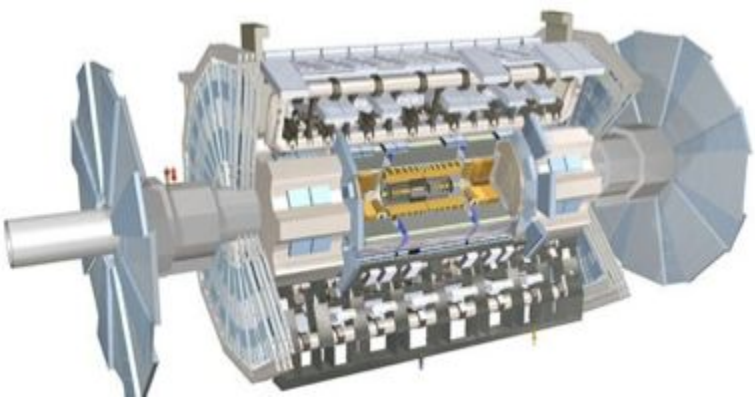
Physics menu definition and trigger monitoring at ATLAS and CMS

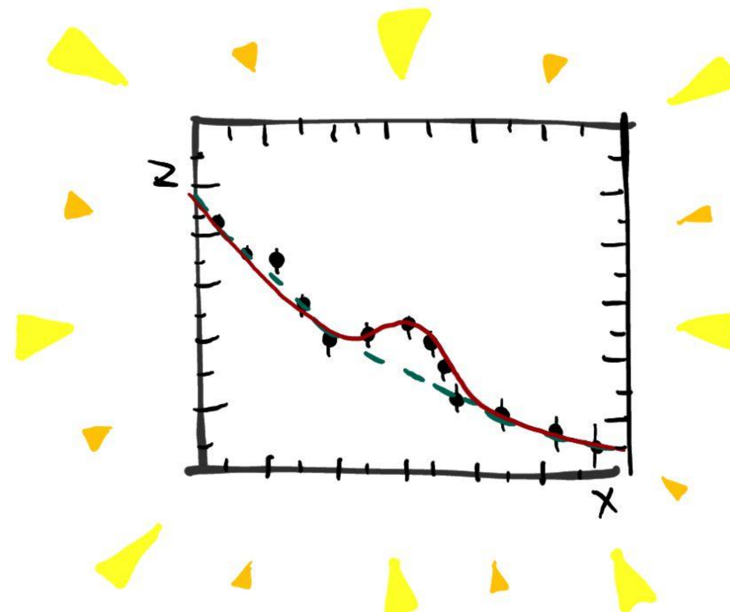
Heather Russell (McGill University)
on behalf of the ATLAS + CMS collaborations

HEP Software Foundation: Reconstruction and Software Triggers Working Group - 22 May 2019

the goal

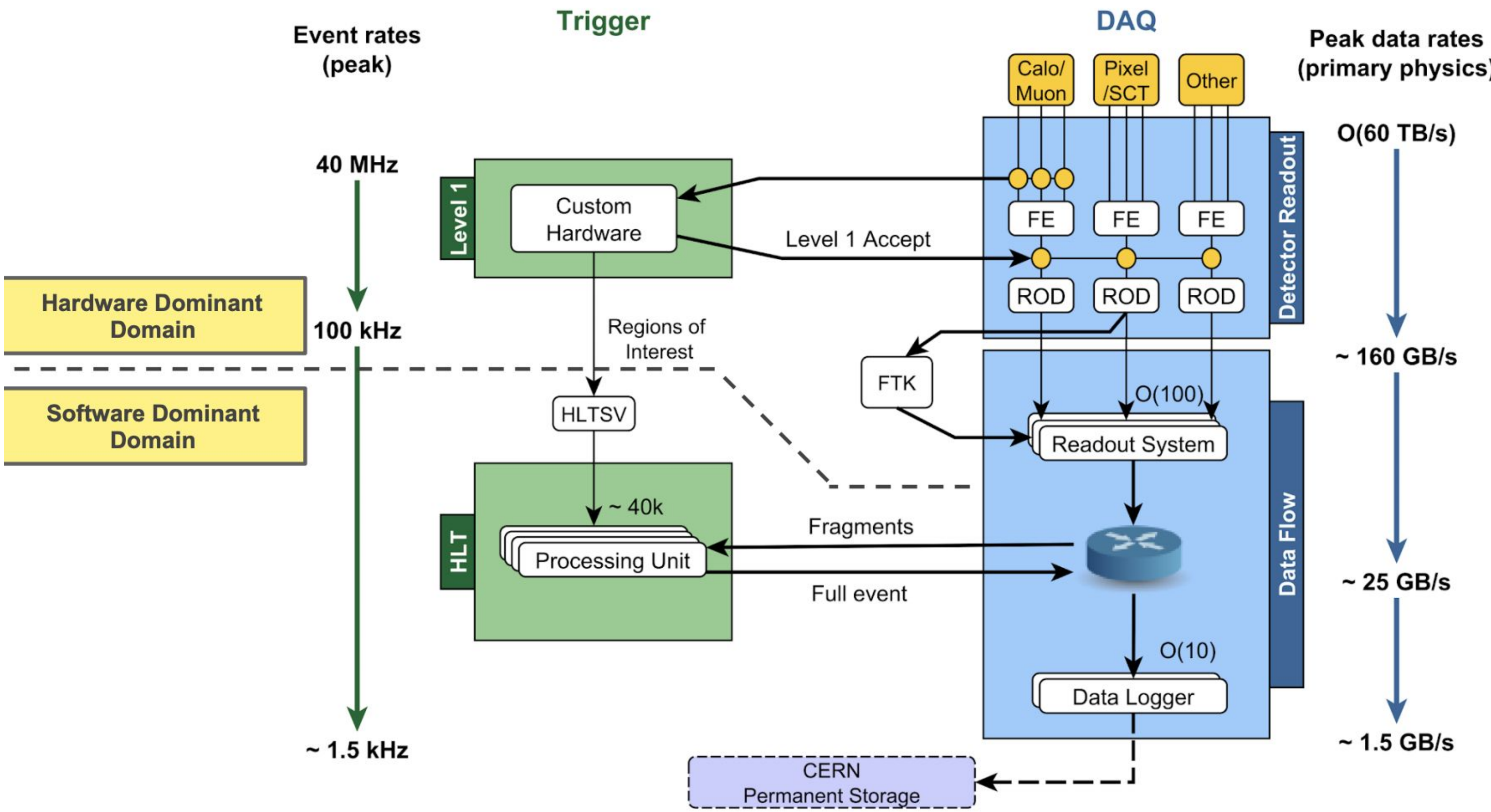


[or detector of choice]



the TDAQ system

graphic: Tim Martin - IEEE eScience talk



[in this case, ATLAS]

aside: a few definitions

what is a physics menu and why do we want to define it?

