Infrastructure for deployment and evaluation of LHCb Trigger Configurations

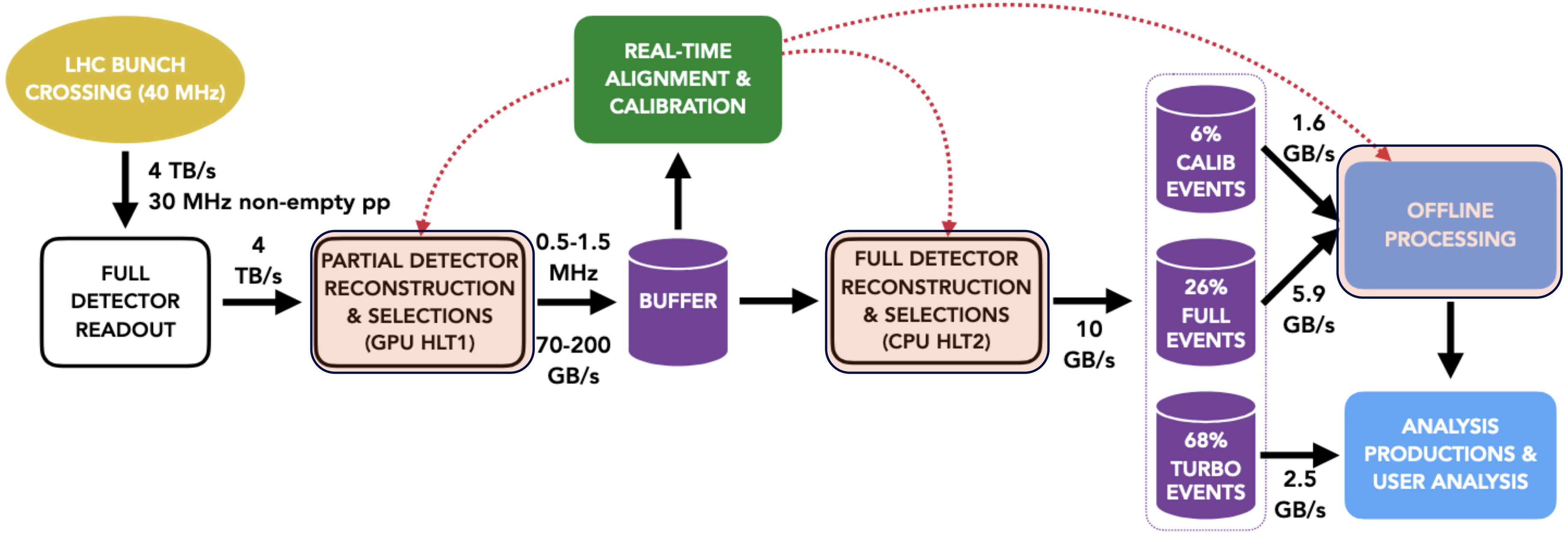
Rehearsal for CHEP24

Luke Grazette[1], Micol Olocco[2], Rosen Matev[3]
on behalf of the LHCb collaboration

1, University of Warwick; 2, Technische Universität Dortmund; 3, CERN

The LHCb Upgrade Trigger

The LHCb Trigger applications* consist of components that run reconstruction algorithms and perform physics object selections.
The number of these components scale from 100s to 10,000s depending on selection stage.



The LHCb Upgrade Dataflow.
The selection stages are highlighted.

All numbers are taken from the LHCb Upgrade Trigger and Online TDR and the LHCb Upgrade Computing Model TDR

*technically this also applies to LHCb Upgrade's Sprucing stage, which is not strictly a "Trigger" but instead a selection stage during the offline processing. For this talk Trigger includes all 3 selections stages, the two high-level Triggers (HLT1, HLT2) and Sprucing.

