pMp - Web based monitoring of the data processing and simulated production at CMS

Antanas Norkus (Vilnius University)

Giovanni Franzoni (CERN)

Norraphat Srimanobhas (Chulalongkorn University)

Jacob Walker (University of Sheffield (GB))

Adrian Alan Pol (CERN)

on behalf of CMS Collaboration

Simulated events production in CMS

- Using centralized service (McM) to hold the configuration of the processing jobs, provide their bookkeeping and prioritization and submit the requests of events production to the computing resources.
- CMS data reconstruction requests are submitted to computing infrastructure.

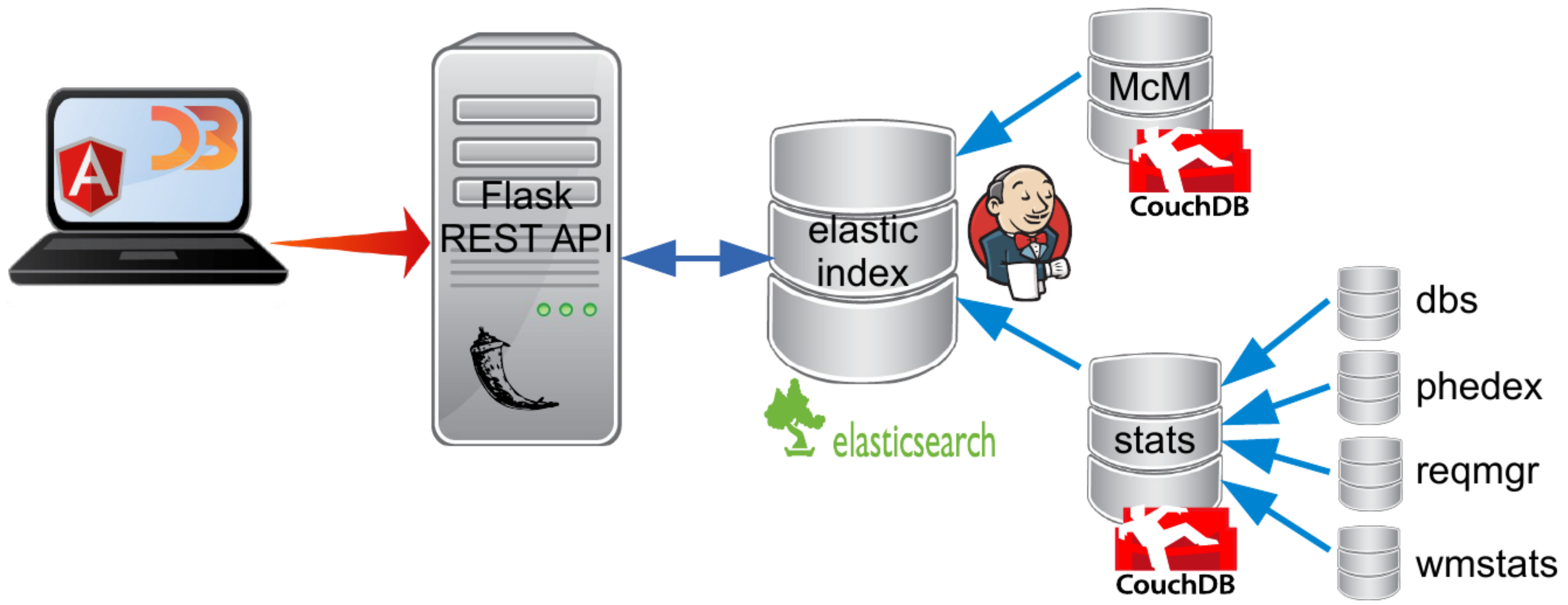
What is pMp?

- pMp is a web based monitoring for various types of requests of simulated events production or data processing submitted to CMS computing infrastructure
- pMp is meant to address the needs of request managers, GEN contacs and analysts by providing:
  - current/historical plots of different type of requests (MonteCarlo, data reconstruction, etc.)
  - key statistics
  - accessibility to whole CMS collaboration

Why we needed monitoring

- Currently we have:
  - **127 campaigns.** Highest level of abstraction where requests belonging to a campaign share same software version and major customizations e.g. mass energy, pile up etc.
  - **407 flows.** Defining how two campaigns are connected where one campaign is an input to second e.g. GEN-SIM (generated events-detector simulation) campaign is an input to DIGI-RECO (Emulation of electronics & High Level Trigger - Event reconstruction).
  - **105967 requests.** User specific configuration where he specifies physics process, needed number of events and customisation of the request parameters.
  - **139905 workflows.** Definition of request in CMS computing's infrastructure. Single request can have multiple workflows (resubmission, recovery etc.).

