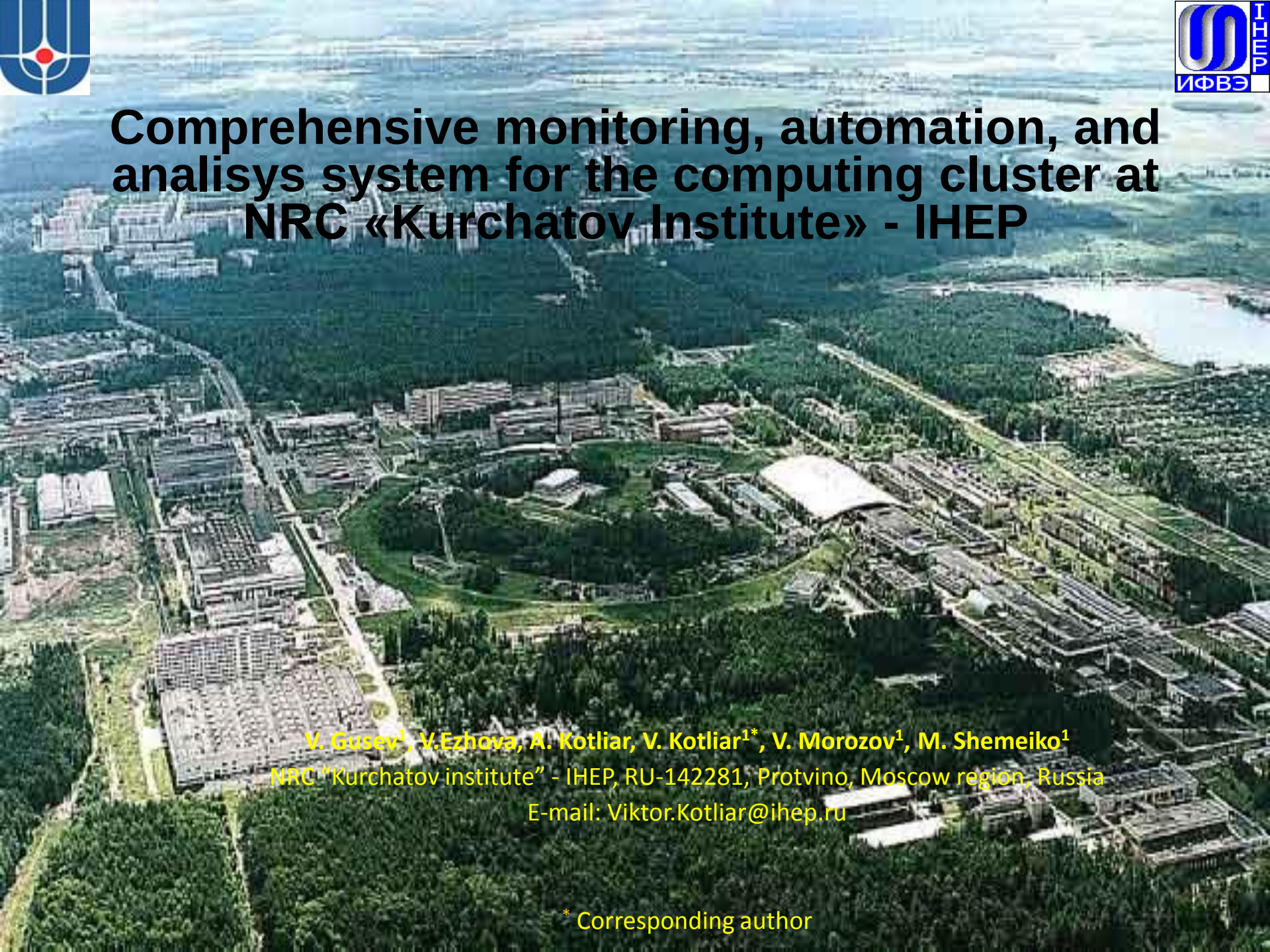
Comprehensive monitoring, automation, and analisys system for the computing cluster at NRC «Kurchatov Institute» - IHEP



