Automatic recovery system in the Belle II operation

Takuto KUNIGO

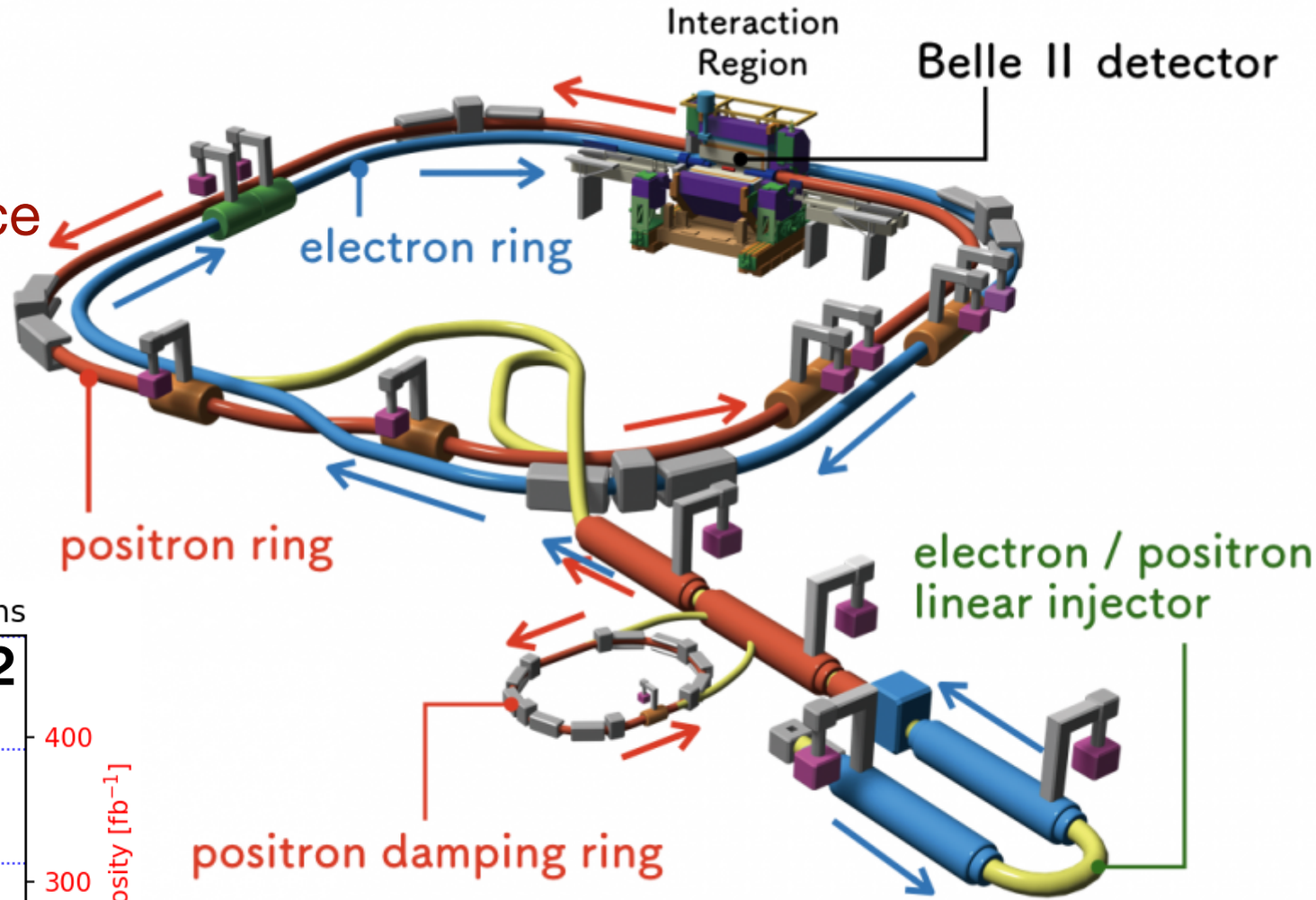
Ryosuke Itoh, Mikihiko Nakao,
Soh Yamagata Suzuki, Satoru Yamada,
Qi-Dong Zhou, Seokhee Park, Dmytro Levit