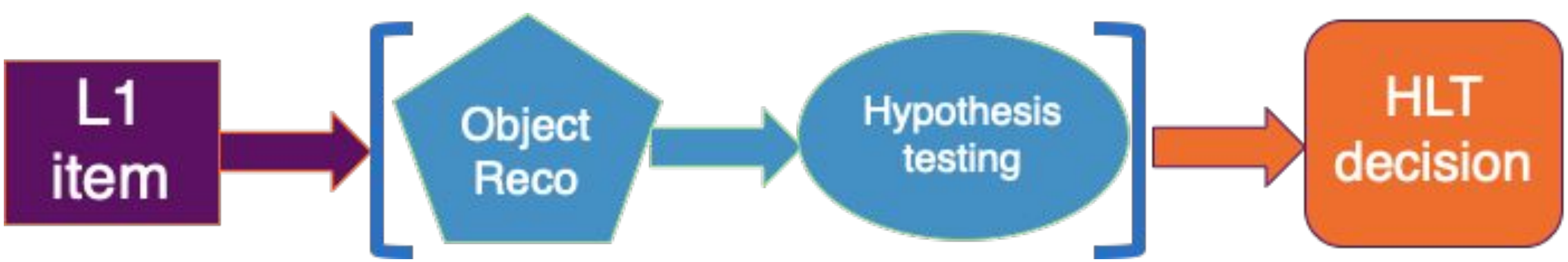
trigger menu: list of the types of events we select in each bunch crossing

for the purposes of this talk, **physics menu: the subset of triggers used specifically for analysis**

Each type of event is configured in the menu with a *chain (ATLAS)* or *path (CMS)*

Each chain/path has a *cost*, in terms of rate and CPU usage

Chain/Path:



aside: a few definitions

threshold: specific cuts applied in the trigger

e.g. E_T in a trigger such as HLT_j100: jet trigger with *threshold* 100 GeV

prescale: a factor *p* applied such that only every *p*-th event is recorded

prescale @ L1: the prescale *p* is applied at L1, after the hardware trigger decision is made

prescale @ HLT: the prescale *p* is applied *before* an HLT chain/path is run (so we don't waste CPU!)

disabled: the trigger prescale is set such that zero events are recorded

enabled: the trigger prescale is >= 1

online/offline: events that occur *during* a run are *online* and those *after* the run has completed are *offline*

defining the menu

roughly **three** ingredients in a physics menu*:

[basic, ‘generic’ triggers]

single muon, electron, photon, jet, b-jet, MET, dimuon, dielectron, multijet, etc.

[analysis-specific triggers]

e.g. two photons + one b-jet + MET + one electron with dPhi(electron, MET) > 2.5

[supporting triggers]

for background estimations and performance measurements, often lower-threshold items that are *prescaled*

*ignoring things needed exclusively for detector calibrations, monitoring, etc.

the request process

- physics analysis realises existing triggers are inadequate and develops selection
- analysis shows appreciable gain over existing menu and implements selection as trigger
- trigger rates, CPU usage are determined
- if necessary, selection or algorithms are tuned to fix rates, CPU usage
- trigger is validated in test samples (i.e. it does what it's supposed to)
- trigger becomes an official part of the physics menu
- trigger is deployed online
- first online data is scrutinized to ensure trigger performs as expected

rate determination

online, we know the rate of *all L1 triggers* defined in the menu, *even if they are disabled.*

hardware tells us rates for all defined triggers

HLT chains/paths are only known *if the chain is running* [recall: do not process events if a chain is disabled!]

so you might ask...

how do you figure out what rate (and CPU usage!) a trigger will have *before it runs online?*

rate determination

periodically collect special datasets

[CMS: zero bias (random), HLT pass-through; ATLAS: enhanced bias (mix of random+passthrough)]

rerun the L1 and/or HLT for *new* or *test* chains on this data

can also run with *exact* prescales that would run online: estimates of CPU estimates, total stream rates

nb: good tests of rates and CPU, but also the code and configuration itself!

putting everything together

menus are designed for the **peak luminosity** of the period they are going to be used in

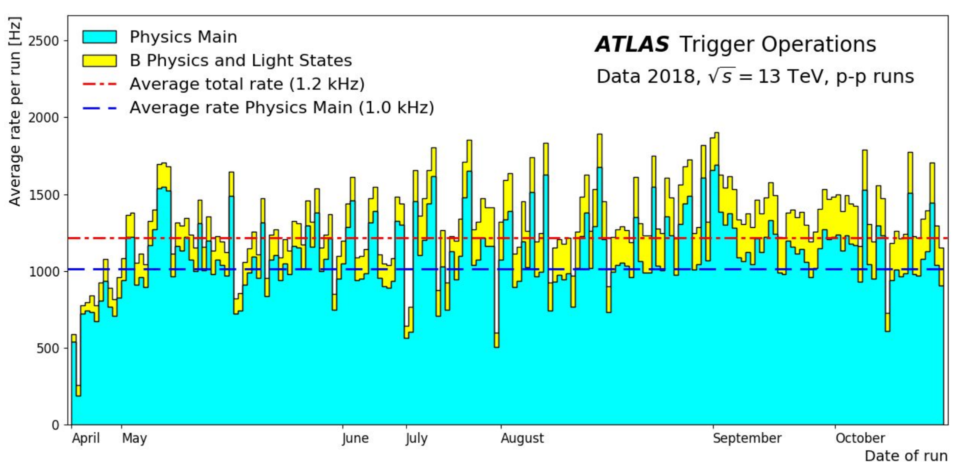
e.g. in 2018, 2 x 10^{34} $cm^{-2}s^{-1}$ (colloquially: 2e34)

try to keep *stable* throughout the **year**

[peak luminosity = peak of the year, not each run!]

must account for all limitations of the system - including target **average** physics output rate of 1 kHz