V. Gusev[1], V.Ezhova, A. Kotliar, V. Kotliar[1*], V. Morozov[1], M. Shemeiko[1]

NRC "Kurchatov institute" - IHEP, RU-142281, Protvino, Moscow region, Russia

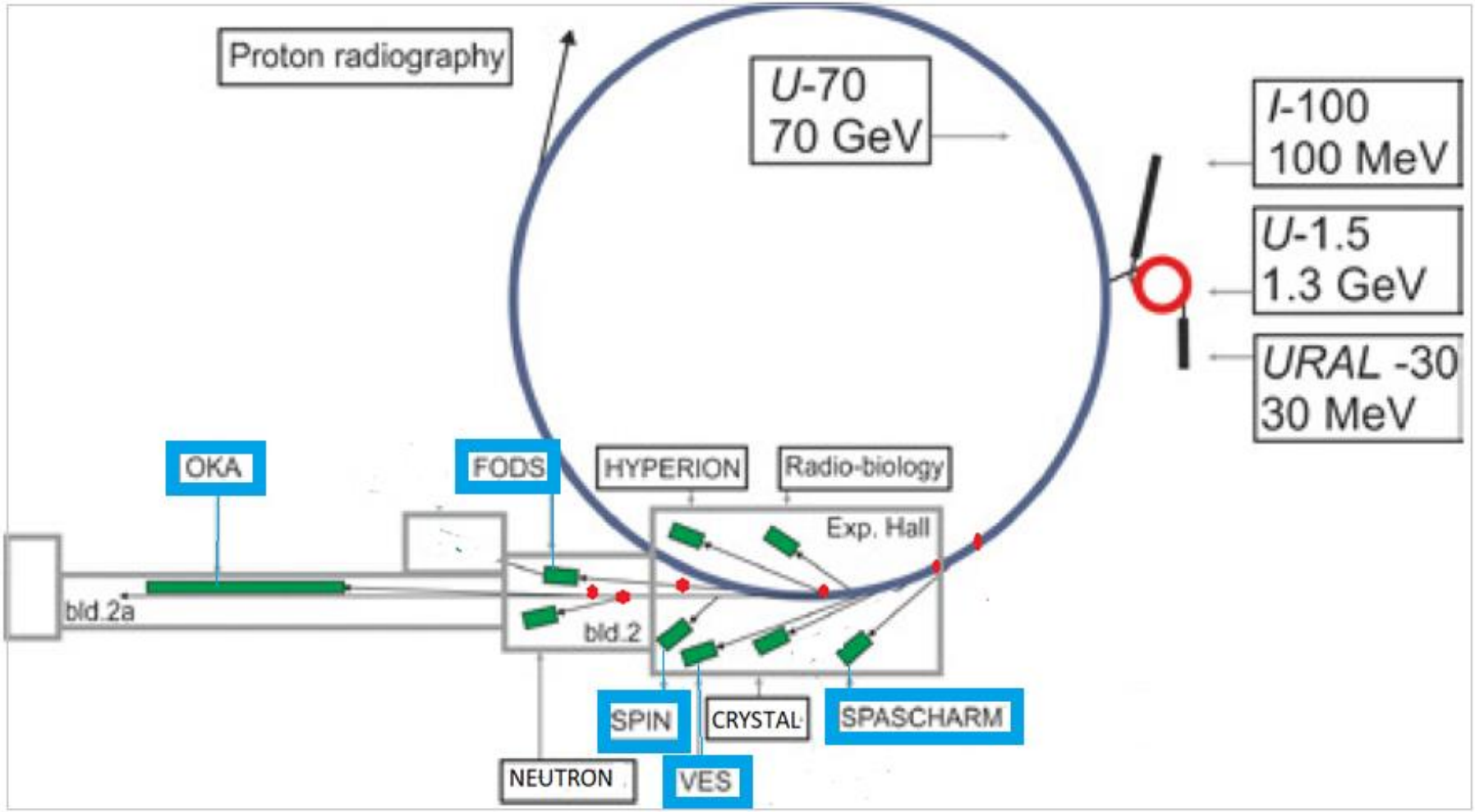
E-mail: Viktor.Kotliar@ihep.ru

[*] Corresponding author

Outline

- Introduction for U-70 infrastructure (physics and compute)
- Cluster infrastructure overview
- Cluster management system
- Monitoring system architecture
- Engineering infrastructure monitoring and monitoring for computation parameters
- Anomaly detection systems
- Key points, future, conclusion