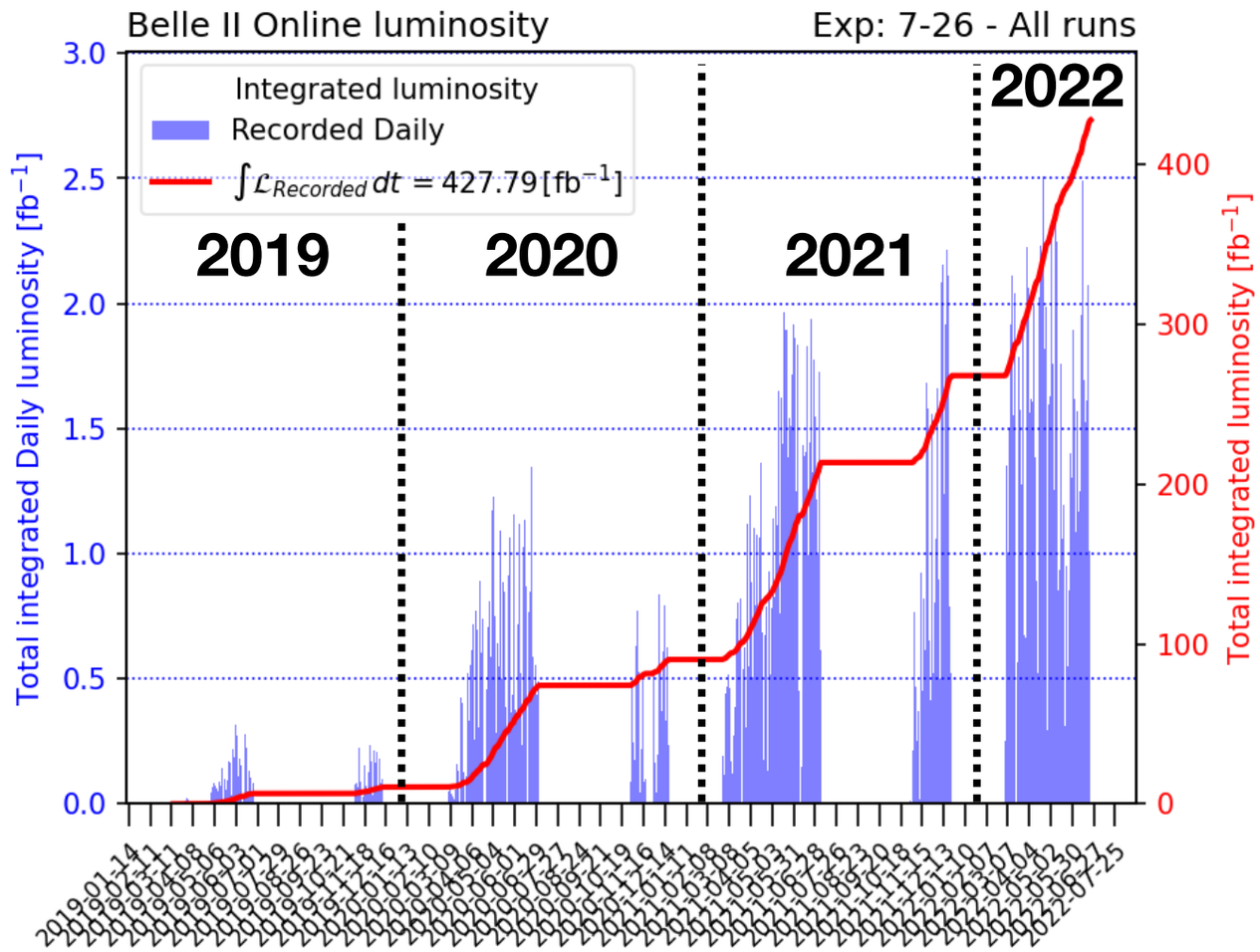
August 2nd, 2022 (in JST)
23rd Virtual IEEE Real Time Conference

SuperKEKB accelerator

4 GeV e^+ 7 GeV e^- collider

- Successor to KEKB (Belle, 1 ab^{-1})
- Long-shutdown 1 (LS1) started since June 2022 to achieve luminosity $O(10^{35}\ cm^{-2}s^{-1})$





World record achieved before LS1
$L = 4.7 \times 10^{34}\ cm^{-2}s^{-1}$

Goal: 50 ab^{-1} (= Belle x 50)

Belle II detector

Search for new physics beyond SM via high precision measurement with **high statistics samples of B/D/tau decays**