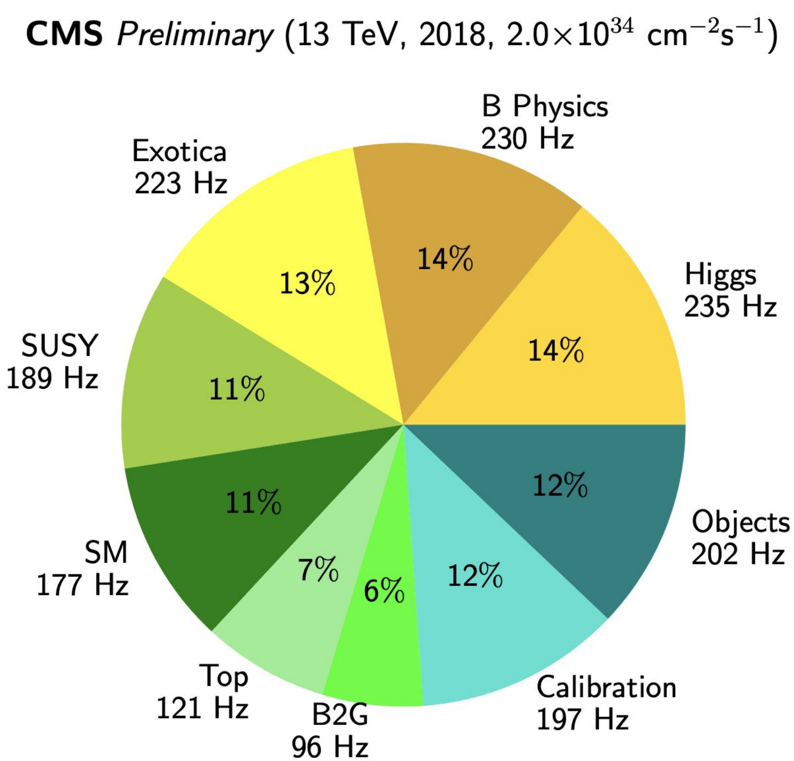
average = time-averaged over all nominal data-taking

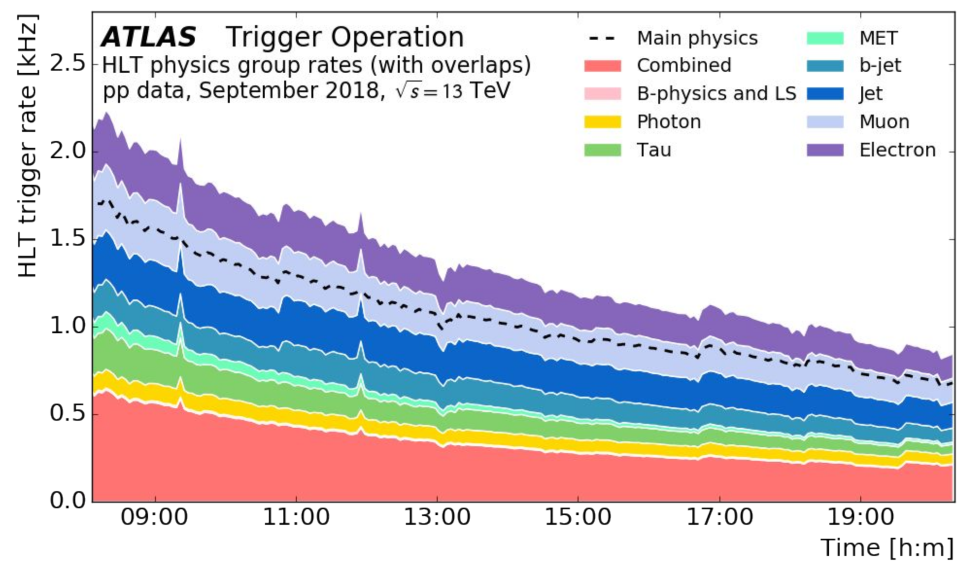


allocating rate

Trigger strategy is driven by the physics priorities of the experiment

- and as we are multi-purpose experiments, physics priorities span a broad range of signatures
- highest rates are the isolated single muon/electron triggers