Infrastructure and technologies



We use:

- AngularJS, D3 for front-end
- Flask and python for back-end server
- Elastic search node for indexing input data.

- McM – MonteCarlo request management system.
- stats – data aggregation service collecting needed data from CMS computing infrastructure

Using pMp

- Via web browser
  - Gives you angularJS web page where you can search with autocompletion and D3js plots to analyze data
- REST API will return data in JSON format:
  - <pmp_url>/<query>/api/announced – returns current information (number of completed events, status etc.)
  - <pmp_url>/<query>/api/historical – returns how completed events grew over time.
  - <pmp_url>/<query>/api/performance/_ – returns requests information of it‘s lifetime events (status changes, priority, physics working group)

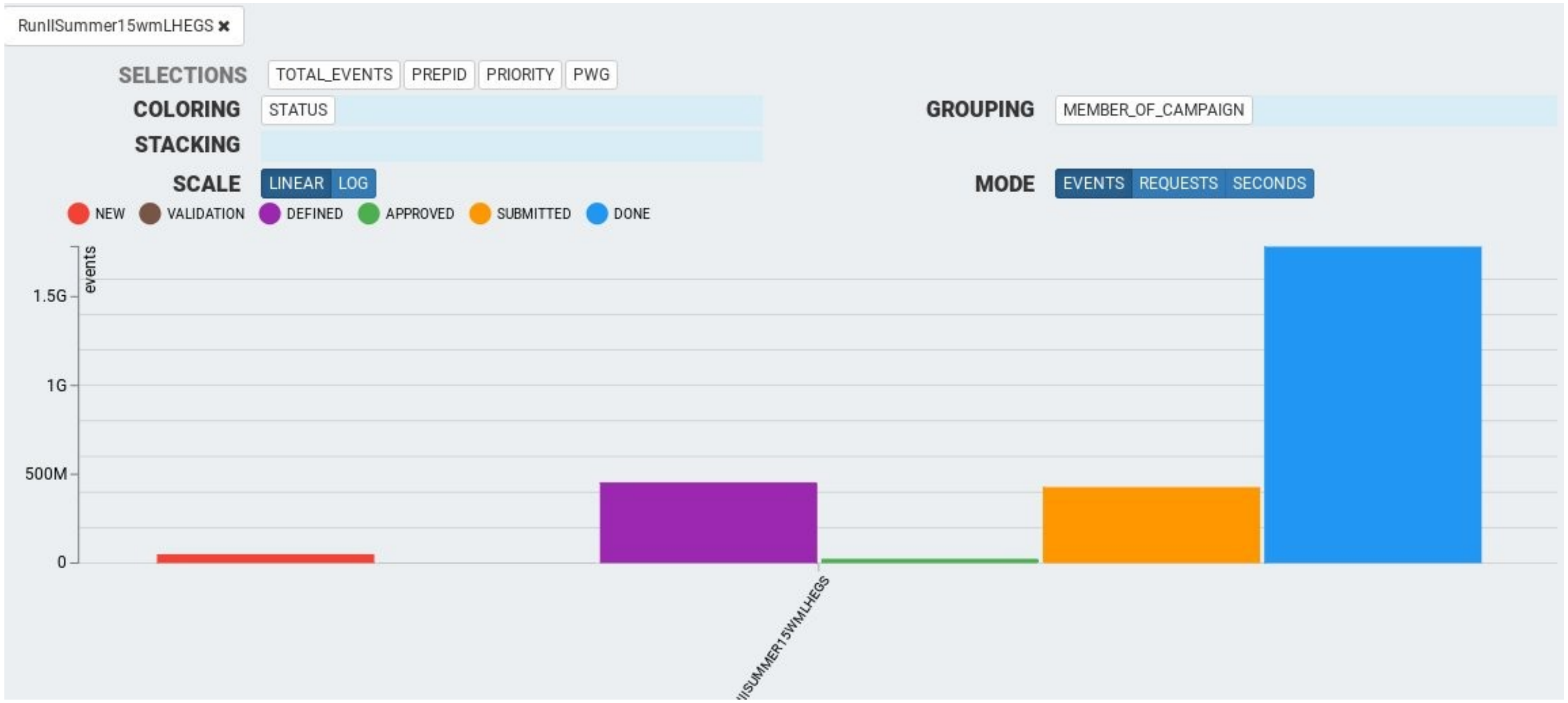
Pros/Cons

- Pros:
  - NoSQL and elasticsearch index data reliability
  - Service own index/DB means running not depending on other services availability
  - rolling updates needs minimal time to be up-to-date
  - REST API
  - Customizations
  - Bookmarkability
- Cons:
  - Data structure only for us, would need some time to support multiple data structure