Inner

Tracking

Pixel Detector（PXD）

Silicon Vertex Detector（SVD）

Vertex detector

Central Drift Chamber（CDC）

PID

TOP counter（TOP）

Aerogel RICH counter（ARICH）

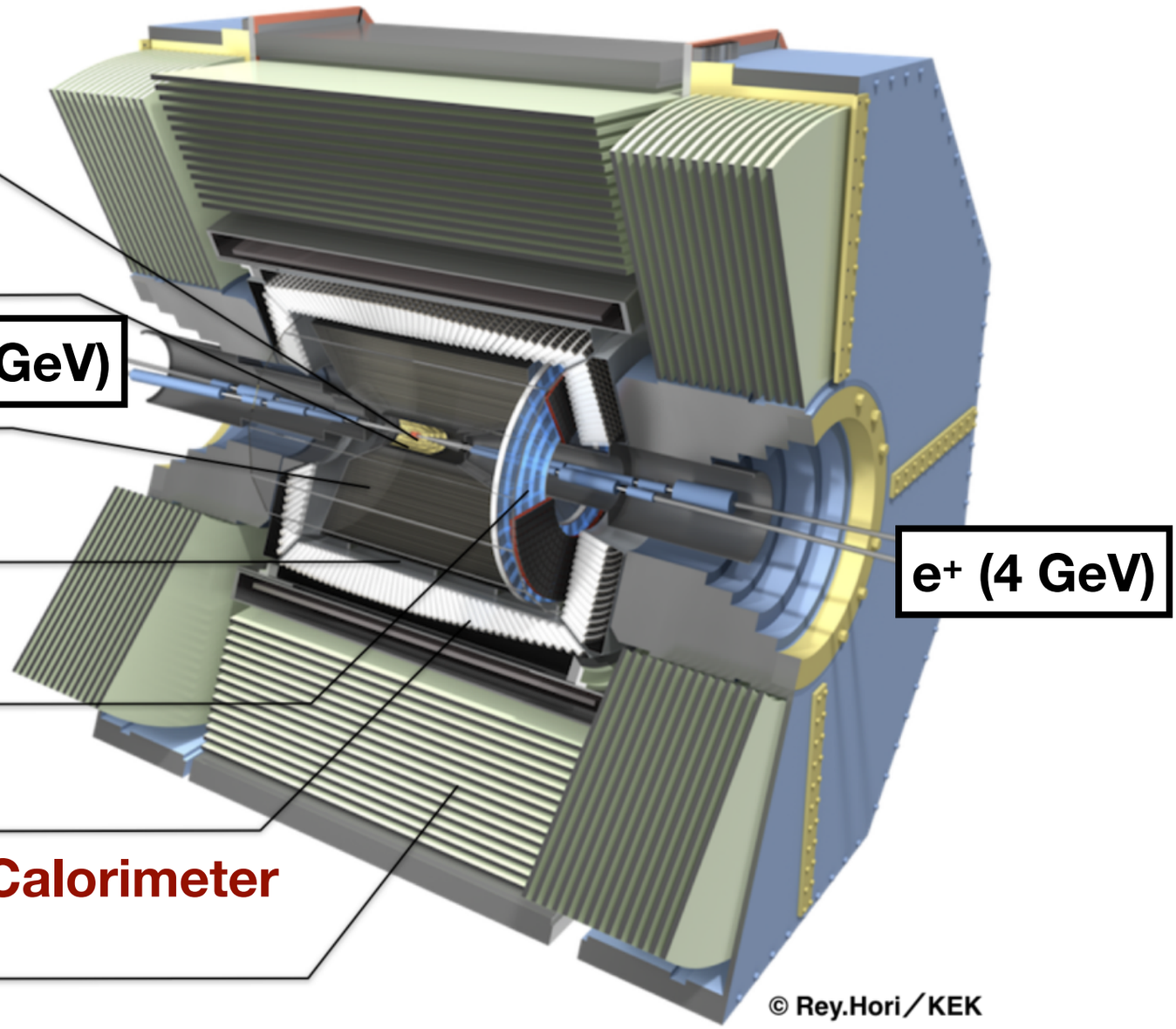
Electromagnetic Carolimeter（ECL）

Calorimeter

K^0_L／Muon Detector（KLM）

Muon spectrometer

Outer



Issues in 2019 runs

Data taking efficiency

- Efficiency is calculated as recorded/delivered luminosity
- Difficult to identify the inefficiency sources

➡ Bad efficiency (less than 70%)

- Link lost at the readout system
- Link lost at the trigger and timing distribution
- Errors in the software trigger (HLT)
- Mis-communication between the control room shifters and the sub-system experts
- (and combination of the above errors)

➡ **Error diagnoses** and **quick recovery** are essential to achieve better data-taking efficiency