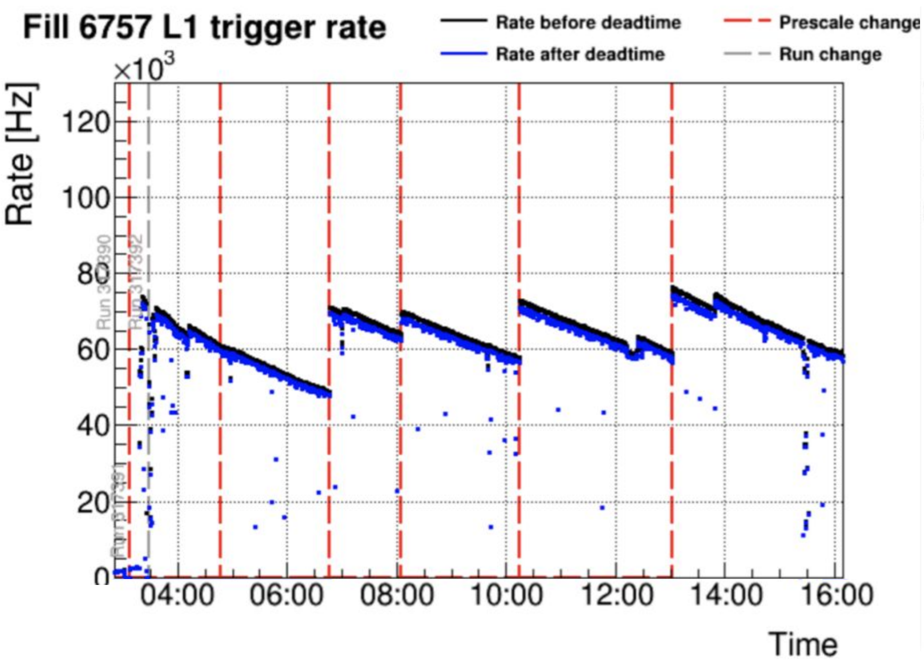


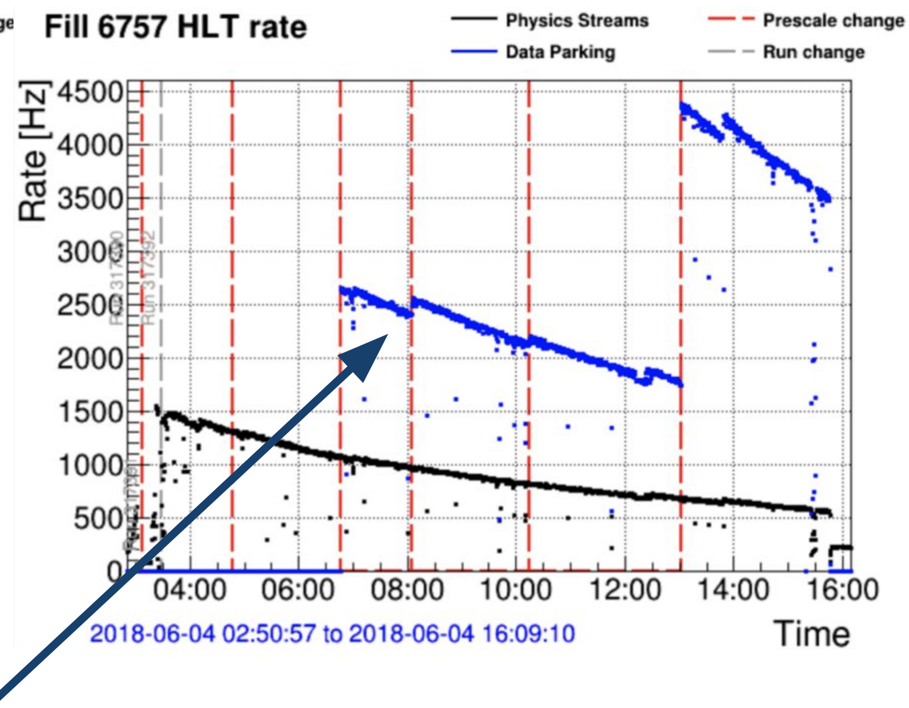


what if the rate *doesn't* fit?

(1) wait for the data (*delayed* reconstruction from *parked* data)

-





if the limitation is tier0 reconstruction

L1 rate ok, HLT readout ok, have sufficient tape storage, and know we'll be able to process at a later date

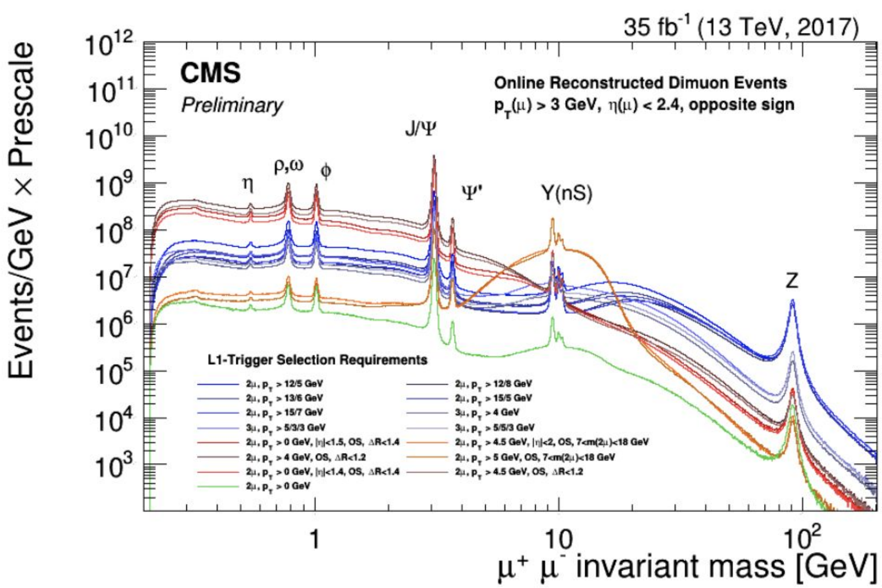
what if the rate *doesn't* fit?

(1) wait for the data (*delayed* reconstruction from *parked* data)

(2) make the data smaller

  (a) data scouting/trigger-level analysis

  only read out the HLT object data - no raw detector data, no offline reconstruction



e.g. from CMS:
low-pT dimuon events

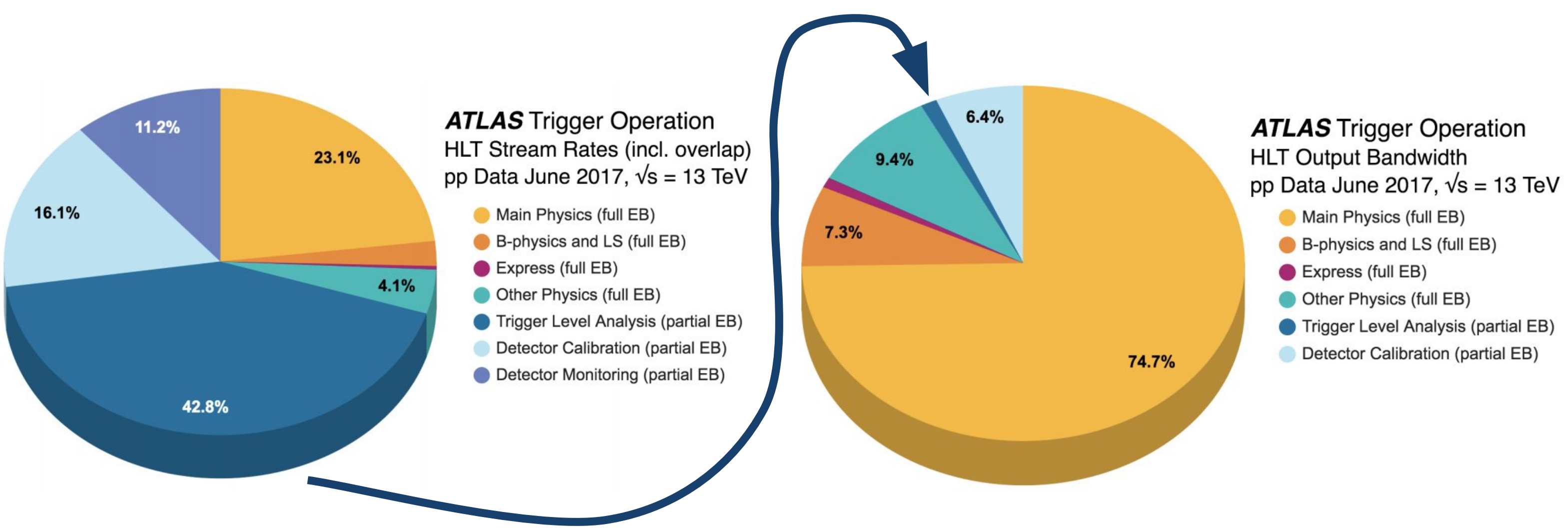
what if the rate *doesn't* fit?

(1) wait for the data (*delayed* reconstruction from *parked* data)

(2) make the data smaller

  (a) data scouting/trigger-level analysis

  only read out the HLT object data - no raw detector data, no offline reconstruction



43% of output rate ⇒ ~1% of output bandwith

what if the rate *doesn't* fit?

