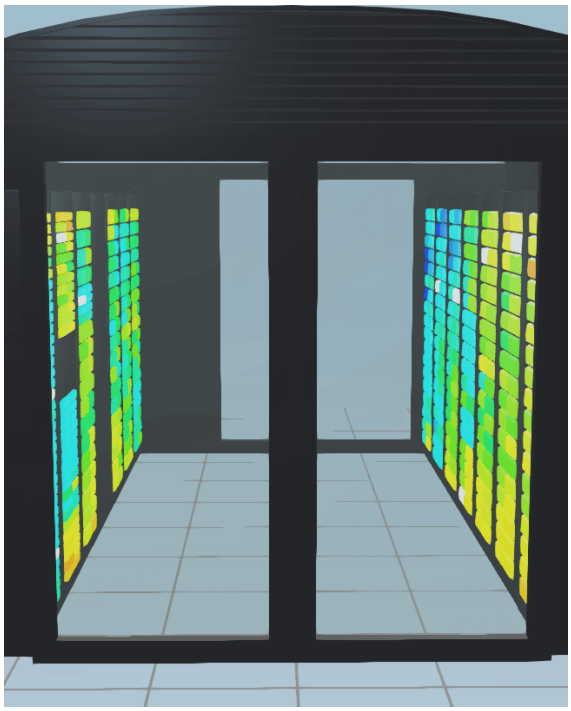
# Visualising HPC System's Load

*Petr Stehlík*



Energy efficiency is one of the most timely problems in managing HPC facilities which can be addressed at different scale and perspective. Using Internet of Things technologies this project focuses on visualising data collected from the Galileo supercomputer in a web application.

The current monitoring system[1] consists of several layers which allow to aggregate in a single point heterogeneous data sources which consist of computing elements, node, job scheduler and facility telemetry of the Galileo supercomputer located at CINECA, Bologna, Italy.

The system was named *ExaMon*[1] and is built on top of MQTT protocol[2] which allows measured metrics to be send to a central broker where received data are processed and stored in KairosDB database utilizing Cassandra cluster.

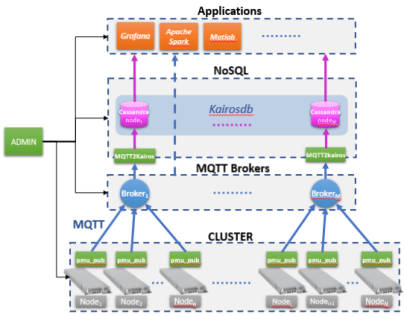



**Figure 1:** Examon Architecture

[1] Shorthand for Exascale Monitoring

This enables us to post-process data in time-oriented fashion in order to visualise them on a time-line and as a single number as well.

Current implementation uses Grafana framework to visualise data stored in KairosDB. Grafana will be replaced by the project's result of creating a dedicated web application for defined use-cases with 3D model of a cluster room showing various metrics of the whole HPC system with focus on energy consumption and efficiency.

## Methods

The whole project can be separated into three phases. First phase is data anylysis where the whole dataset of available metrics was presented, how they are distributed and eventually processed on the back-end. Datasets can be divided into multiple levels of aggregation:

### Per-core level

The most low-level data can be found in core's registers such as IPS, Lx-cache misses and more. It also provides info about its load and temperature.

### Per-CPU level

Each node consists of two CPUs and each of them can provide data about its C-states, energy counters and its frequency.

### Per-node level

Most of the information available on node-level basis are coming from IPMI.[3] Via this interface we can access info about node's utilization, multiple temperature sensors and average power consumption.

### Per-cluster level

The Galileo's cluster room was equipped with several environmental sensors. This dataset is not currently available due to technical problems.

### Per-job level

Data gathered using the PBS scheduler's hooks. This dataset is aside from previous ones since it points to allocated and used resources of the job submitted to the queue. This data are stored directly to Cassandra cluster ommitting KairosDB.

With each level we can aggregate the lower levels (except job-level data).

This is especially useful for core-level data which are mostly too dense for any comprehensible visualisation.