Introduction physics

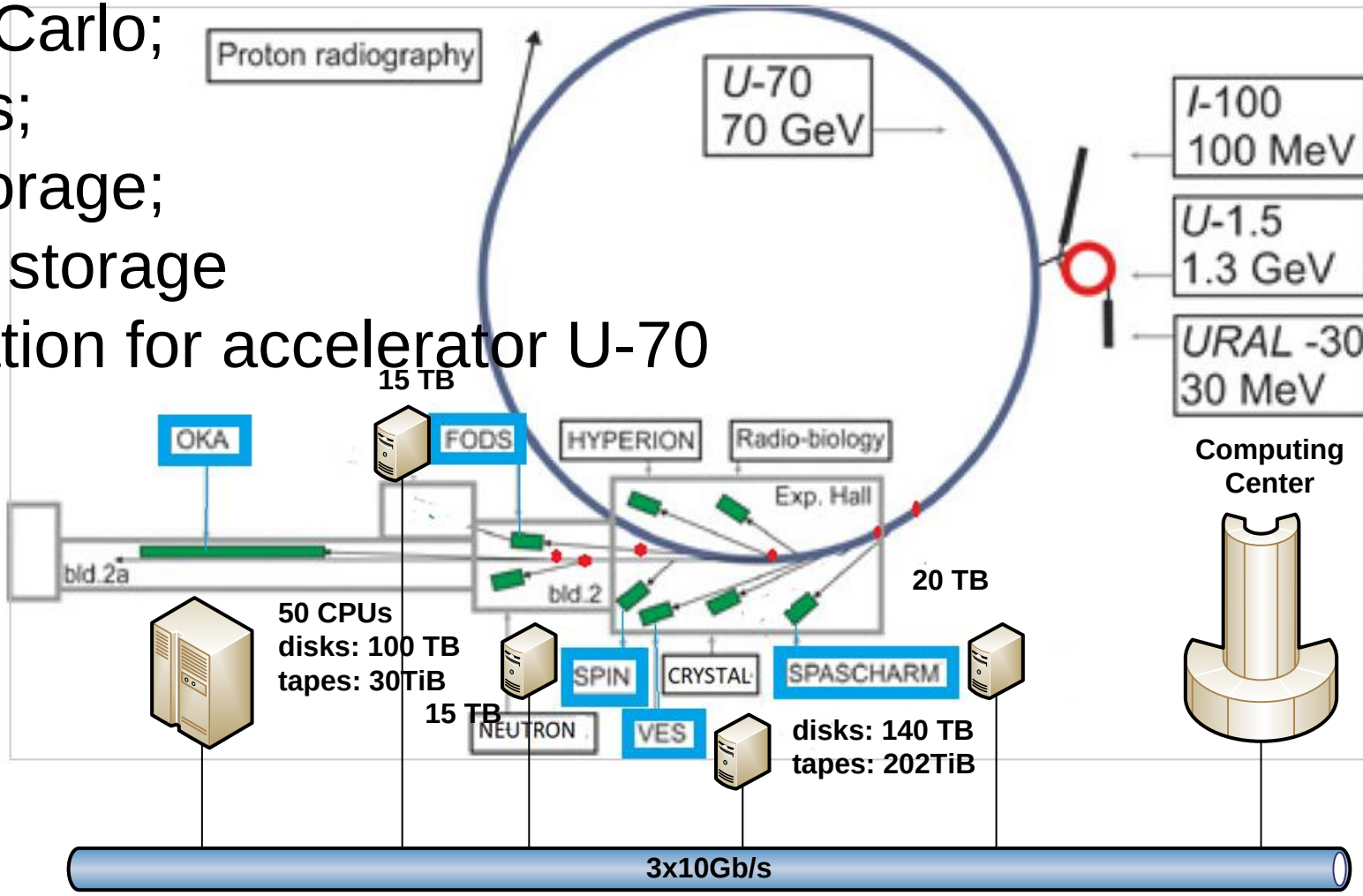


Introduction computing

Support high-energy physics experiments

- Reconstruction;
- Monte Carlo;
- Analysis;
- Data storage;
- Archive storage

24/7 operation for accelerator U-70



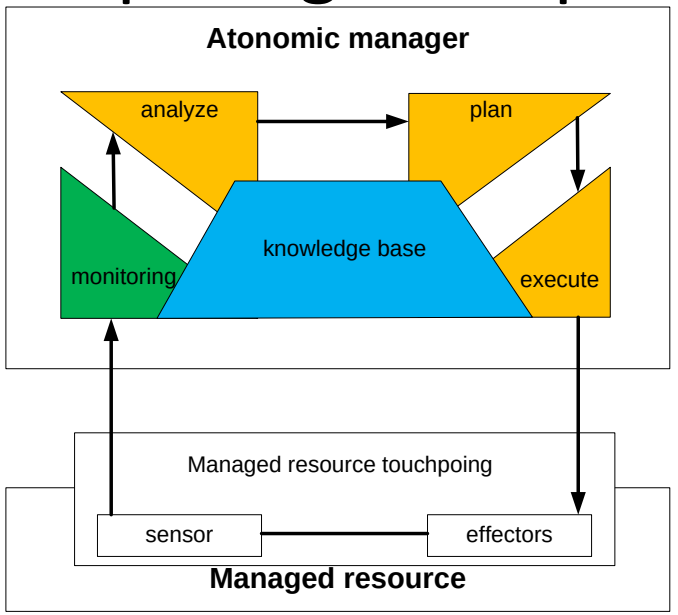
Cluster Infrastructure Overview

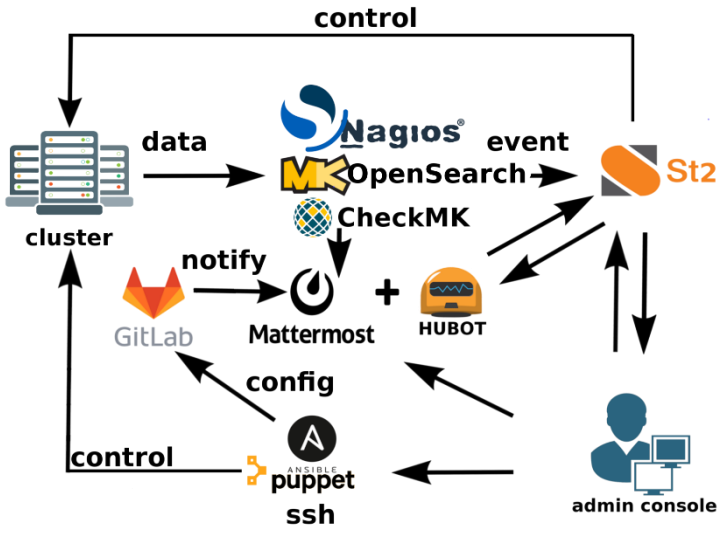
- Engineering resources:
  - Power: 2 main UPS (64 kW + 96 kW), 30+ smaller UPS.
  - 30 racks 7-14 кW per rack;
  - Cooling: 6 precision air conditioners.
- Computing resources:
  - 150 servers, 3,000 CPUs;
  - 2.5 PB storage (50 servers);
  - Networking: 1,000+ 1 Gb/s connections, dual 10 Gb/s internet links.
- Software:
  - Queue-based task isolation for multi-experiment resource sharing.
  - Tiered storage (high-performance, big-data streaming, tape archives).