Types of plots

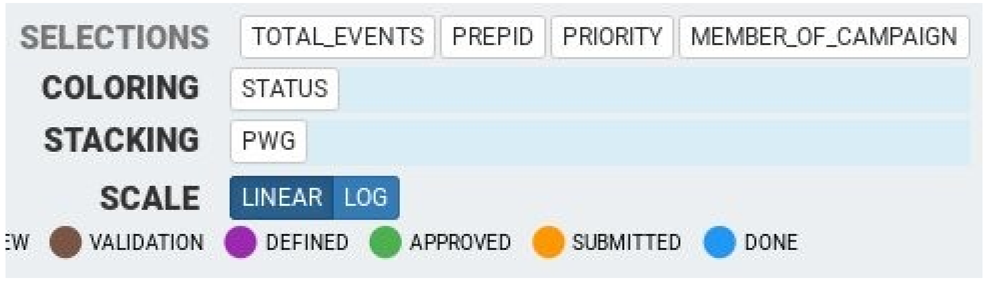
- ***Present***
  - Display current status of requests which are in one of statuses (new, validation, defined, approved, submitted, done, upcoming)
- ***Historical***
  - Is a linear chart displaying progress in production with a reference to expected number of events
- ***Performance***
  - Shows statistics for time intervals between request’s status

Present statistics(1)


RunIISummer15wmLHEGS ✖
SELECTIONS TOTAL_EVENTS PREPID PRIORITY PWG
COLORING STATUS
STACKING
SCALE LINEAR LOG
GROUPING MEMBER_OF_CAMPAIGN
MODE EVENTS REQUESTS SECONDS
● NEW ● VALIDATION ● DEFINED ● APPROVED ● SUBMITTED ● DONE
events
1.5G
1G
500M
0
IISUMMER15WMLHEGS

- Display current statistics in the system:
  - number of events done
  - number of events approved for submission
  - number of events defined and ready to be approved
  - number of events currently being tested
- Capability of dinamically filtering on name of specific request, flow, requestor physics group

Present statistics(2)


SELECTIONS TOTAL_EVENTS PREPID PRIORITY MEMBER_OF_CAMPAIGN

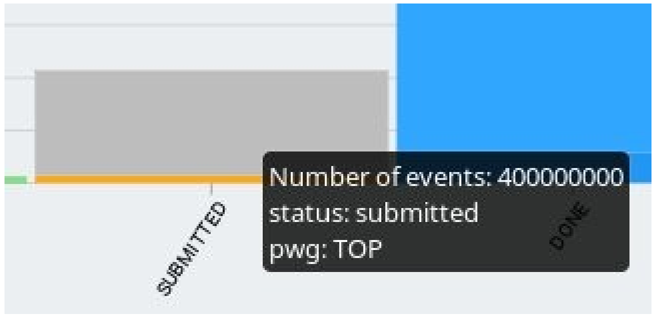
COLORING STATUS

STACKING PWG

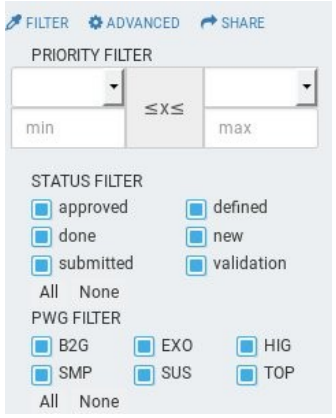
SCALE LINEAR LOG

EW ● VALIDATION ● DEFINED ● APPROVED ● SUBMITTED ● DONE

With plot customization user can specify key parameters how he want his plot be shown. Great for physics working groups to simply see their requests


Number of events: 400000000
status: submitted
pwg: TOP

On hover we can see details of each stack e.g. TOP physics group has 400M events submitted for selected campaign.



