-1.

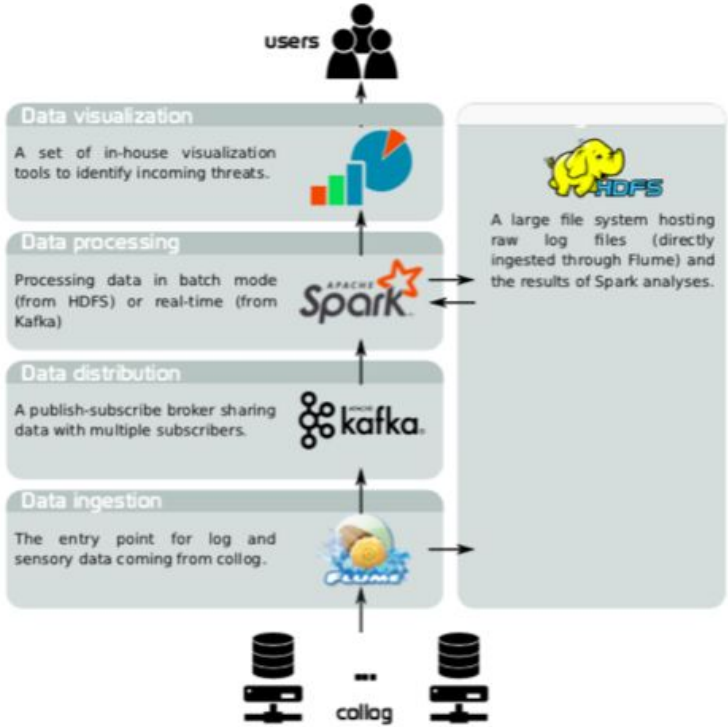
- "*Monitoring and Analytics at INFN Tier-1: the next step*" (*B. Martelli et al*), CHEP 2019 (Adelaide)
- "*A big data infrastructure for predictive maintenance in large data centres*" (*F. Viola et al*), FRUCT25 2019 (Helsinki)

Technology choices: all *open-source software*, *standards* in industry (e.g. Apache) as much as possible.

No CNAF-specific design choice, i.e. work easy to be collaboratively explored by other Tiers: **contact us (below) if interested!**

Work in progress: pool of students (coord. Prof. Bonacorsi, University of Bologna) exploring and comparing a variety of ML-enforced approaches before moving them to production

- more next slide



ML for log processing at a computing center

Small team of students at University of Bologna started to help CNAF personnel through quick “prototype and validate” cycles of a variety of ML-enforced techniques for log text processing and information extraction.

Not-exhaustive list:

- **Supervised learning** of labelled good/bad days in StoRM logs to predict future behaviour (*L.Giommi et al*)
- Collection and harmonization of system logs and **prototypal Analytics services** with the Elastic (ELK) suite (*T. Diotalevi et al*)
- **Clusterization of unstructured log entries** based on measurements of Levenshtein distance (*S. Rossi Tisbeni et al*)
- Unsupervised analysis and exploitation of **volatility as a metric for an anomaly detection** prototype in log-based predictive maintenance (*F. Minarini et al*)
- **Log-agnostic template extraction** on anomaly time windows in logs (*L. Decker De Sousa et al*)

Long-term goal: add predictive capabilities to the site operations (e.g. raise warnings to operators before problems occur), possibly without needing to process the entirety of logs corpus.

Other sites, and/or interested individuals, are welcome to **contact us and join!**

NLP and ML for OpInt 1/2

Maria G and Luca Clissa

One of the crucial tasks of managing distributed, heterogeneous and dynamically changing computing environments is to detect abnormal behaviour, analyse it and resolve in an efficient way.