(1) wait for the data (*delayed* reconstruction from *parked* data)

(2) make the data smaller

(a) data scouting/trigger-level analysis

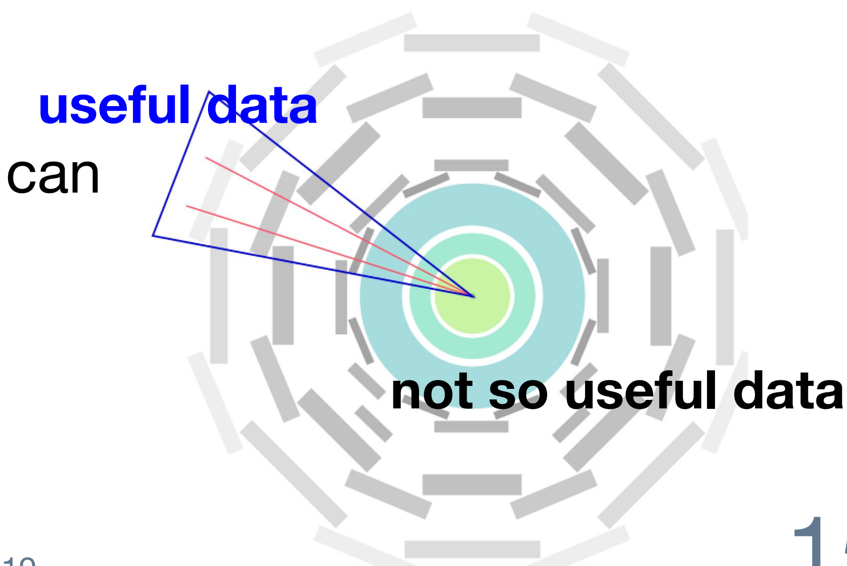
only read out the HLT object data - no raw detector data, no offline reconstruction

(b) partial event building

only save parts of the raw data (certain subdetectors or certain regions) - not full events

limited use for physics analyses, but can be useful for supporting triggers

e.g. data for online and offline efficiencies from events with collimated topologies



what if the rate *doesn't* fit?

(1) wait for the data (*delayed* reconstruction from *parked* data)

(2) make the data smaller

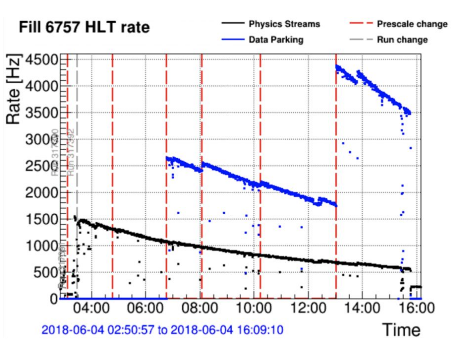
  (a) data scouting/trigger-level analysis

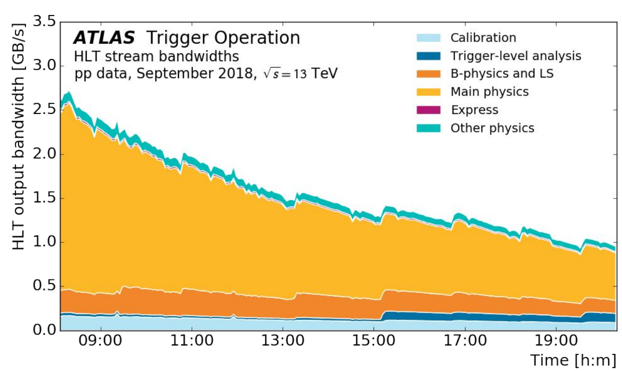
    only read out the HLT object data - no raw detector data, no offline reconstruction

  (b) partial event building

    only save parts of the raw data (certain subdetectors or certain regions) - not full events

(3) record data only at luminosities below the peak luminosity (i.e. at the *end of fill)*





ok, i have my trigger. we're done now, right?