PRIORITY FILTER

min ≤x≤ max

STATUS FILTER

approved, defined, done, new, submitted, validation

All None

PWG FILTER

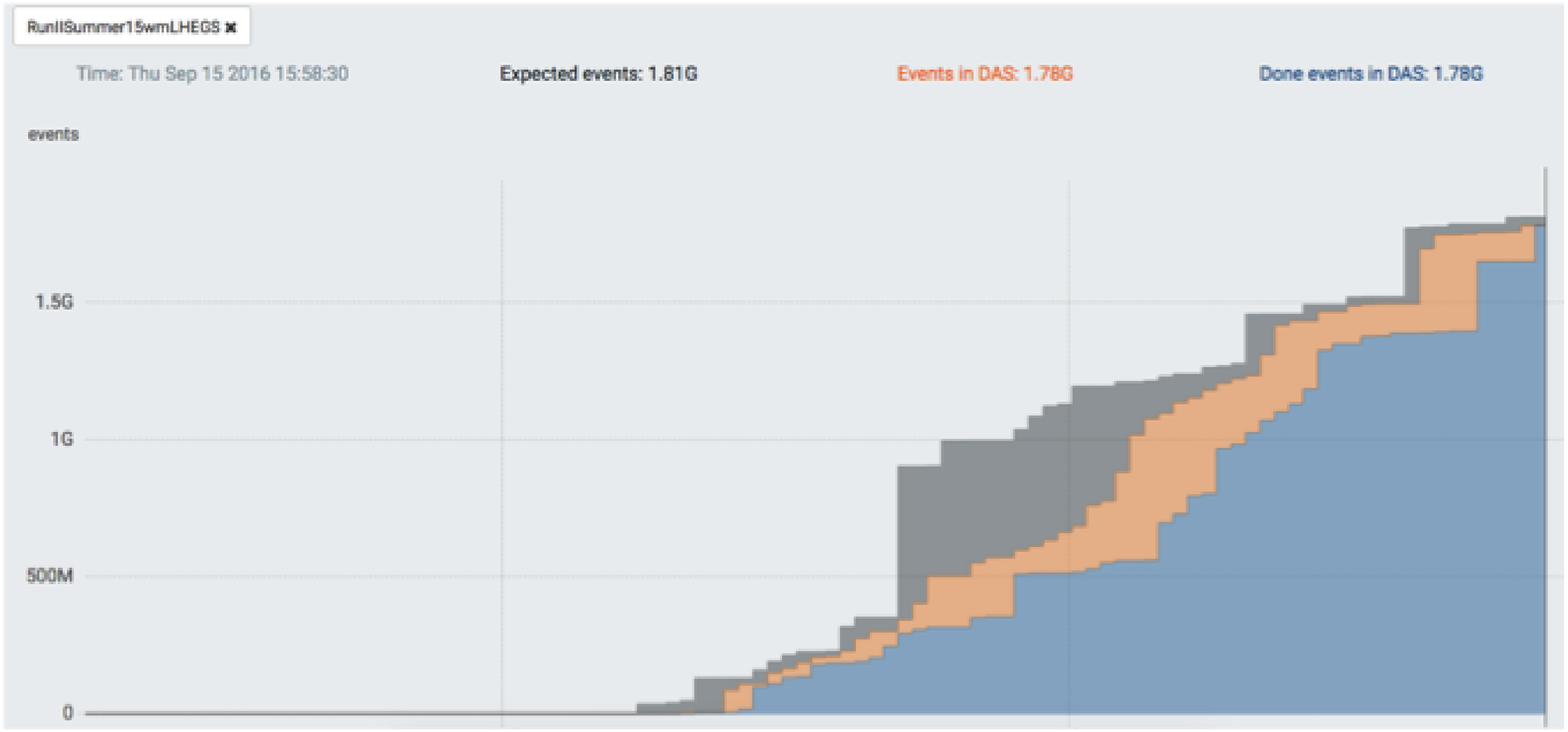
B2G, EXO, HIG, SMP, SUS, TOP

All None

If user doesn't want to see the big picture, he can filter out his selection:

- Specify priority of his wanted requests (lets say extra ugrent ones)
- Display requests only in specific status(-es)
- Show only selected Physics working group info.