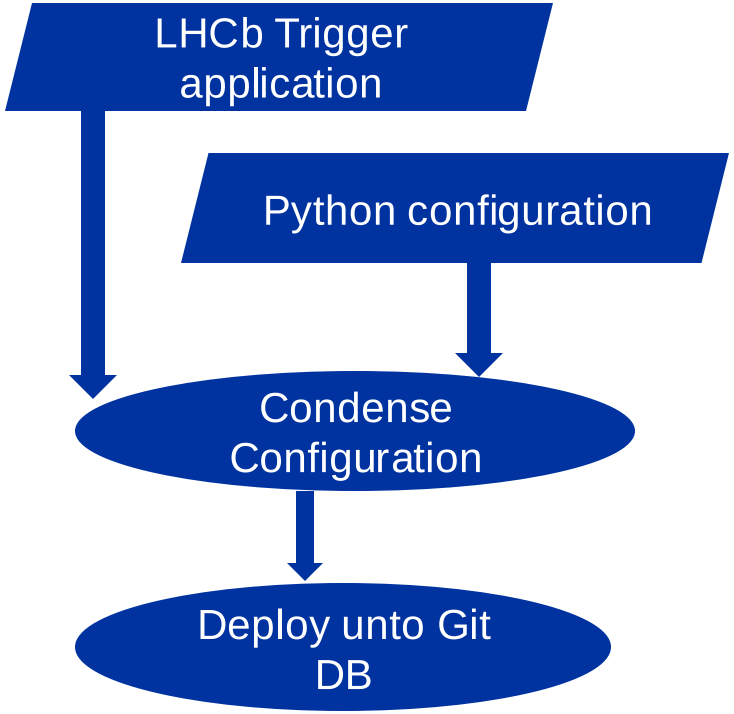
Configuring a Trigger

Data flow, control flow and parameters of the selections for Trigger components are configured via `Python`:

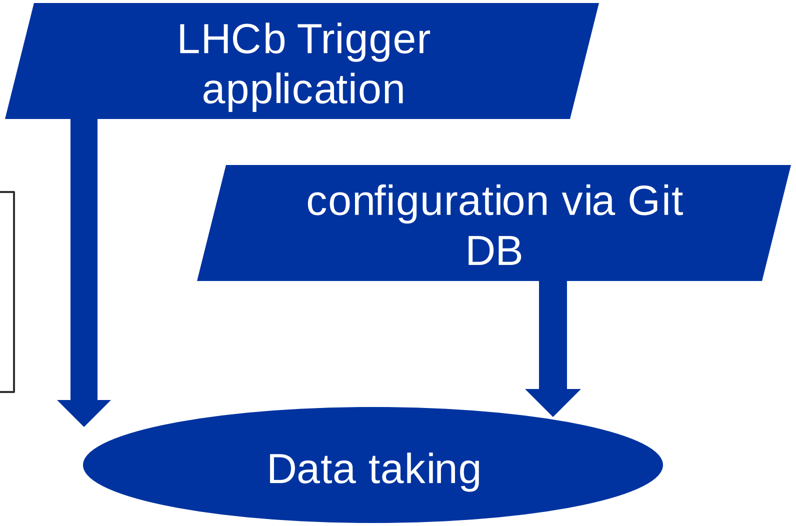
- possibility to change the Trigger without deploying new binaries (compiled components).

As this is used for data taking, it is essential to **reproduce a given production configuration** and query it:

- **condense** the configuration into a basic form
- **store** it within a Git database
- **access** Git database for data-taking.



Sketch of the process to deploy a configuration



Sketch of accessing the deployed configuration for data-taking.