Cluster Management System Core Concepts with Autonomous Computing Principles

Four aspects of self-management

1. Self-configuration
   - Configure themselves automatically
   - High-level policies (what is desired, not how)
2. Self-optimization
   - Hundreds of tunable parameters
   - Continually seek ways to improve their operation
3. Self-healing
   - Analyze information from log files and monitors
4. Self-protection
   - Malicious attacks
   - Cascading failures





Control Loop Workflow:

- **Monitor → Analyze → Plan → Execute** (with shared knowledge base).

Monitoring System Architecture

web-Applications for analysis:
- Monitoring;
- Applications for “unknown” problems

Alarm systems

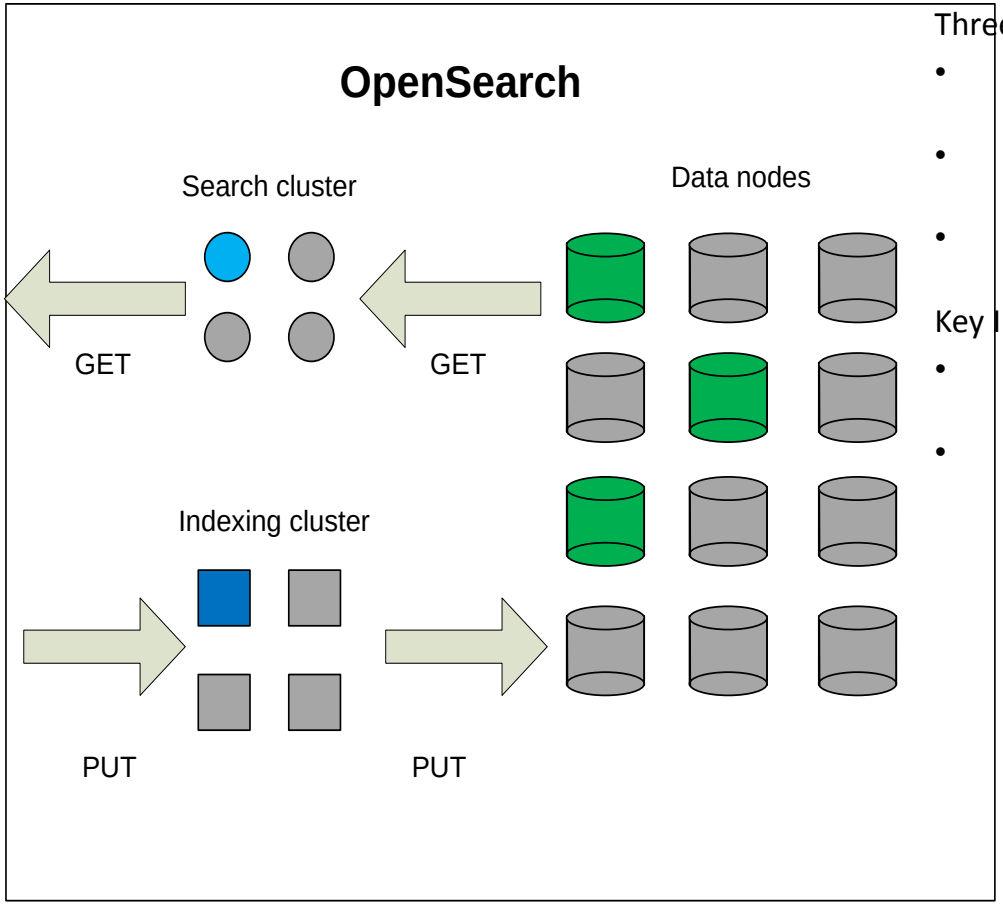
REST API

Application SDK

Collectors:
- Environment data;
- Infrastructure data (symmetra, pdu, cooling system, ups);
- cluster data;gpu usage, xen usage;
- Data from external monitoring systems.