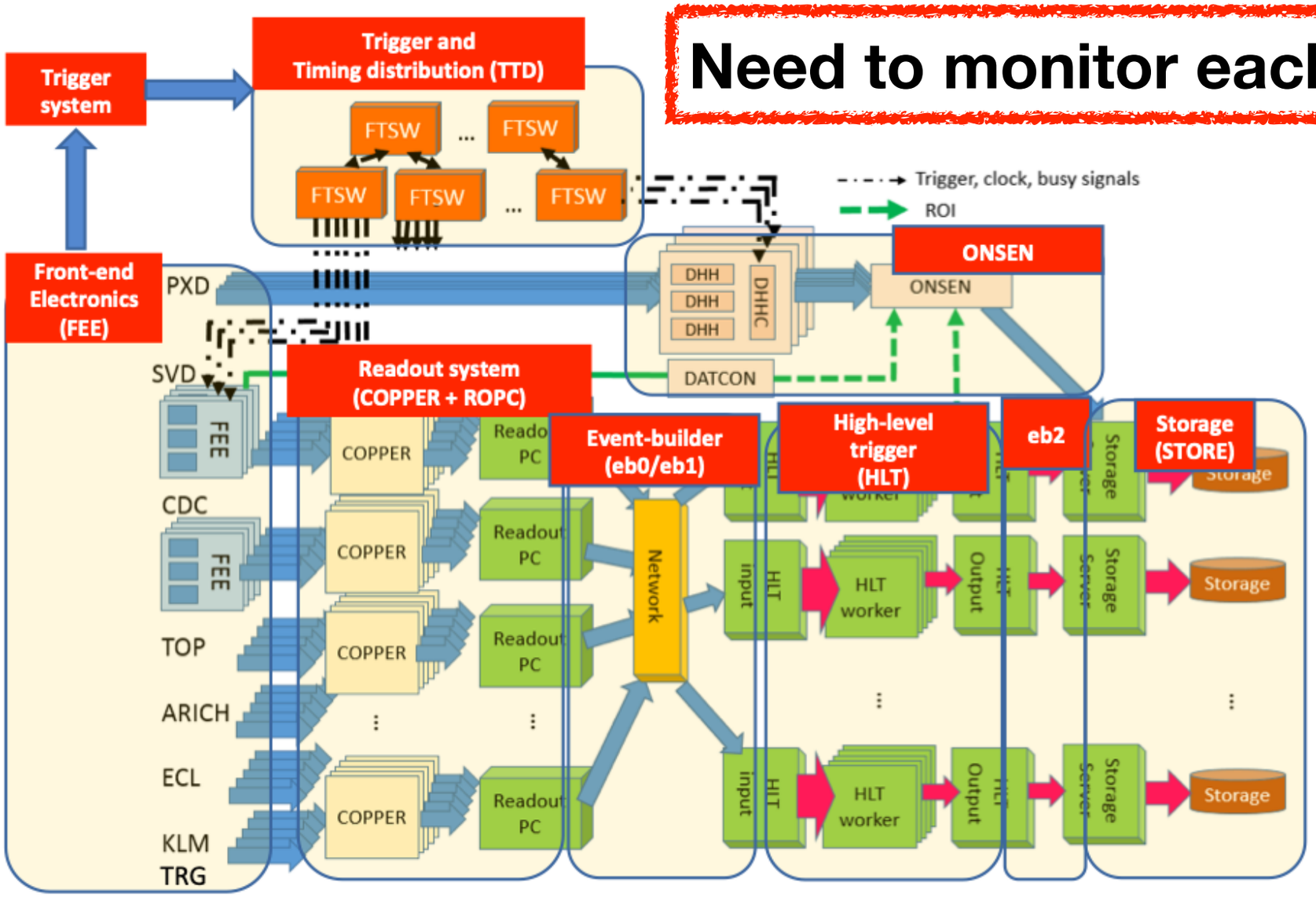
➡ Need to monitor the Belle II DAQ system efficiently

Belle II DAQ system

Distributed hardware and software components

- Detector front-end electronics
- Readout system + Event builder
- 2-staged trigger system (Level-1: hardware, HLT: software)
- Storage