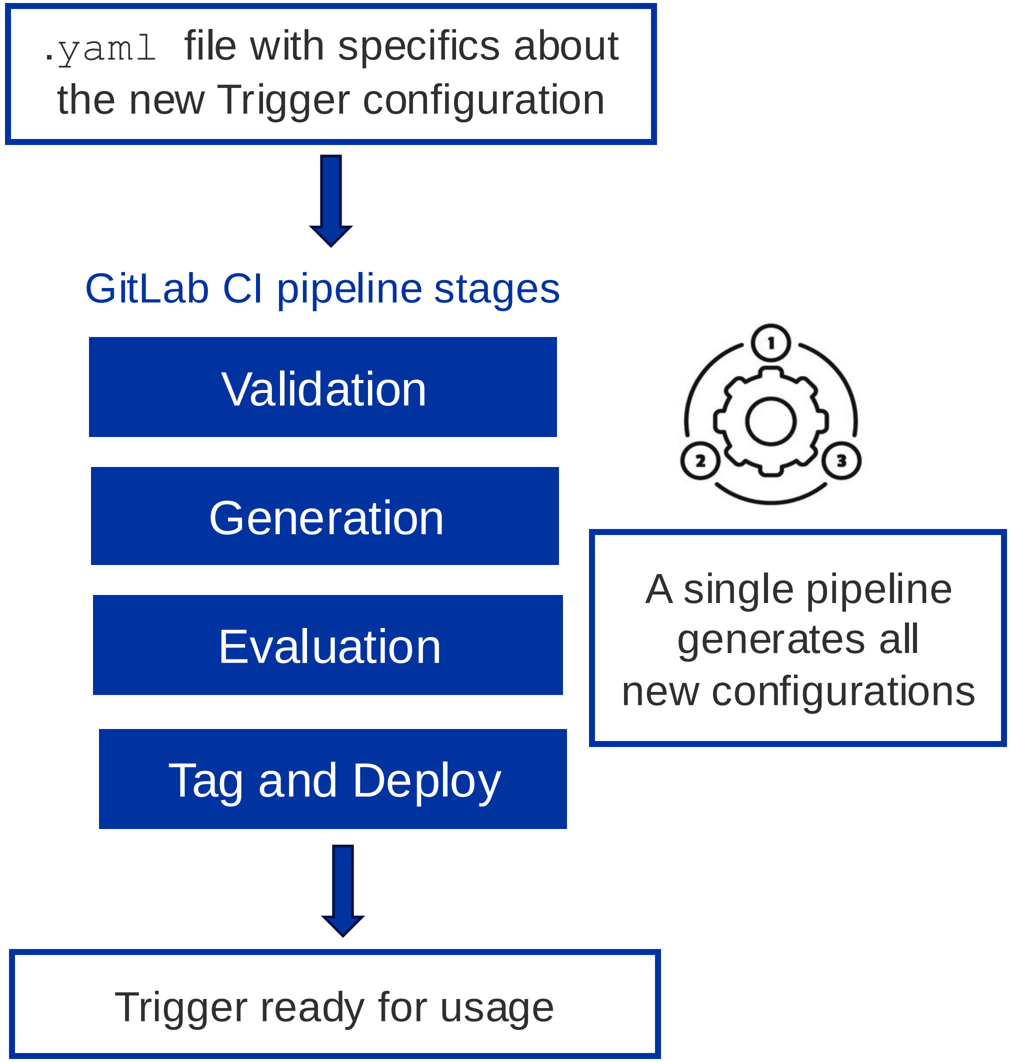
Creating the configuration

Old procedure

***Manual* generation** of the Trigger configurations via a list of detailed instructions to be followed for every new configuration.

New procedure

An **infrastructure** for *generating* and *validating* the configurations using GitLab Continuous-Integration pipelines.
This automation ensures ***consistency*** and ***reproducibility*** of the generated configurations, by providing the benefits of ***validation***, ***evaluation*** and ***deployment*** for the generated configurations.



New Workflow for creating a Trigger configuration

Automation: Generation

Generation of the Trigger configuration requires the compiled binary of the LHCb Trigger applications.

Released binaries on CernVM-FS are used.
Sometimes necessary to generate within pipeline.

Compiling can be **time consuming and computationally expensive.**

- the LHCb Trigger application stack reduces unnecessary repeated compilation by using a `ccache`.