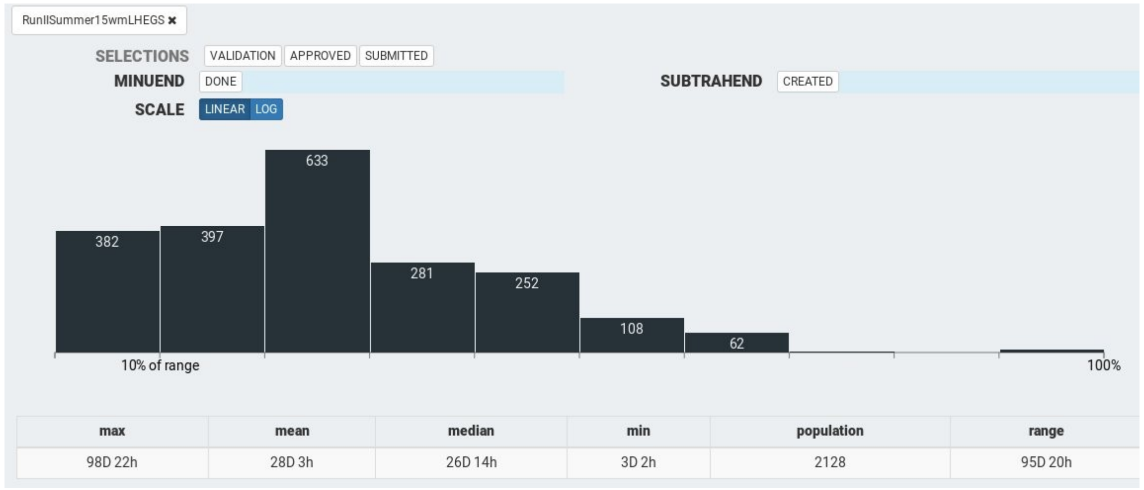
Historical statistics



Plot showing events completion over time:

- Grey showing total number of events submitted for processing
- Yellow showing completed events from currently running requests in system
- Blue displays total done events in datasets which are fully produced and ready for analysis.

Performance statistics

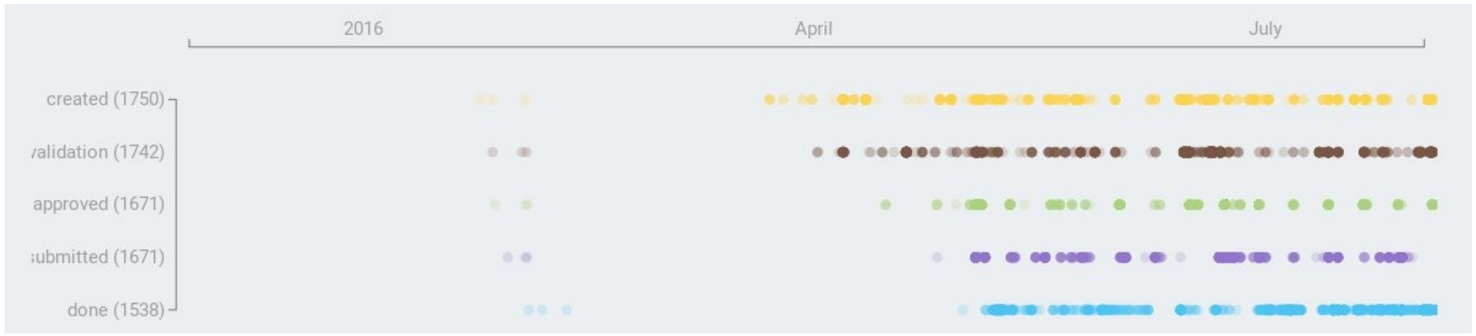

RunIISummer15wmLHEGS ✖

SELECTIONS VALIDATION APPROVED SUBMITTED

MINUEND DONE SUBTRAHEND CREATED

SCALE LINEAR LOG

max	mean	median	min	population	range
98D 22h	28D 3h	26D 14h	3D 2h	2128	95D 20h



- Chart histogram displays time distribution for all campaigns in selected campaign.
- Table provides key statistics, time, range and total number of requests.
- Scatter plot shows campaign‘s requests activity versus time.

What we learned

- Request manager infrastructure:
  - noSQL is schemaless, so your data structure can evolve without DB changes. You need even a basic monitoring to spot data inconsistency
  - **Never ever delete** entries from DB if you monitor history.
- Overview of production status:
  - Spotting possible production issues when underproducing/overproducing number of events