REST API

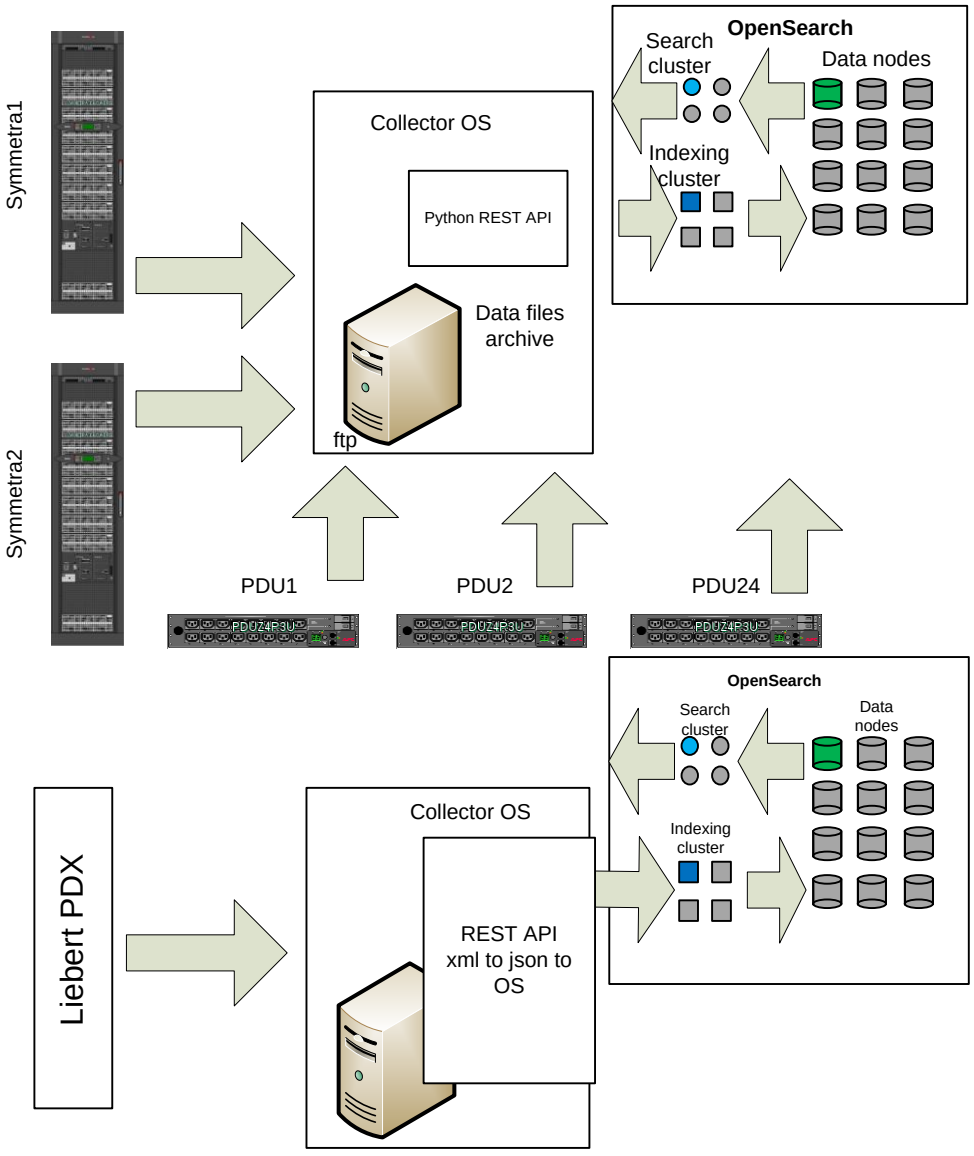
Applications SDK



Three-Tier Design:
- Indexing Cluster: Data collection and aggregation (REST API).
- Data Nodes: Scalable storage (horizontal scaling).
- Search Cluster: Real-time querying (OpenSearch Dashboards).

Key Innovation:
- Modular “collectors” for new parameters without core changes.
- Supports Python SDK, custom scripts, and third-party tools.

Parameter Collections in Practice

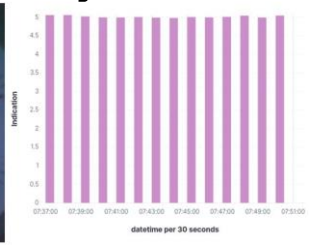


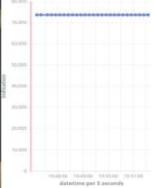
Engineering Equipment:

- **UPS/PDU:** Data via FTP → Python processing → REST API.
- **Cooling (Libert PDX):** Direct sensor access → dashboards for engineers.
- **Sensors:** Temperature, humidity, water pressure/consumption (CV for analog-to-digital conversion).