Using GitLab's cache functionality for CI tests, the `ccache` is propagated between pipelines.

- Preventing redundant compilation.
- Reducing time and computing resources being wasted.

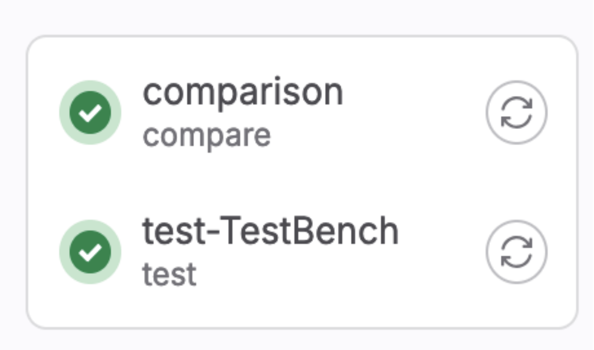
```yaml
generation:
  stage: generate
  allow_failure: false
  artifacts:
    reports:
      dotenv: job.env
    paths:
      - tmp/*.log
      - tmp/*.txt
      - tmp/tck_repo
      - tmp/stack*
      - tmp/lhcb-metainfo
    expire_in: 2 mos
  cache:
    key: ""
    paths:
      - tmp/stack/.ccache/
  script:
```

A snippet from the gitlab-ci.yml, using the GitLab cache feature to reduce duplicated compilation.

Automation: Validations

Before a trigger configuration is deployed, it must pass checks on:

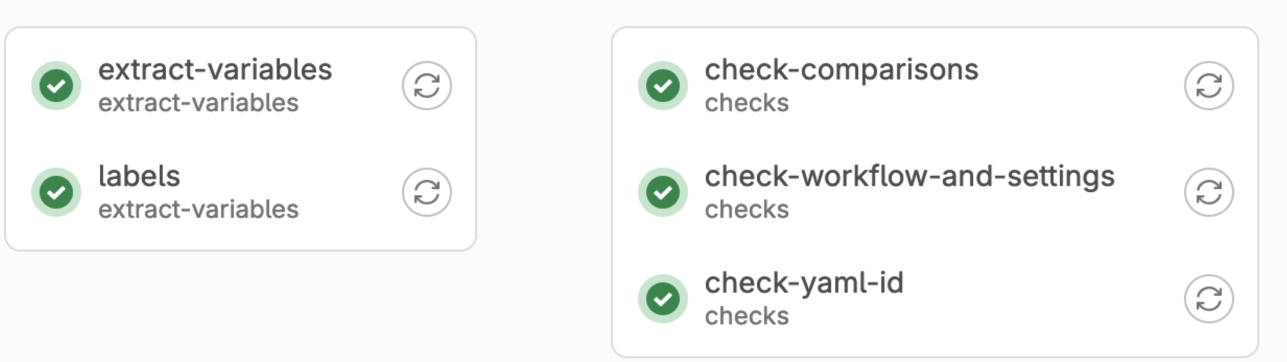
- **Comparison** of the trigger configurations and **performance evaluation** of the resultant trigger.
- **Validation** of the provided specification(s), checking for misconfigurations. *(pre-generation)*



GitLab CI jobs for Evaluation of Trigger configurations

```
"projects": {
    "Allen": "4.12",
    "Detector": "1.35",
    "Allen": "4.13",
    "Detector": "1.36",
    "Gaudi": "38.1",
    "LCG": "105a",
    "LHCb": "55.12",
    "Lbcom": "35.12",
    "Rec": "36.12"
    "LHCb": "55.13",
    "Lbcom": "35.13",
    "Rec": "36.13"
}
```