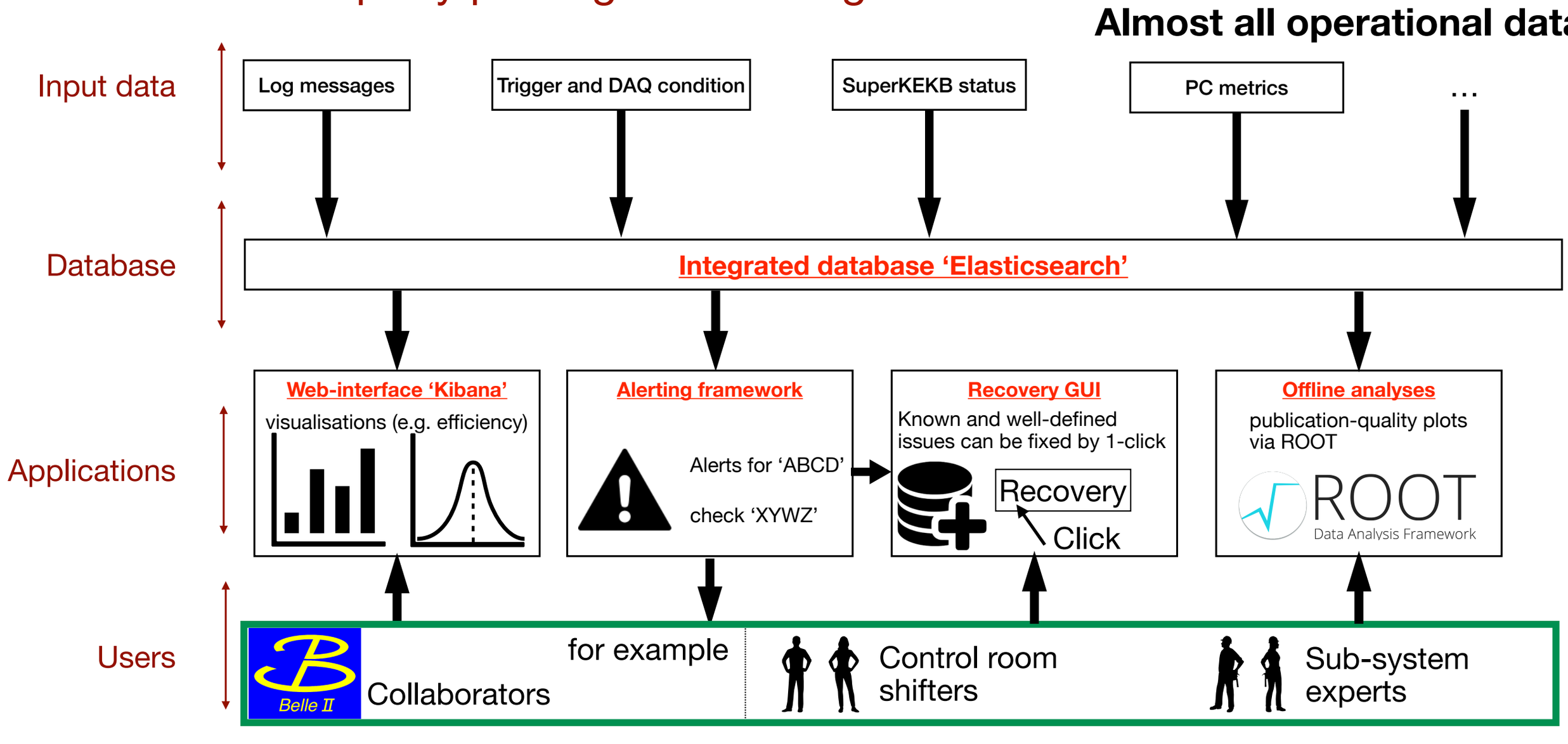
Need to monitor each component



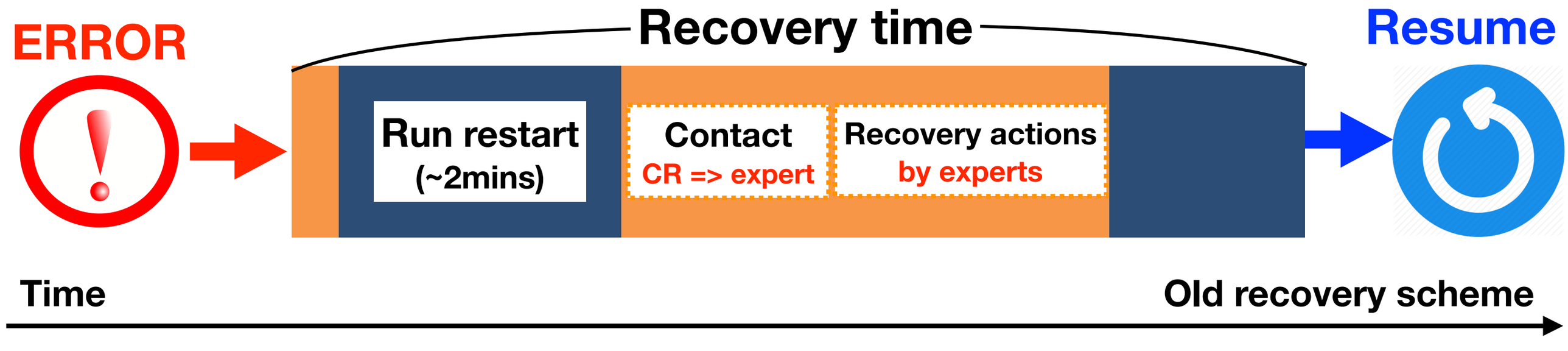
Monitoring system

Efficient monitoring system developed via Elastic stack

- Elasticsearch: distributed, JSON-based, and fast search and analysis engine
- Logstash and beats: data shipper/converter
- Kibana: web-interface for visualisation
- Elastalert: third-party package for alerting