monitoring

*dq = data quality

Online rate monitoring | **Online event monitoring** | **Offline monitoring**

CMS Roles
both
ATLAS Roles

L1 experts

rest of shift crew

L1 oncall expert

Prompt feedback expert

Trigger expert analyst

Trigger shifter

HLT oncall expert

Secondary HLT oncall

L1 experts

menu oncall expert

DQ/debug oncall expert

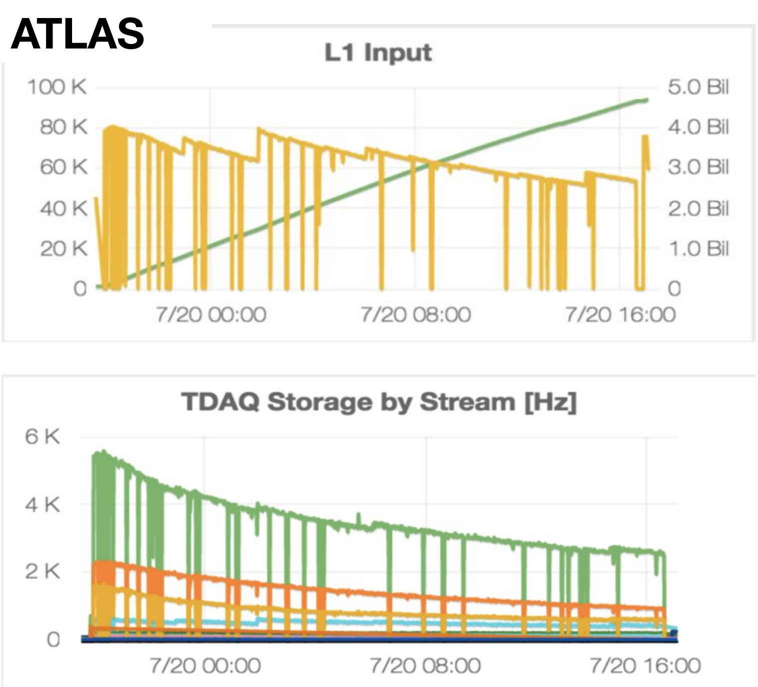
signature (HLT) DQ experts

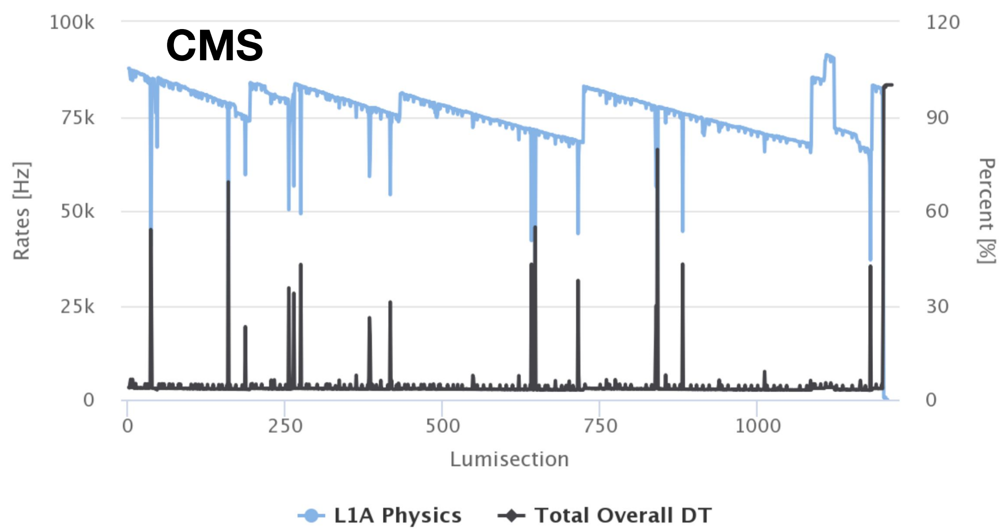
rate monitoring

most problems can be quickly seen in the rate of a trigger

(as long as you have a good reference!)

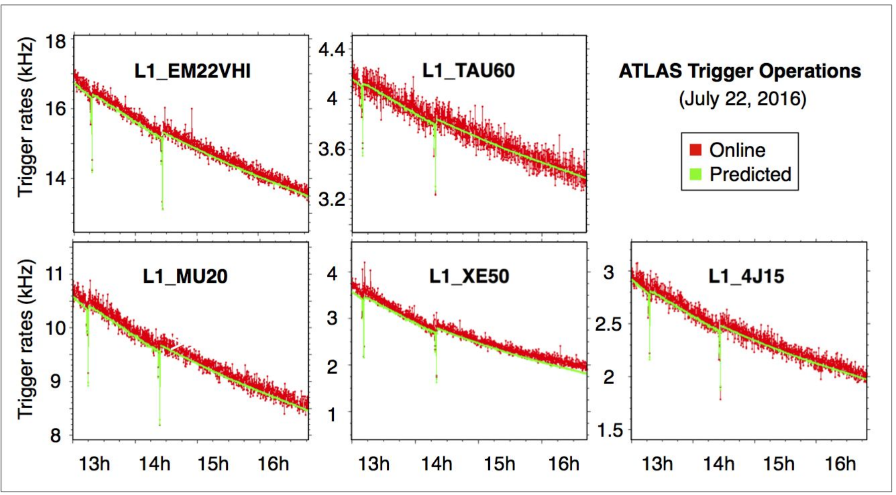
both ATLAS and CMS live monitor L1 and HLT rates

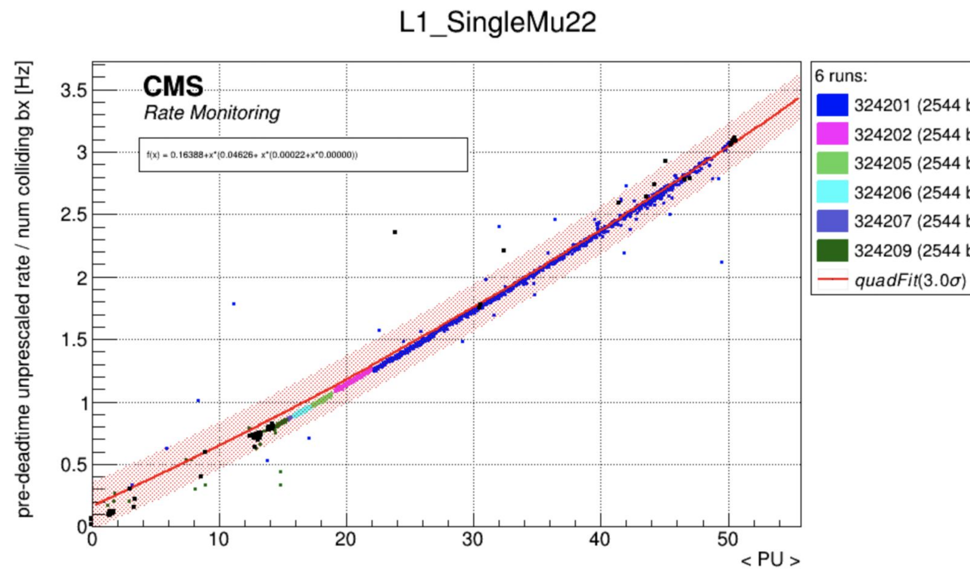




rate monitoring

L1 and HLT trigger rates are also monitored against **expected** rates





Monitored by trigger shifter in the control room
Problems must be actively followed up by the shifter
Fit is taken from rate @ start of run but with specified shape
Any deviations are subjective

Monitored by oncall experts
Fit is from a previous ‘good’ run
Problems are automatically flagged and shifter is notified to contact expert
Deviations are objective (based on fit uncertainty)

additional online monitoring

What happens if the rates deviate?

or what if the rate is not affected, but the events are?

ATLAS

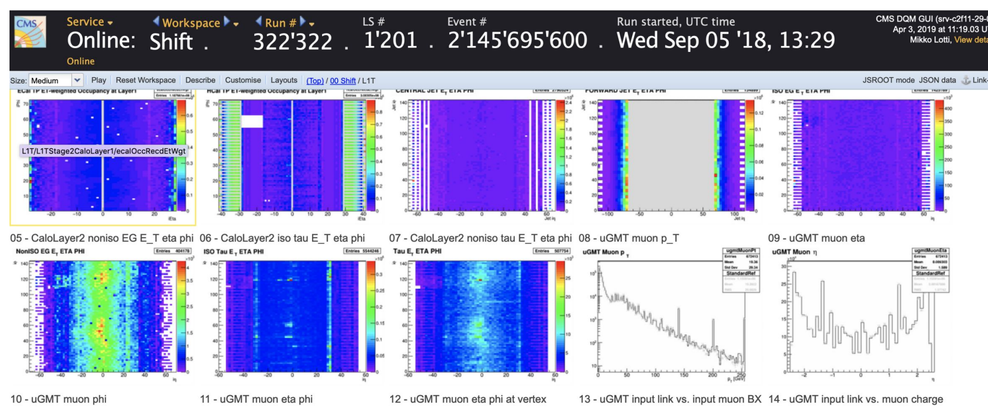
Trigger shifter checks histograms filled by trigger algorithms' monitoring for every event

flagged **red**, **yellow** or **green**

based on user-defined algorithm result, compared to validated reference

CMS

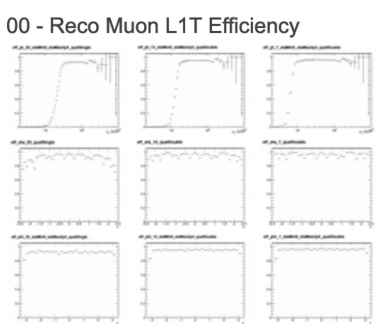
Trigger shifter checks histograms filled by L1 trigger information, comparing to reference [can have user-defined algorithms as well]