### Visualisation

Second phase was to visualize data stored in KairosDB in simple yet insightful way in a lightweight web application. The application, called ExaMon Web, uses Angular framework as its base on top of which several other libraries were used. Worth mentioning are Dygraphs and Bootstrap. The former library produces powerful time-oriented charts utilizing the canvas element in web browser. The latter is a CSS framework to produce uniform user interface across the whole application.

Compared to Grafana, the created web application feels more lightweight, fast and easier to use because of the prepared datasets which are being used. The balance between configurability and the ease of use must have been found. We concluded the best way to achieve this was to enable time selection on given datasets but restrict configurability of the charts themselves. This way user is not bogged down with configuration and only focuses on prepared data.

If there is such desire to see other metrics the Grafana framework is still available right next to the ExaMon Web. As an additional feature, compared to Grafana, we can perform more advanced queries using the KairosDB REST API.

### Live Data

The last phase was to utilize the live stream of MQTT messages right in the ExaMon Web. Two use-cases were defined for the MQTT messages depending on their origin.

### PSB Jobs

Each PBS job goes through a specific set of events during its lifecycle and every job is assigned a unique job ID. Using the ID user can subscribe to such MQTT messages and view various information on the ExaMon Web job dashboard. The dashboard also uses Cassandra cluster in case the job is already finished and stored in the cluster. This way user can see additional data about their job.

Using the job data a user can view detailed info about allocated resources of the given job such as CPU load, system utilization and more as seen in 2. With this information the user can asses some conclusions about their program. How effective it is, where are the slow parts and even perform a top-down analysis for performance issues. Also they can view how the program performed in terms of energy efficiency.

### 3D Model of Cluster Room

Second use-case is designed for general public and partly for system administrator. The use case is separated into two different parts. First one is very similar to job dashboards where data are displayed as time-serie charts with the difference in aggregation level which is at the cluster level. This means we can easily display, for example, the cluster's CPU load.

The other part is the most crucial in terms of interactive data visualisation. An accurate 3D model of the Galileo cluster was created using Blender and with the help of Blend4Web incorporated into the ExaMon Web. Even further, deeper integration was realized utilizing WebSockets (using Socket.io library) that enables us to create a reactive paradigm model instead of polling-based one. The model inside the page receives live data that has been published by the nodes and send to the broker. A subscription model was developed to accomodate large amount of visitors. The model then colours each node based on the minimum and maximum value of all received data. Weighted moving average was used in order to accomodate for sudden spikes in data using the given formula:

$$v_{new} = v_{current} + v_{previous} \times (1 - \alpha)$$

where $v_{previous}$ value is set to the first available value and $\alpha = 0.75$ as a default value was chosen based on short-term evaluation.

## Results

ExaMon Web can be split to two major parts: 1) Tool for overseeing job submitted to PBS queue and 2) Cluster-level visualisation and analysis each of them designed with a prepared use-case scenario.

### Job Visualiser

The main task of job visualiser is to inform a user about their submitted job. The job ID is then used in the ExaMon Web job lookup. UI offers to query by manual input or, for ease of access, the list of active jobs and the last finished job are shown.

In order to capture and keep this lifecycle data a new tool in form of a Python class had to be developed. The `JobManager` subscribes to MQTT topics that send job-related data. This data is then stored in internal volatile database. This way it can keep track of all submitted jobs. The `JobManager` is aware of the job's state and sorts the incoming job data according to their lifecycle.

The class is designed to be expandable and configurable. This means we can add callback functions to certain points of job's lifecycle. This is used when a user subscribes to a live job and with each new received messages with the same job ID the job record in the database is processed and send to the user via a WebSocket.