Strategy to minimise the downtime



There were too much **“overhead”** for recovery

➡ Removed the overhead by introducing **the automatic recovery**

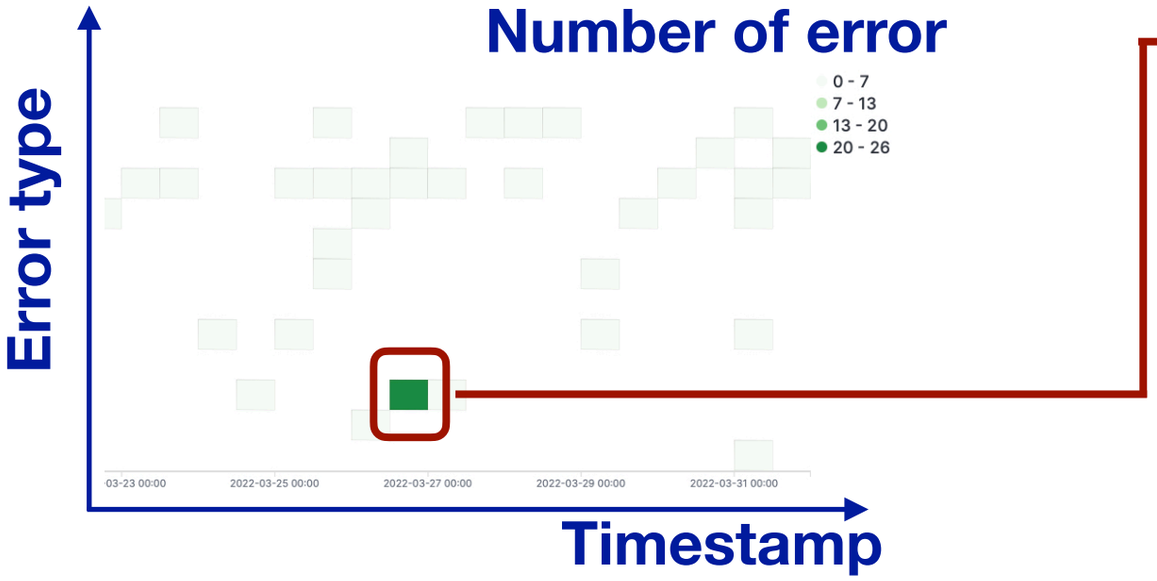
Automatic recovery

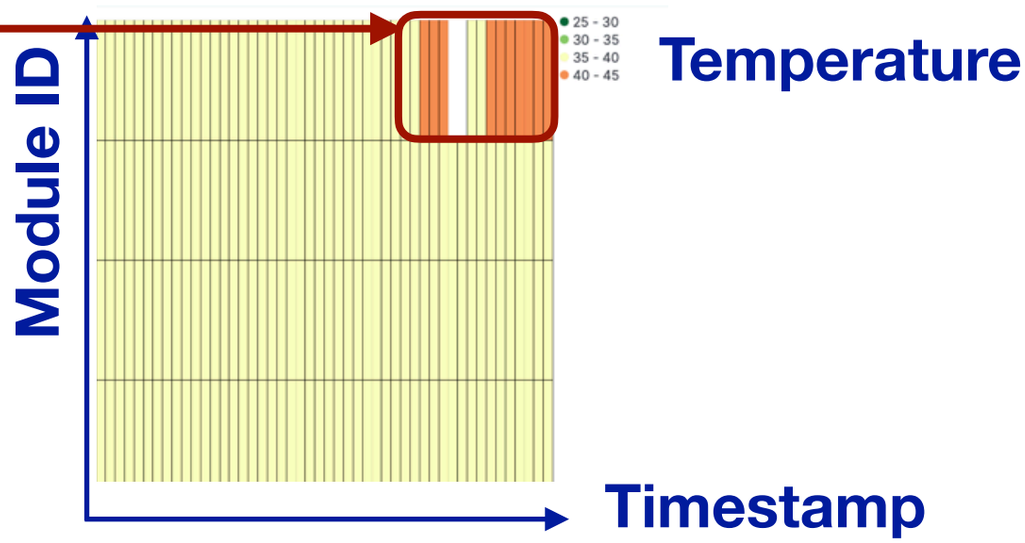
- Three steps defined:
  1. Define the condition for the error detection
  2. Manual recovery via the GUI (“1-click” recovery)
  3. Automatic recovery

Step 1: Error diagnoses

Online visualisations

- Kibana accesses Elasticsearch and visualises operational data
- Any time-series data can be visualised in 1D/2D histograms, graphs





- Frequency of each error
- Temperature of hardware component
- Correlation between data-flow and luminosity
- Efficiency and inefficiency sources of data-taking

Correlation between different operational data can be checked

- Identify the error source and try to remove it
- If it takes long time to remove the error source:
  - ➡ Implement recovery actions in step 2 and 3

Alerting system

Elastalert

- is a third-party tool allows us to implement **alerting function**
- **Alert to:** e-mail, RocketChat, or **perform command**

Elasticsearch (database)