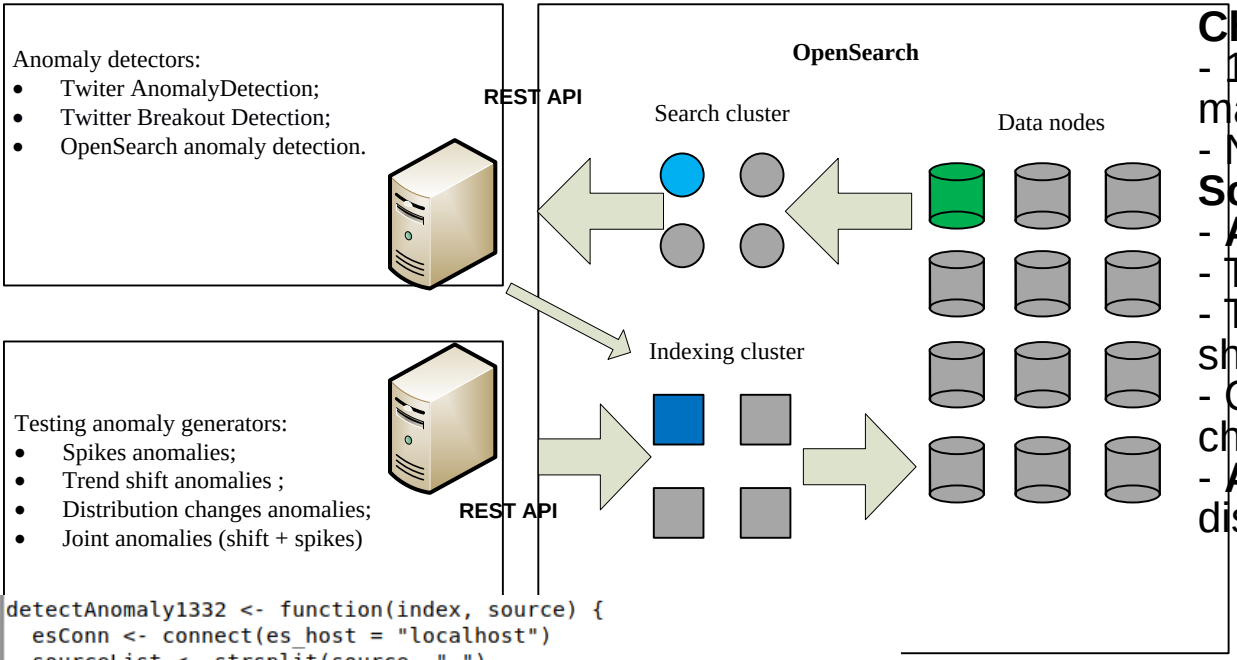
Computational Parameters:

- Custom collectors: Virtualization, storage disks, GPUs, cluster tasks efficiency.





Anomaly detection system



Challenges:
- 18,000+ metrics → impossible for manual tracking.
- Need for proactive failure prediction.

Solution:
- **Algorithms:**
- Twitter Anomaly Detection (spikes).
- Twitter Breakout Detection (trend shifts).
- OpenSearch ML (distribution changes).
- **Anomaly Types:** Outliers, shifts, distribution changes, joint anomalies.

```
detectAnomaly1332 <- function(index, source) {
  esConn <- connect(es_host = "localhost")
  sourceList <- strsplit(source, ",")
  if (length(sourceList[[1]]) != 2) {
    stop("Errors in source list", call.=FALSE)
  }
  date <- sourceList[[1]][[2]]
  param <- sourceList[[1]][[1]]

  typeData <- Search(esConn, index = index,  size = 10000,
    sort = paste(date, ":desc", sep=""),
    source = source)$hits$hits
  dateParam <- unique(sapply(typeData, '[[', 4),MARGIN = 2)
  tsData  <- as.POSIXct(strptime(dateParam[date,],
    "%Y-%m-%dT%H:%M:%S"))
  attributes(tsData)$tzone <- "UTC"
  tsFrame <- data.frame(tsData,unlist(dateParam[param,]))
  tsFrame <- tsFrame[nrow(tsFrame):1,]
  anomaly <- AnomalyDetectionTs(tsFrame, max_anoms=0.01, alpha=0.05,
    direction='both', plot=FALSE, only_last='hr')
  return (anomaly[["anoms"]])
}
```

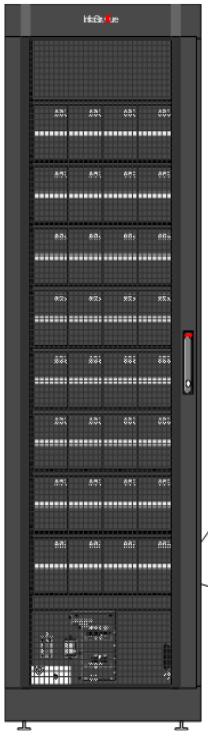
Power failure: symmetra1.ipmi.ihep.su: On battery power in response to an input power problem. 0x0109

Anomaly detected: Symmetra1 problem 2025-06-17T18:25:24+03:00 0.2

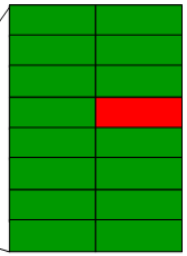
Case Study: Success Stories

Detect SYMMETRA battery problems:

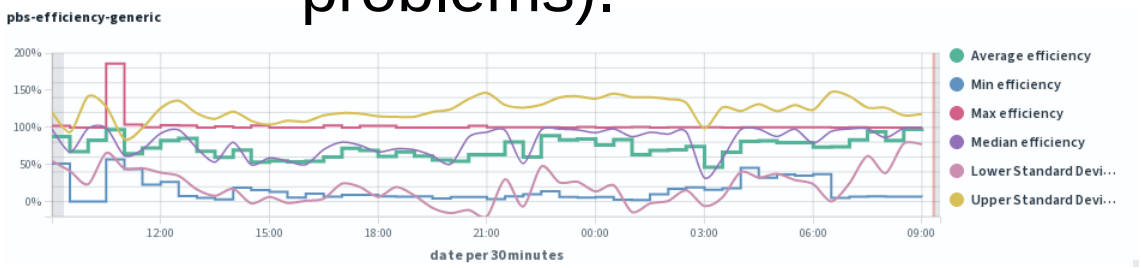
Mar 12 07:15:51 192.168.66.10 UPS: At least one faulty battery exists. 0x0119
Mar 12 07:15:51 192.168.66.10 UPS: A battery fault exists. 0x0207
Mar 12 07:15:57 192.168.66.10 UPS: The internal battery temperature exceeds the critical threshold. 0x012C
Mar 12 07:17:50 192.168.66.10 UPS: The internal battery temperature no longer exceeds the critical threshold. 0x012D