Log messages are a good data source for abnormal detection:

1. clusterization of log messages
2. extract textual patterns of the clusters
3. detect anomalies

Existing approaches of log parsing and clusterization:

- Frequent Pattern Mining algorithms (SLCT, LogCluster)
- Machine Learning algorithms (LKE, LogMine, LogSig)
- Heuristics methods (Drain, IPLoM, AEL)

Disadvantages:

- not enough accurate and efficient for the analysis of multi-source logs, which are often present in practical applications
- require some parameters to be tuned manually, thus not allowing a fully automated execution

Our approach: based on *Neural Networks (CBOW + Skip-gram)* and *Machine Learning algorithms (K-Means, DBSCAN, Hierarchical)* with the ability to detect clusterization settings automatically.

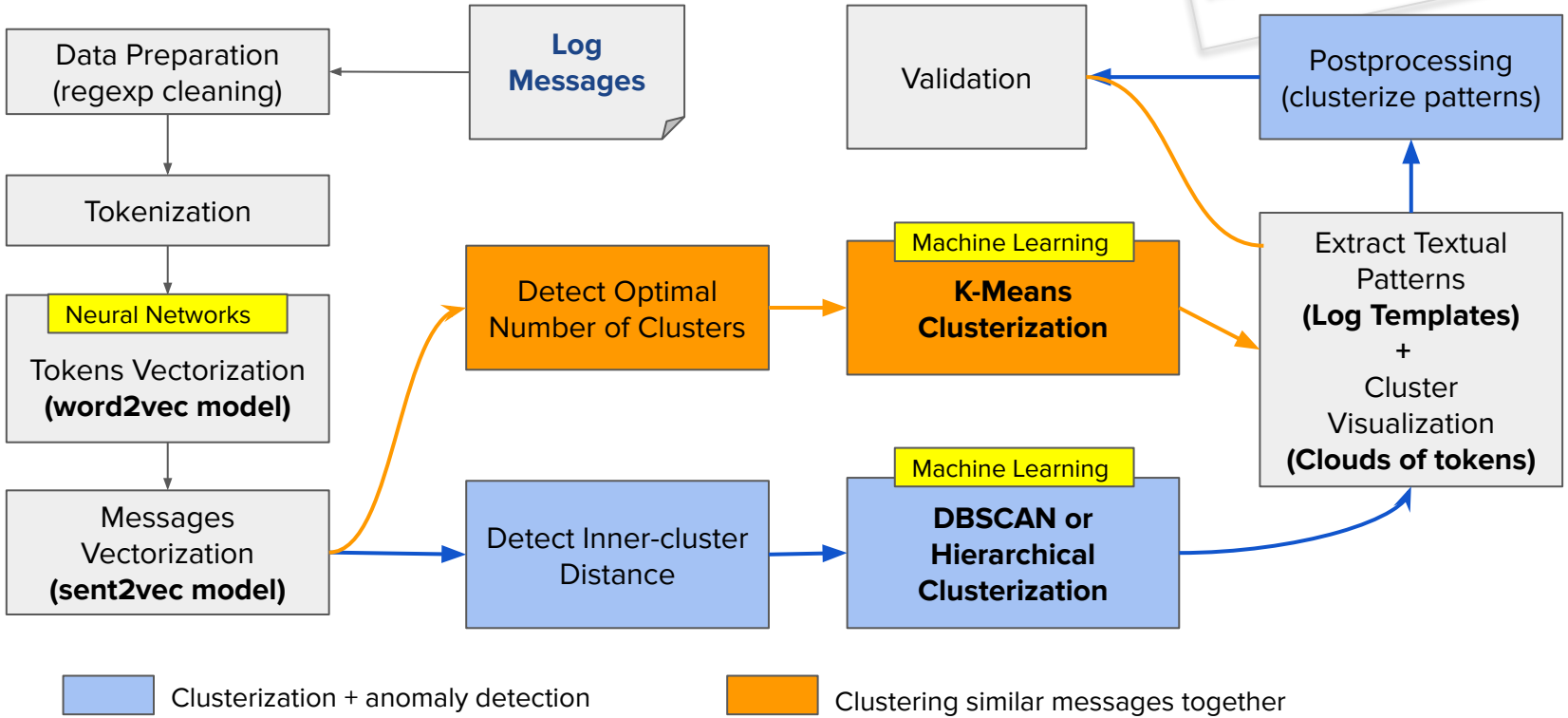
Advantages:

- fully automatic pipeline of clusterization
- high accuracy in the processing of multi-source logs
- trained vocabulary model
- parallelized algorithms allow to improve performance significantly

As a result, clusterization of various log messages will optimize and ease shifters operations.

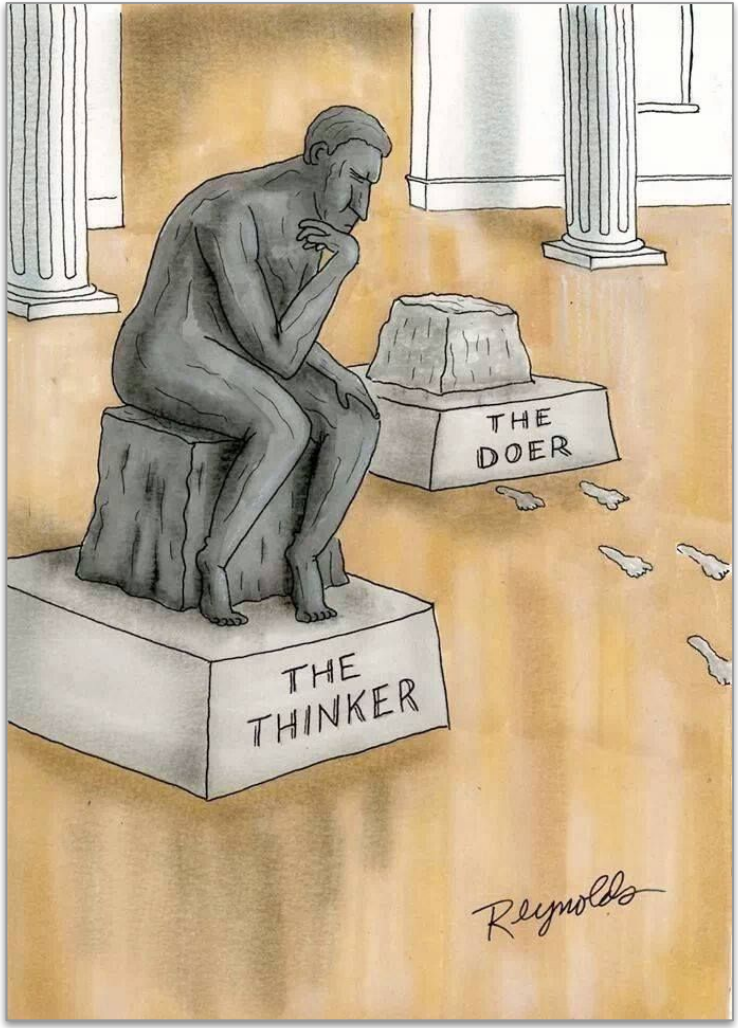
NLP and ML for OpInt 2/2: Logs Clusterization

Maria G and Luca Clissa



Summary and next steps

- Operational Intelligence activities successfully started 9+months ago
- Vibrating community
- Teams of people interested/working on various areas
- White Paper draft (OI -> Documents) available
- Thinking BIG: working in parallel on
  - **global general approach and**
  - **practical reachable milestones with measurable impact**


THE DOER
THE THINKER
Reynolds

Operational Intelligence - details

- Operational Intelligence website
- White Paper Draft
- Github repository: https://github.com/operationalintelligence
- Dockerhub repository: https://hub.docker.com/orgs/operationalintelligence
- E-group for communication: operational-intelligence@cern.ch
- Indico category: https://indico.cern.ch/category/11205/