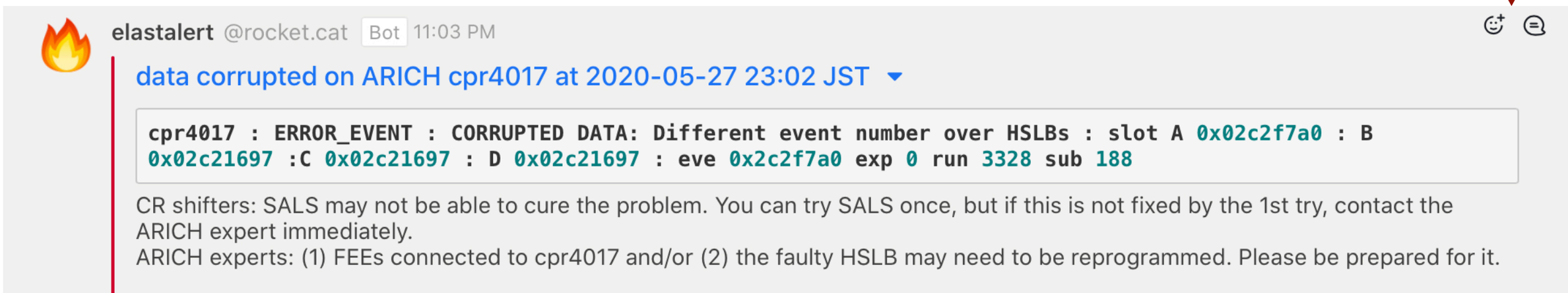
Check

Elastalert

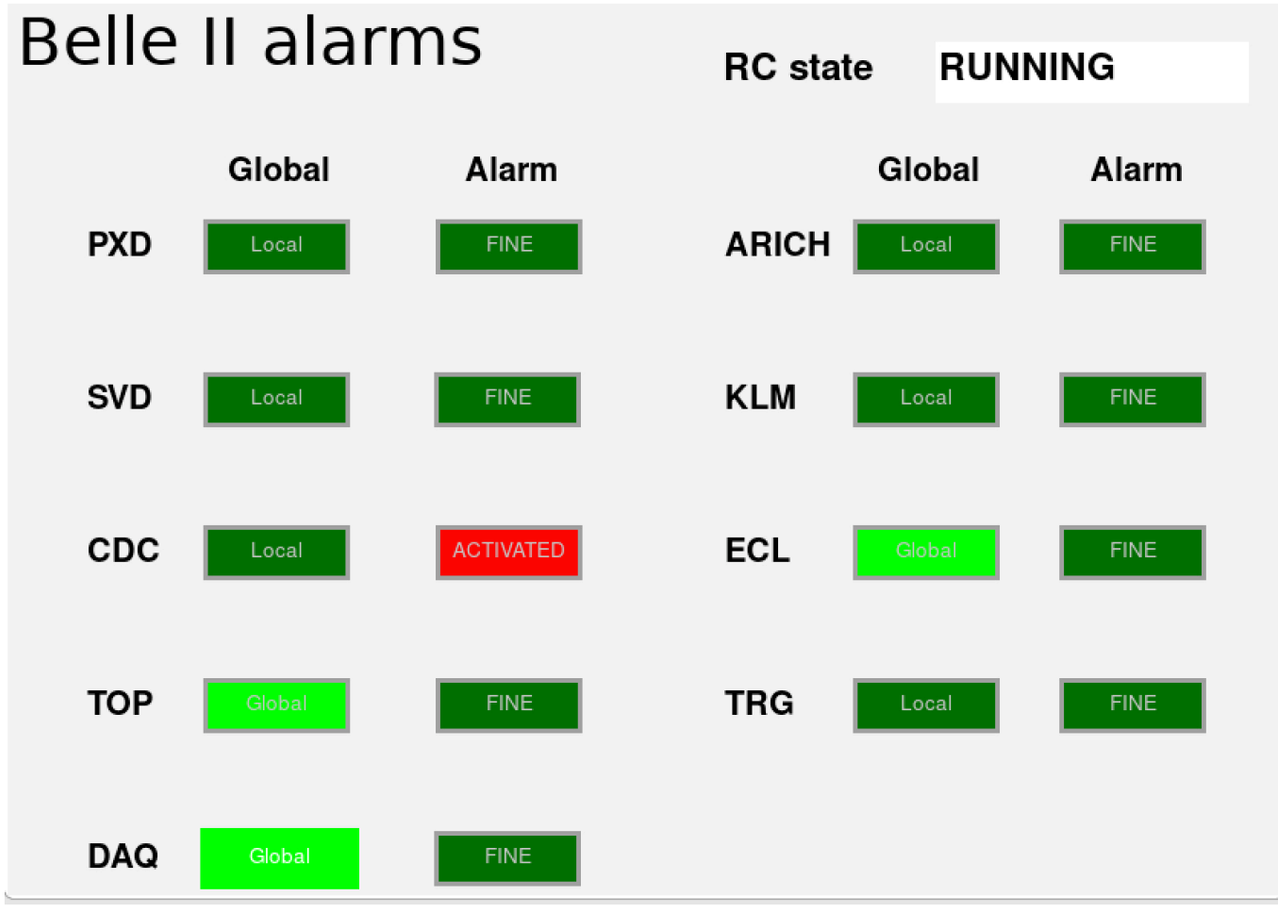
If rule is satisfied

If some parameters changed
If specific log-messages detected
If error frequency is higher than a threshold
If there is spike-like errors
If there is no heartbeat-log from a node

Example: Automatic advice for the control room shifters via the RocketChat (chat tool)

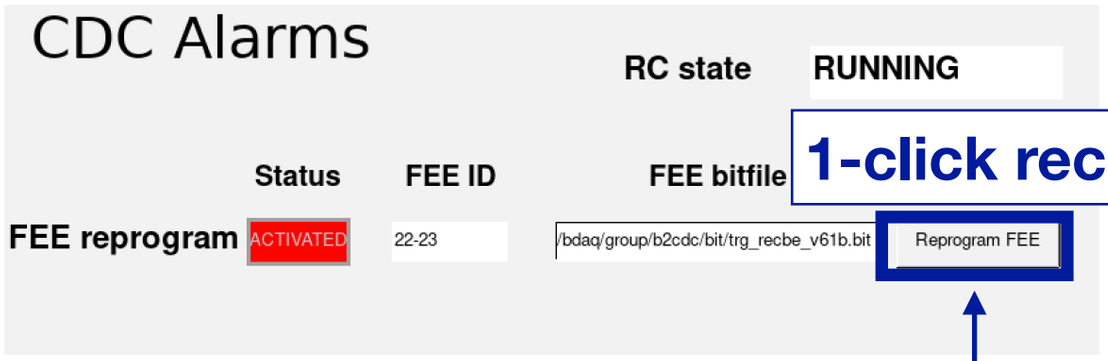


Step 2: Manual recovery



based on EPICS and CSS opi

- Alerting system activate the alarm corresponding to the detected error
- Only activated alarm is shown