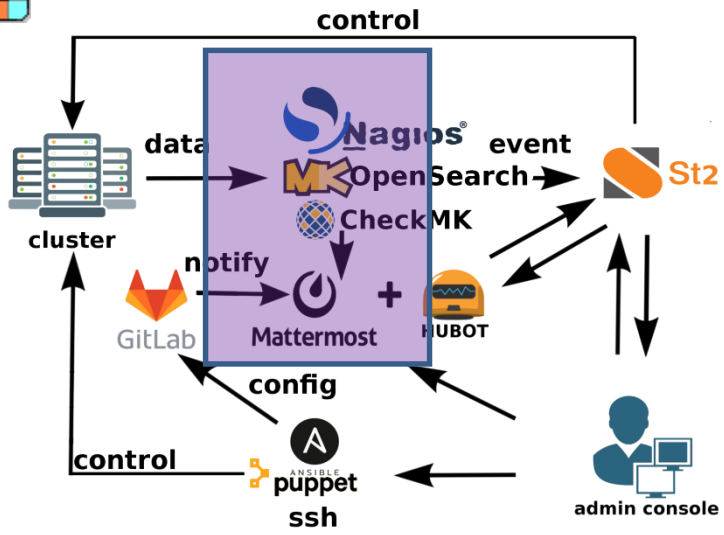
- Ibat >0
- Check temperature for battery blocks and find a bad one
- find\replace a bad battery in the block



- **UPS Battery Degradation:** Detected abnormal charging currents → preventive maintenance.
- **Task Efficiency Optimization:**
  – Dashboard tracks CPU time vs. wall time
  – Improved cluster scheduling policies (disk I/O).
- **Future:** Machine learning integration for predictive? analytics (for example PS problems).

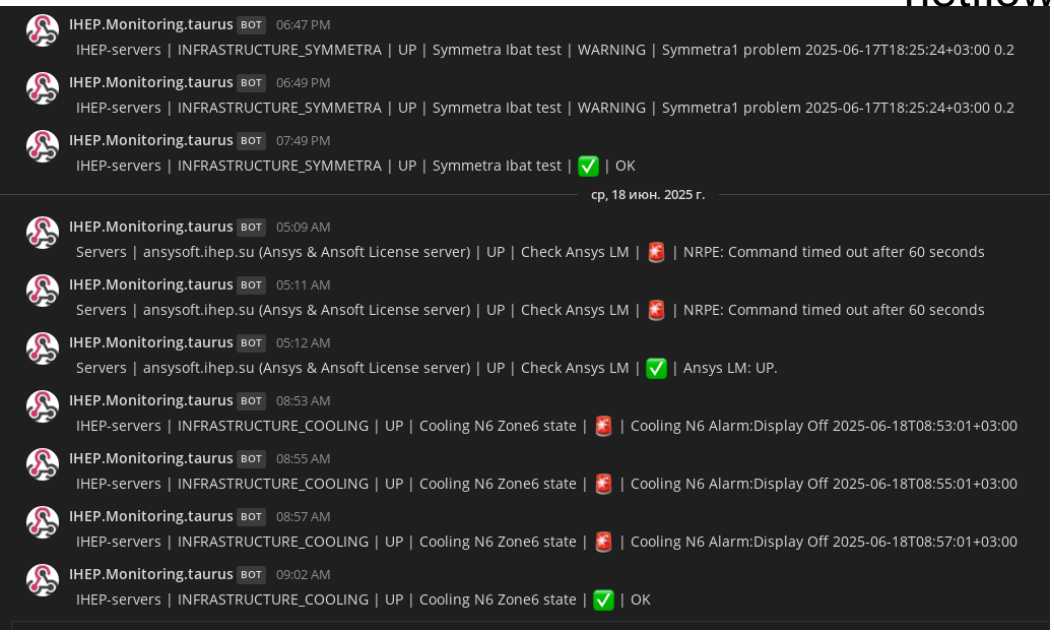


Computation parameters monitoring



Service oriented monitoring
Tools:

- Nagios (modified version, historically) + Checkmk (18k+ metrics 250hosts);
- Splunk(dashboards)/Rsyslog (logs);
- Collectl (real-time).
- Cacti for network monitoring (sflow and netflow collectors for traffic analysis)



- Recently moved to the Mattermost(self hosted) chat notifications from all system;
- Sysadmin on duty just check the status of the system through usual monitoring and then check changes for the system (notifications)
- No need for monitoring walls!

Conclusion

Achievements:

– Unified, scalable system using open-source tools (OpenSearch, Python, R).

– Anomaly detection reduced downtime and improved resource efficiency.