Snippet of the output of a ***comparison*** job between two configurations



Gitlab CI jobs for validation of Trigger configuration specifications provided by the user

```
STARTING SUMMARY OF JOB
------------------------------------------------------------------
SUCCESS:: v4r13 is an ancestor of f8a9e7af74c25ae2eabce02120ab78857bff59b6.
SUCCESS:: Verified that only difference between v4r13 and f8a9e7af74c25ae2eabce02120ab78857bff5
9b6 are python files. Suitable for use.
SUCCESS:: Verified that https://gitlab.cern.ch/lhcb/Allen/-/raw/f8a9e7af74c25ae2eabce02120ab788
57bff59b6/configuration/python/AllenSequences/hlt1_pp_forward_then_matching_1000KHz.py exists f
or configuration
------------------------------------------------------------------
SUCCESS :: workflow and settings are compatible.
FINISHED SUMMARY OF JOB
```

Snippet of the output of a ***check-workflow-and-settings*** job, verifying that the Trigger sequence exists for a provided configuration.

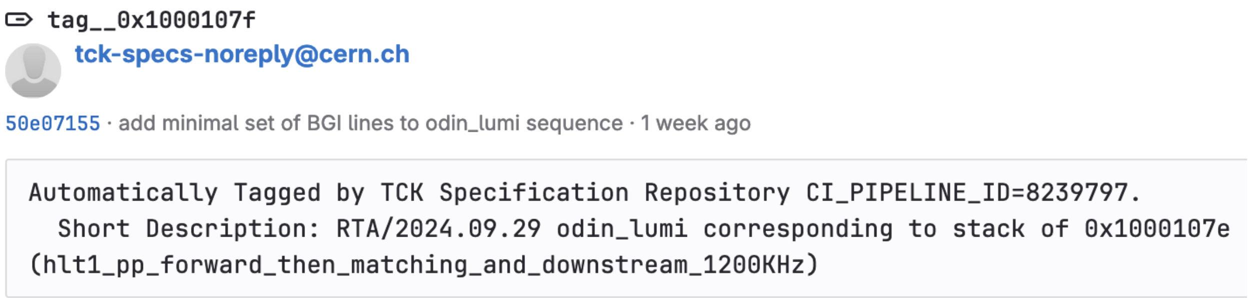
Automation: Deployment

Upon merging the request for new trigger configurations, the pipeline (on the `main` branch) includes extra stages for tagging and deployment of the newly generated trigger configurations, such as:

- new configuration(s) in the Git database
- associated encoding/decoding key(s)
- tag(s) for reproducibility and posterity

These are available immediately on the relevant Git database remotes.
Approximately 10 minutes later on CernVM-FS for use.

GitLab CI jobs for Tag and Deployment of newly generated configurations



A tag related to a recently generated configuration, filled with the relevant brief information and the details to find more comprehensive information if necessary.

Benefits of the new infrastructure

tck/0x1000109b.yaml 0 → 100644 +11 −0 Viewed


```yaml
TCK: 0x1000108b
workflow: "commit-based"
parameters:
    process: "Hlt1"
    type: "odin_lumi"
    label: "RTA/2024.09.29.2 with odin_lumi for emittance scan"
    stack: "RTA/2024.09.29.2"
    settings: "odin_lumi"
    commits:
      Allen: "255b6528f6ce53a0ffe7fa578beffc00e25b40bd"
    comparison: 0x10001080
```