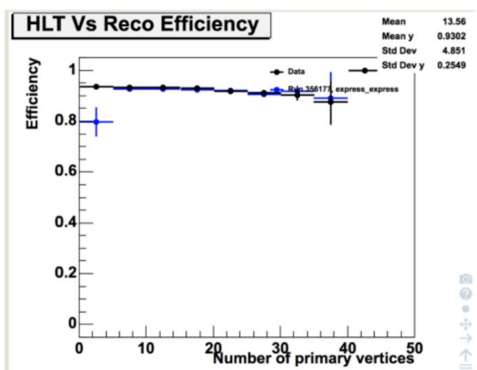


offline monitoring

Select the high-quality data for physics analyses

- initial sign-off performed with small dataset that is processed rapidly, designed for data quality checks
  - [**express** stream - 20 Hz ATLAS, 50 - 100 Hz CMS]
- further signoff on full physics dataset





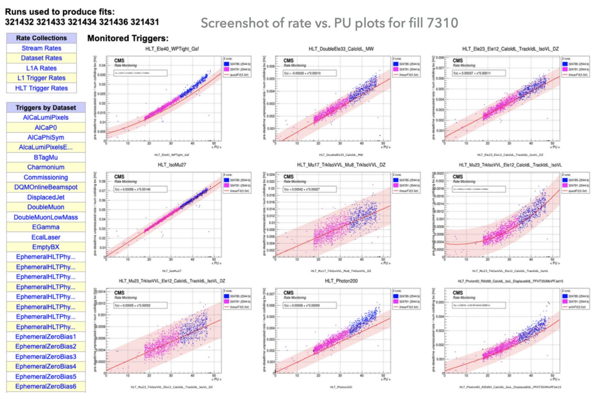
both ATLAS and CMS validate data using detailed DQM plots for L1 and HLT

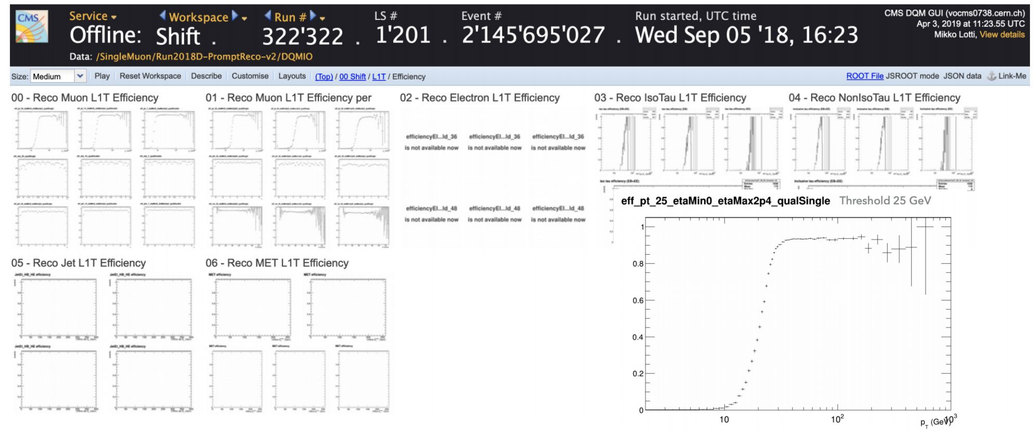
offline monitoring

- Select the high-quality data for physics analyses

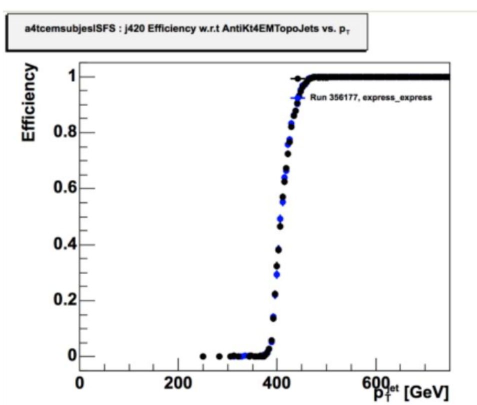
e.g. turn-on curves, properties of selected events, detailed rate comparisons:

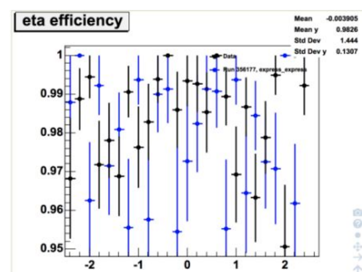
CMS

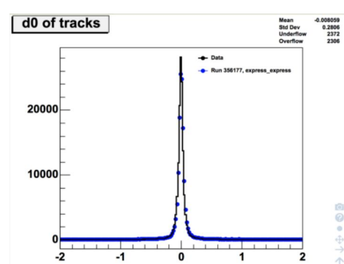


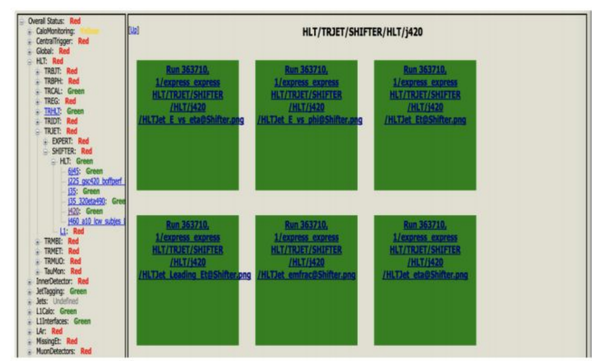


ATLAS





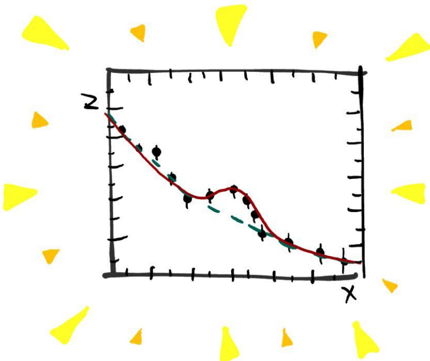




summary

there is no one correct answer to how to construct and monitor a physics menu

- many differences and many similarities between the ATLAS and CMS strategies
- but we both successfully recorded ~150/fb of p-p data
  - *not mentioned: many other special configurations, also with their own dedicated trigger menus!*
- we are learning from each other - two fruitful cross-talks in January and April



useful links

most plots shown in this talk were from presentations at the two cross-talks:

ATLAS-CMS chat 1: https://indico.cern.ch/event/770403/

ATLAS-CMS chat 2: https://indico.cern.ch/event/803880/

ATLAS public results: https://twiki.cern.ch/twiki/bin/view/AtlasPublic/TriggerPublicResults

some relevant publications:

- Performance of the ATLAS Trigger System in 2015 [EPJC 77 (2017) 317]
- Trigger Menu PUB notes highlighting new features, breakdown of main triggers/rate and overall performance released after data taking each year:
  - Trigger Menu in 2015 (ATL-DAQ-PUB-2016-001)
  - Trigger Menu in 2016 (ATL-DAQ-PUB-2017-001)
  - Trigger Menu in 2017 (ATL-DAQ-PUB-2018-002)
  - Trigger Menu in 2018 (in preparation)

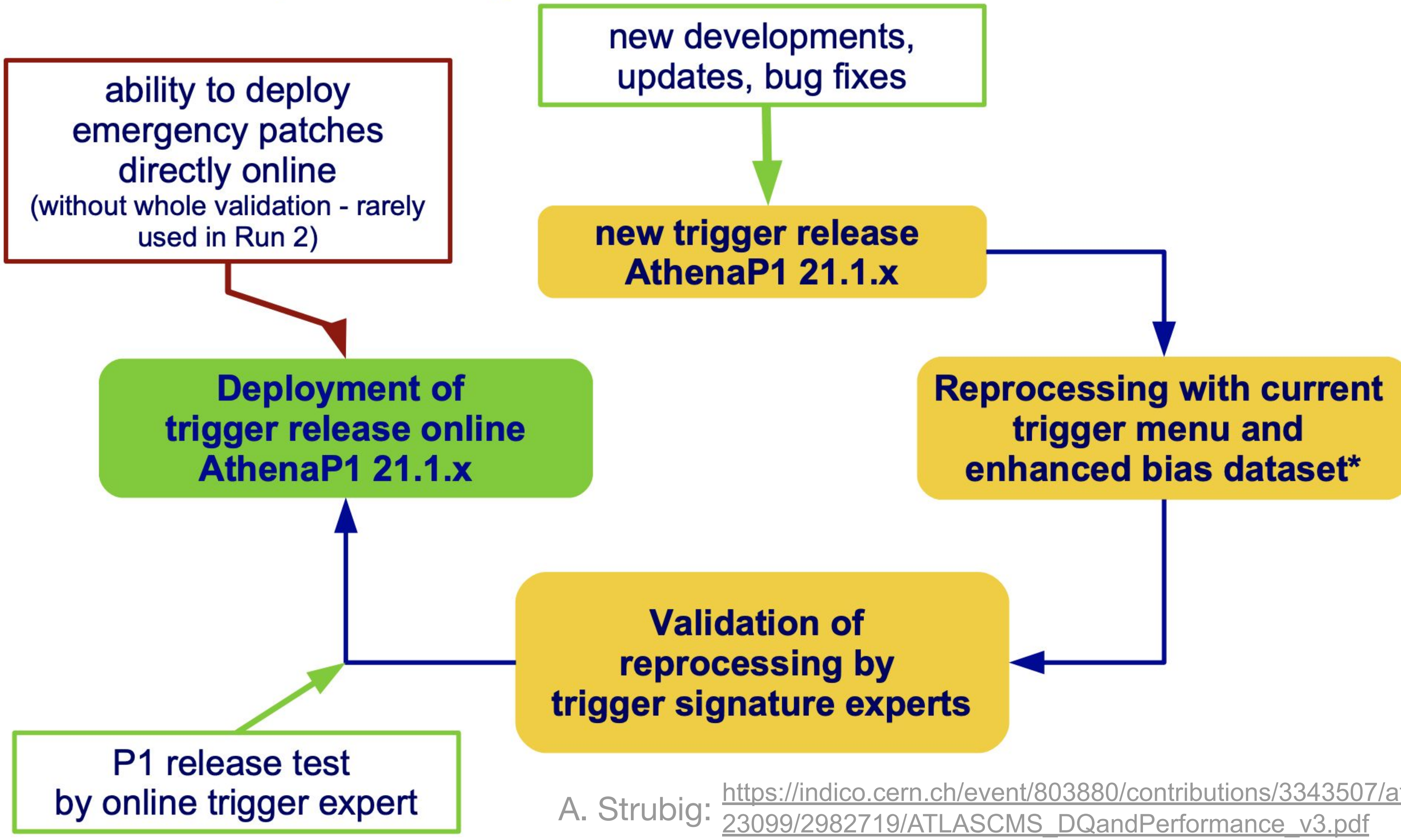
CMS public HLT results: https://twiki.cern.ch/twiki/bin/view/CMSPublic/HighLevelTriggerRunIIResults

CMS public L1 results: https://twiki.cern.ch/twiki/bin/view/CMSPublic/L1TriggerDPGResults

release validation - ATLAS

- New release can be deployed ~ every week if needed – in 2018 releases changed ~ bi-weekly

  * details of steps on following slides



A. Strubig: https://indico.cern.ch/event/803880/contributions/3343507/attachments/1823099/2982719/ATLASCMS_DQandPerformance_v3.pdf

release validation - CMS

RELEASE VALIDATION

M. Lotti: https://indico.cern.ch/event/803880/contributions/3343508/attachments/1823163/2983019/ATLAS_CMS_X_TALK_03042019_Lotti.pdf

- Releases validation is used to monitor changes in software and is part of the TEA workflow
- To validate different releases RelMon automatic report page is used
  - Done by comparing the relVal DQM plots
  - Goal is to understand reasons behind possible changes in the performance plots
  - Shifters submit Jira tickets about the validation and report the differences
    - Experts then help to locate the origin of problems