1-click recovery

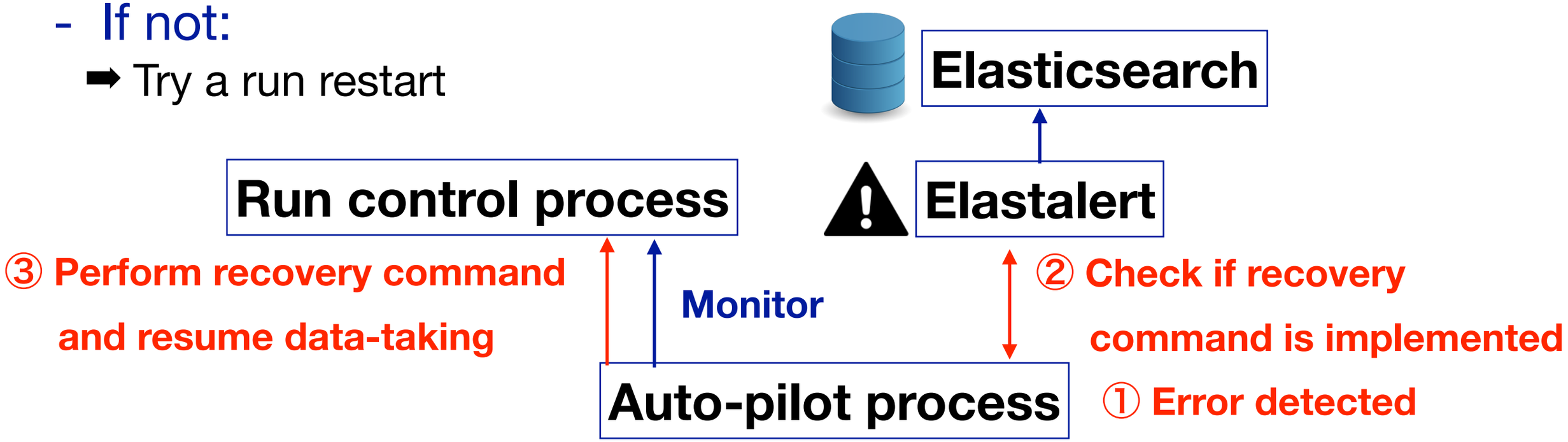
Associated with a proper recovery command

- Before trying the automatic recovery, we brush-up the recovery command in the manual recovery via a GUI
- Shifters perform the appropriate recovery command **without any confusion** and **without contacting experts**
- Recovery command is well-tested in the manual recovery
  ➡ Automatic recovery in step 3

Step 3: Automatic recovery

Auto-pilot mode

- can take the required actions in case of:
  - Beam dump
  - HV trip
  - ERROR and BUSY
- communicate with Elastalert to check the automatic recovery command is implemented
  - If recovery command is implemented:
    ➡ Automatically perform the recovery command, and resume data-taking
  - If not:
    ➡ Try a run restart



Alert configuration

Each alert is defined in a yaml file

This alert is triggered when a specific error message is detected

① For manual recovery (activate the alarm)

Actual recovery command is defined in the GUI-side

EPICS alarm process variable (PV)

Problematic module IDs

```
52 command: ["/home/vagrant/bin/caput.py", "B2_alarm:CDC:FEEREPROGRAM:ID", "{FEEs}"]
53 recovery_command:
54   host: "ttd11"
55   user: "b2shift"
56   rcrequests_before_recovery: "STOP-ABORT"
57   command: "cd scripts/cdc_config && ./config_parallel.py --ID caget(B2_alarm:CDC:FEEREPROGRAM:ID)"
58   reset:
59     PV: "B2_alarm:CDC:FEEREPROGRAM:ID"
60     value: "-1"
```

Actual recovery command to be performed

② For automatic recovery

Auto pilot process receives the information:

- Which alert is triggered
- Actual recovery command
  - On which host, and by which user, the recovery command need to be performed
  - Reset command

Summary

- Belle II data-taking is going well while we have encountered various problems in our data-taking
- To minimise the time without data-taking, we have developed an **efficient monitoring system** based on Elastic stack
- Using the system, we achieve the automatic recovery in three steps:
  1. **Error diagnoses** via the online visualisations
  2. **Manual recovery** as a intermediate step to establish the proper recovery action
  3. **Automatic recovery**

Future plan

- To implement more automatic recoveries
- To implement a new feature: e.g. tolerable error which does not stop run but affects data quality ➡ Automatic recovery at the next run stop
- Full automatic data-taking combined with the detector control (such as high-voltage control)
- If possible (Machine learning to catch a sign of the up-coming errors)