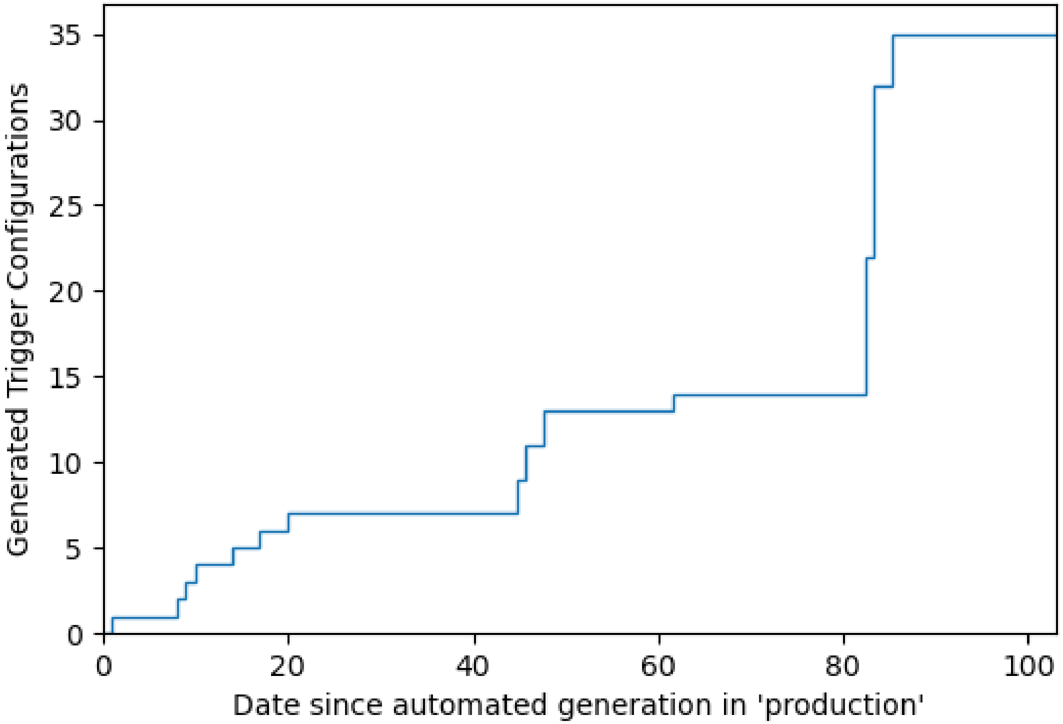

A specification for a requested trigger configuration

The LHCb Trigger operators, together with expert supervision, make sure to adapt the trigger to the needs and conditions of the data-taking.

A large benefit from having this automated infrastructure is that it is much easier for the operators to generate trigger configurations.

➡ **Trigger Operators have more ability to focus on the "higher level" information:**

- Physics changes needed for the Trigger.
  - Informs the trigger configuration request.
- Evaluations of the new triggers' performance metrics.
- Responding to other data-taking demands.



Plot showing total trigger configurations generated since the new automated procedure

Future Prospects

Reminder: the Trigger configurations (detailing the control flow, dataflow and parameters of selections) **are stored in simplified state in a Git database.**

This format is intentionally **'query-able'**, allowing the creation of an **interface and/or webpage** (like the LHCb Stripping pages, a Legacy project) to:

- **Probe** in-detail a specific configuration
- **Compare** changes between configurations
- **Understand** the dependencies of specific selections

This would be of particular benefit to ***analysts***, aiding understanding of how the LHCb's Trigger changed during data-taking.

StrippingWMuLine

Properties:

OutputLocation	Phys/WMuLine/Particles
Author	S. Bifani
Postscale	1.0000000
HLT1	None
HLT2	None
Prescale	1.0000000
L0DU	None
ODIN	None

Filter sequence:

LoKi::VoidFilter/StrippingGoodEventConditionEW
CheckPV/checkPVmin0
LoKi::VoidFilter/SelFilterPhys_StdAllLooseMuons_Particles
FilterDesktop/WMuLine

Main properties:

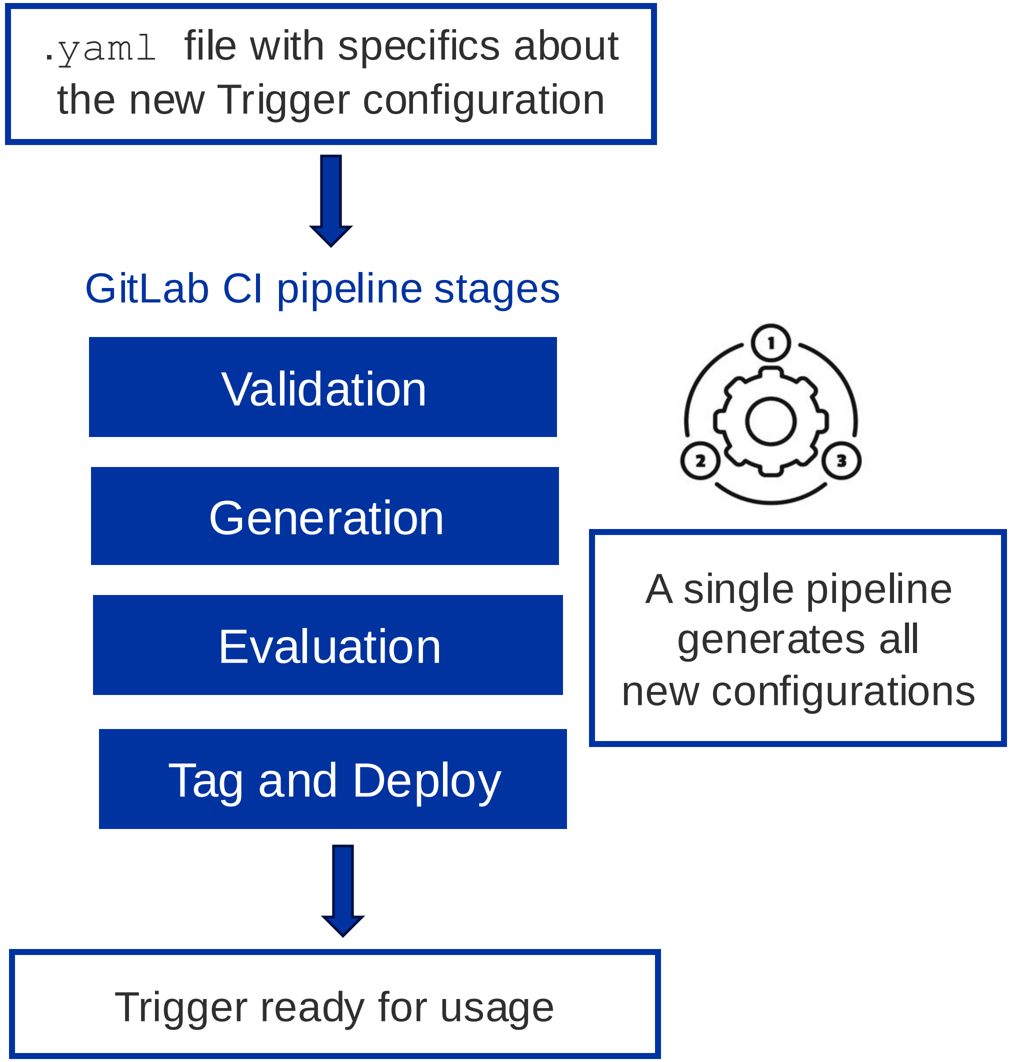
CloneFilteredParticles	True
Code	(PT > 20000.0)
DecayDescriptor	None
ForceOutput	True
IgnoreP2PVFromInputLocations	False

An example of the LHCb Stripping pages, for a single line and configuration

Summary

- The LHCb Triggers are compiled algorithms that are configured via `python`. These configurations are captured and stored for reference and usage.

- An **automated workflow** via GitLab is now used to generate configurations in a **consistent and reproducible** way, reducing both manual and technical burdens on trigger operators.
  - It provides validations, generation, evaluations and deployment for new trigger configurations.
  - The query-able nature of the trigger configurations allows promising future improvements to aid the LHCb Collaboration's long-term understanding of our Triggers and data-taking.



New Workflow for creating a Trigger configuration

Thanks for Listening

Luke Grazette[1], Micol Olocco[2], Rosen Matev[3]
on behalf of the LHCb collaboration

1, University of Warwick; 2, Technische Universität Dortmund; 3, CERN