If the job is finished the application will assess all information as for an ac-

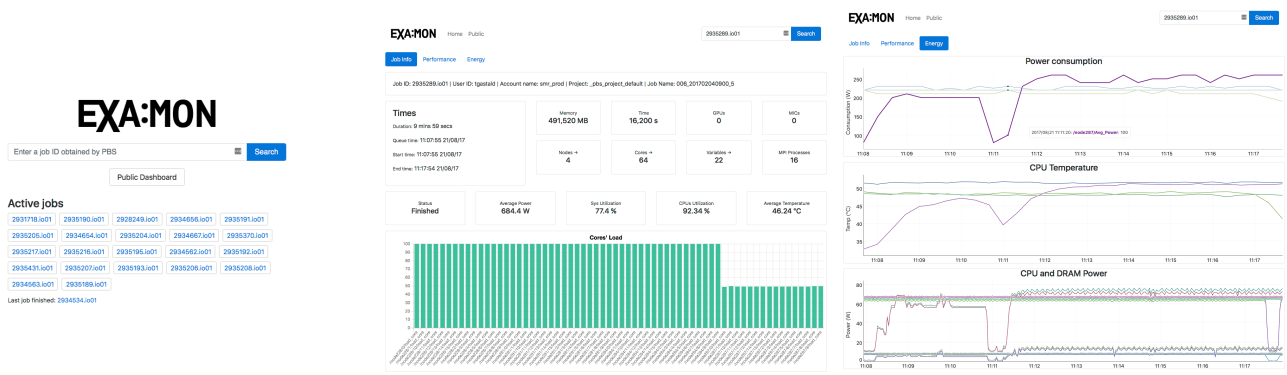
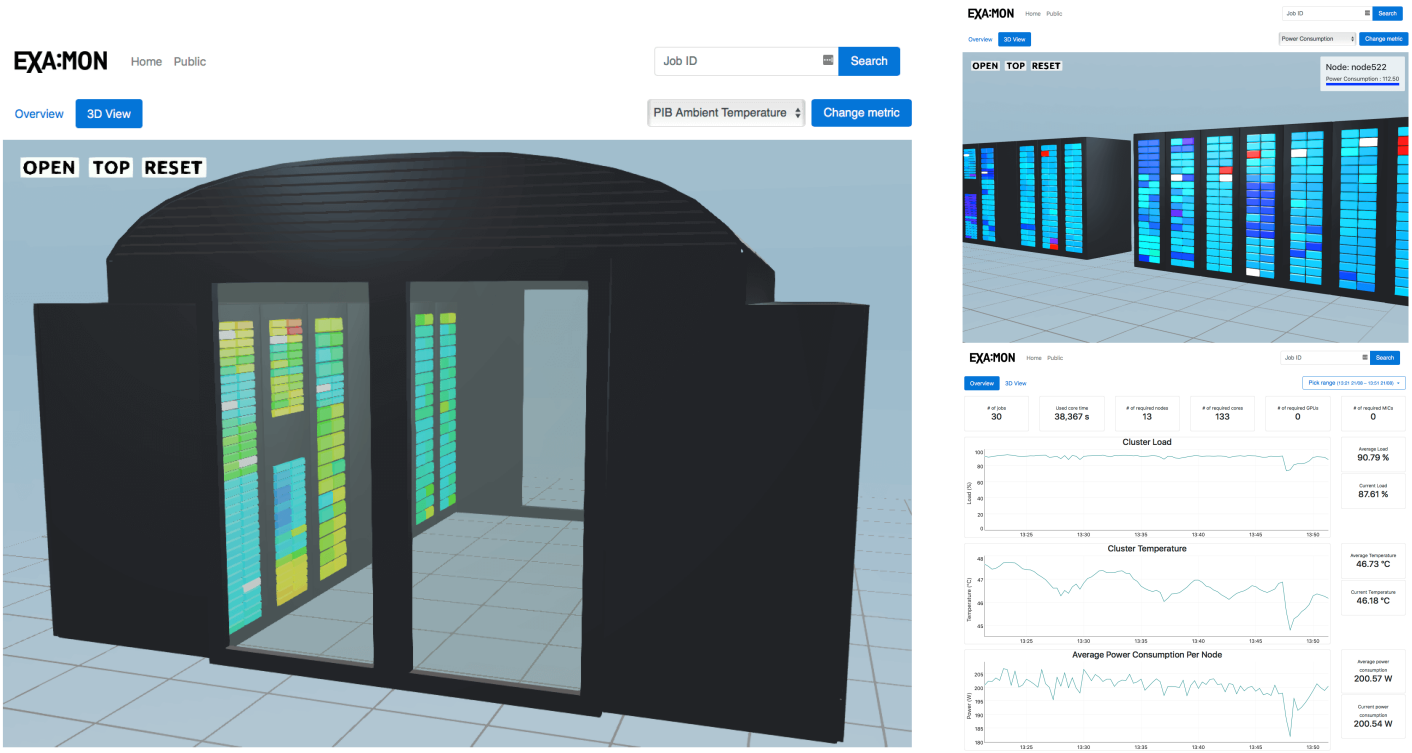