– Chat-ops with monitoring allowed to easily maintain a very complex distributed monitoring system

- Dashboard on-demand and complicated search requests for monitoring data is really a feature “must have” in distributed environments

Impact: Reliable support for U-70 accelerator high energy physics experiments in 24x7x365 mode.

Key points and Future

- JSON format for every monitoring data very useful and allow to intergrade anything (on the size of our cluster infrastructure)
- We are not use real-time monitoring for parameters – standard reaction time 5 min (such setup also used for service monitoring to avoid service flaps or glitches)
- For future we plan to try integrate LLM for knowledge base part, for anomaly detection, for monitoring analysis (chat nature of Mattermost in core is the bonus here)
- Analysis for monitoring will need CPU power (clusters? We still in the beginning)

Future almost here

DeepSeek already see anomalies in data for APC symmetra

1. **Data Characteristics:**
   - 2,417 data points spanning from **2025-06-13 03:50 to 2025-06-17 19:25 (UTC+3)**.
   - `Ibat` values are predominantly negative (indicating discharge), typically ranging between **-0.6 and 0**.
   - Most measurements are at **5-minute intervals**, with occasional duplicates (same timestamp and value).
2. **Anomaly Detection Criteria:**
   - **Positive `Ibat`**: Indicates unexpected charging behavior (should be negative during discharge).
   - **Rapid Current Changes**: Absolute difference >0.35 between consecutive measurements (≤10 min apart), indicating sudden load shifts.
   - **Extreme Values**: `Ibat < -0.7` (not observed in this dataset).

Identified Anomalies:

7 anomalous data points were detected:

Timestamp (UTC+3)	Ibat	Anomaly Type	Description
2025-06-17T18:25:24+03:00	0.2	Positive current	Unexpected positive value (charging)
2025-06-17T18:45:24+03:00	-0.6	Rapid change (-0.2 → -0.6)	Sudden discharge spike (Δ=0.4)
2025-06-17T03:10:26+03:00	-0.1	Rapid change (-0.6 → -0.1)	Abnormal recovery (Δ=0.5)
2025-06-17T03:15:26+03:00	-0.6	Rapid change (-0.1 → -0.6)	Sudden discharge spike (Δ=0.5)

Notable Patterns:

- **Positive Current Anomaly** (2025-06-17 18:25): The **only positive `Ibat` value (0.2)** in the dataset. This is abnormal as the system should discharge (negative current) under normal operation.
- **High-Discharge Spikes**: All other anomalies involve abrupt drops to **-0.6** (e.g., from -0.2 in 5 minutes). These suggest sudden load surges or system disturbances.
- **Temporal Clustering**: Anomalies occurred on **2025-06-17** (5 events) and **2025-06-15/16** (2 events), indicating potential operational issues during these periods.

Future almost here

DeepSeek already can help in problem solving

Critical Alert Analysis & Response Guide

GPU Alert:

`ALDAN3-gpu | GPU0 | 😱 TeslaV100S-PCIE-32GB power utilization at 118% of 250W (295W)`

Immediate Actions

1. **Verify Current State**

```
# Check real-time GPU metrics
nvidia-smi --query-gpu=index,name,temperature.gpu,power.draw,power.limit,clocks.sm,utilization.gpu
```

Look for:

- Power draw consistency (fluctuating vs. sustained 295W)
- GPU utilization >90%
- Temperature >85°C

2. **Identify Offending Process**

```
# List processes using GPU0
nvidia-smi --id=0 --query-compute-apps=pid,process_name,used_memory,gpu_util --format=csv
```

3. **Check System Logs**

```
# Search for GPU/driver errors
dmesg | grep -iE "nvidia|gpu|power|thermal"
journalctl -b -0 | grep -i nvidia
```

Future almost here

DeepSeek already can create site specific knowledge base (based on server history)

Procedure: Failed Disk Replacement in Linux Software RAID

Pre-requisites: - New disk (`/dev/sdb` in this example) physically installed - Healthy reference disk with valid partition table (e.g., `/dev/sda`) - Root/sudo access

---

1. Verify Disk Failure and RAID Status

```
# Check RAID status
cat /proc/mdstat

# Inspect disk health (repeat for all disks)
smartctl -a /dev/sda
smartctl -a /dev/sdb
dmesg | grep -i error  # Check kernel logs for disk errors
```

---

2. Replicate Partition Table

```
# Copy partition scheme from healthy disk to new disk
sfdisk -d /dev/sda | sfdisk /dev/sdb

# Verify partition replication
fdisk -l /dev/sda /dev/sdb  # Compare outputs
```

---

3. Add Partitions to RAID Arrays

```
# Add partitions to respective RAID devices
mdadm --add /dev/md0 /dev/sdb1
mdadm --add /dev/md1 /dev/sdb2
mdadm --add /dev/md2 /dev/sdb3
mdadm --add /dev/md4 /dev/sdb6
mdadm --add /dev/md5 /dev/sdb7
mdadm --add /dev/md6 /dev/sdb8
mdadm --add /dev/md7 /dev/sdb9

# Monitor rebuild progress
```


Thank you!
Any questions?