**Figure 2:** From left to right: intro page with jobs lookup, currently running jobs and last finished job; job's info dashboard with a finished job; job's energy dashboard.




The public section with 3D model and its possible other possible arrangement. Also a public overview dashboard with preselected timerange of 30 minutes.

tive job and add additional info to the page as seen in 2.

In both cases a user can view the perfomance and energy usage of their job. Each of the dashboard disposes of preselected interactive time-series charts.

### Cluster Visualiser

The cluster visualiser is very similar to job visualiser in terms of used components and the form of data. The main difference between them is that cluster visualiser is mainly designed for general public which doesn't run jobs on the cluster but is interested on how a HPC facility performs.

The intro dashboard shows several charts aggregated to cluster level with averaged and last values right next to each chart. User can also select time range in which the data will be shown.

The second part a precise 3D model developed in Blender and used in Blend4Web framework which is incorporated into the ExaMon Web application. The model utilizes the same class as Job Visualiser to capture incoming MQTT messages but this time for metrics published by cluster's sensors and tools. The manager computes weighted moving average which are then available for querying.

Once a user opens the 3D model the application subscribes to given metrics and waits for available data. A minimum and maximum value is computed and according to these values each node is color-coded in usual colors ranging from red to blue in a HSL color model.

The application then receives new data for each node as soon as they are made available by the manager and recolors given node.

User can operate the 3D model in usual way (panning, rotation and zooming) and can see detailed info about each node by clicking it. This highlights the node and show a legend.

## Discussion & Conclusion

The ExaMon Web application was successfully developed and plans for public deployment are already arranged. The application is already running on one of CINECA's virtual machine in staging environment. The expected web application was delivered with several improvements and additions which will help the team at UNIBO to further develop the whole ExaMon system.

The application can be further expanded with new dashboards such as *System Administrator* dashboard. Such dashboard can help the system administrator of Galileo supercomputer to quickly assess valuable insight of the whole cluster which would be otherwise very complicated and time-consuming.

Using this application CINECA can show general public how a supercomputer performs and what it takes to run it. The users of Galileo can comfortably and easily view detailed info about their jobs and how they behave in real-time and in real-world conditions and adjust their programs to perform better and more efficiently.

## References


[1] Beneventi, Francesco, et al. "Continuous learning of HPC infrastructure models using big data analytics and in-memory processing tools." 2017 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE, 2017.

[2] Locke, Dave. "Mq telemetry transport (mqtt) v3.1 protocol specification." IBM developerWorks Technical Library (2010).

[3] Kaufman, Gerald J. "System and method for application programming interface for extended intelligent platform management." U.S. Patent No. 7,966,389. 21 Jun. 2011. APA


### PRACE SoHPC Project Title

Web visualization of Energy load of an HPC system

### PRACE SoHPC Site

CINECA, Italy

Petr Stehlík

### PRACE SoHPC Authors


Petr Stehlík, BUT, Czech Republic


### PRACE SoHPC Mentor

Dr. Andrea Bartolini, UNIBO, Italy

### PRACE SoHPC Software applied

Angular, Dygraphs, Bootstrap, Blender, Blend4Web

### PRACE SoHPC Acknowledgement


I would like to express my gratitude to all people at CINECA and UNIBO who made this project possible and to all people who helped during the development of ExaMon Web. I would like to also thank my family and close friends for all the support I received.


### PRACE SoHPC Project ID

1705