The ATLAS Trigger Simulation with Legacy Software

Catrin Bernius, SLAC,
on behalf of the ATLAS Collaboration

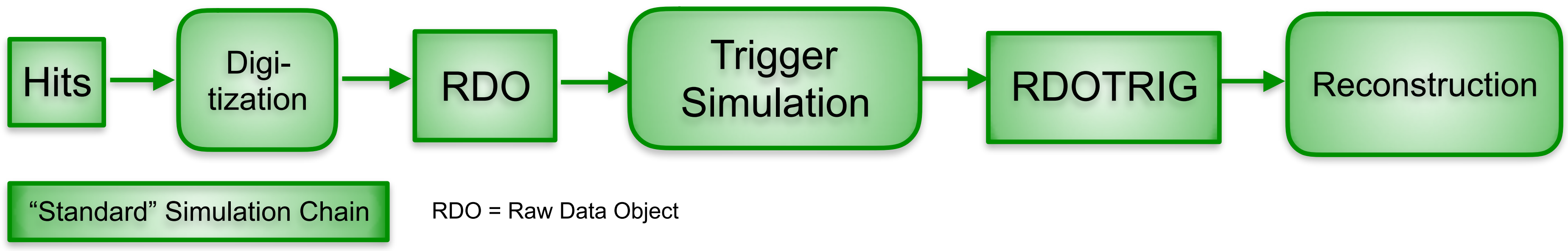
SLAC NATIONAL ACCELERATOR LABORATORY

The accurate simulation of the detector response and event selection processes is important for physics analyses at the LHC. In particular the **trigger response simulation** is crucial for determining the overall selection efficiencies and signal sensitivities and should be done **with the same software release with which data were recorded.** This requires potentially running with software dating many years back, so-called legacy software. Having a **strategy for running legacy software in a modern environment** is therefore essential when data simulated for past years start to present a sizeable fraction of the total.

Motivation

- Capability to produce new simulated data for all data taking periods needs to be maintained
- Latest software is used for event generation and reconstruction, simulated trigger response needs to be in agreement with the simulated data taking period → Original trigger algorithms and selections from the period are applied
- **Legacy Simulation**: Use the trigger software and conditions data that match the simulated data-taking period dating potentially many years back

Simplified ATLAS Simulation Chain



ATLAS "Standard" Simulation

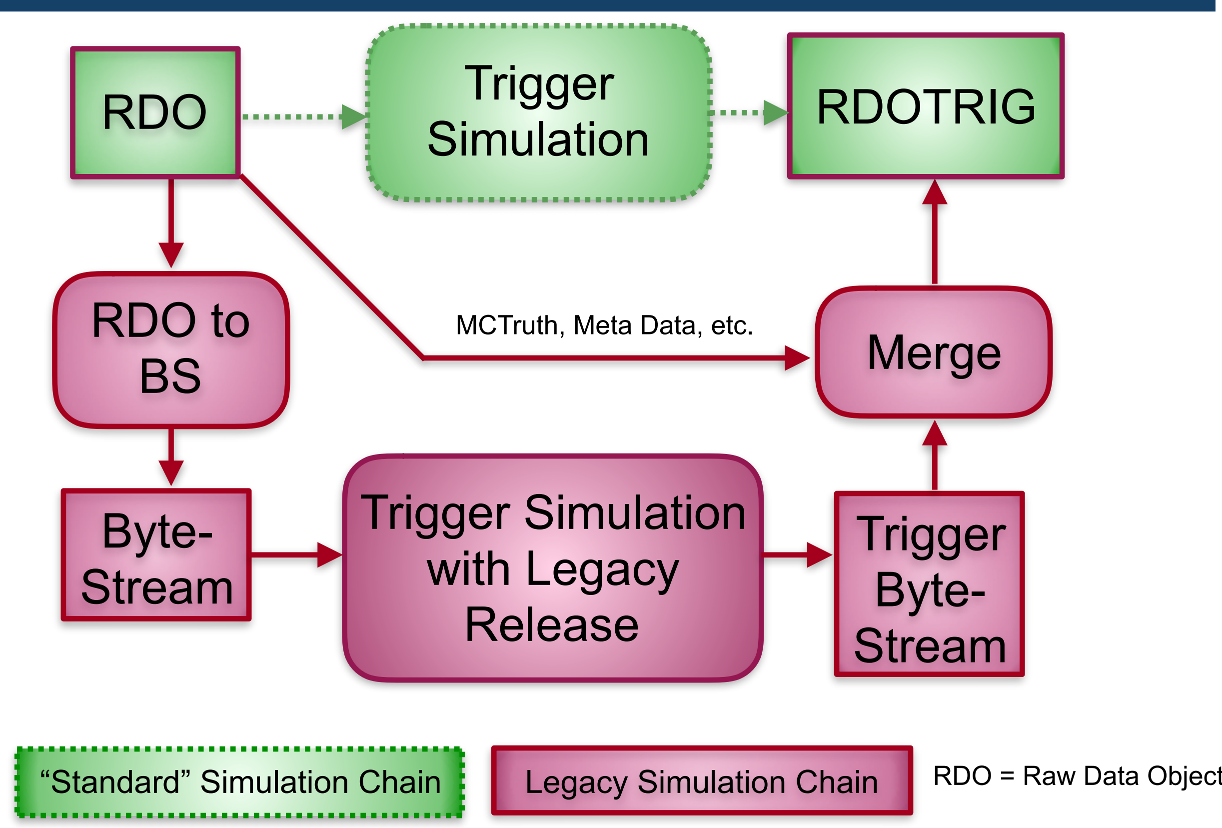
- **One software release for all simulation steps**
- Event reconstruction and detector simulation reflect the best current knowledge about the detector, use most recent software developments, geometry and conditions
- **Data exchange** between simulation steps uses **Raw Data Object (RDO) format**
- Simulation of trigger response has to emulate the online selection of the respective data taking period

Options for Trigger Simulation

- **Porting old trigger selection code to new simulation release**
  - Requires maintenance and manpower efforts
  - Old selections, algorithms, configuration and conditions data to be kept operational along the most recent trigger selections
  - Conflicting requirements, knowledge preservations, maintenance of infrastructure services
- **Re-running legacy trigger selection code "as is" from old releases**
  - Reduces maintenance effort compared to first option
  - Focus on technical aspects e.g. creation and maintenance of environment for re-running legacy code

Legacy Trigger Simulation

- Split of the "standard" simulation chain into sub-steps to use **different releases for the trigger simulation**
- Requires the long term conservation of trigger software releases, configuration data and conditions data



Data Format Issues: "Raw Data Objects" (RDO)

- Container format based on ROOT technology
- **Payload format is community dictated**, may change with new releases
- Merge step to re-add MC truth information and data processing parameters (Merge → RDOTRIG)
- **Forward compatibility**: Output data from the newer detector simulation release need to be readable by the old trigger simulation release
- **Backward compatibility**: Output data from the old trigger simulation release need to be readable by the newer reconstruction release
- Release-dependent changes in the RDO format make it difficult to reach compatibility with many old trigger releases

Data Format: ByteStream (BS)

- Input/output format for trigger simulation module
- Container format based on uint32 arrays, tightly coupled to **detector readout hardware**
- **Forward compatibility**: detector simulation releases need to be able to convert the detector raw data to BS format with a given payload version
- **Backward compatibility guaranteed**: All ATLAS event reconstruction releases required to read BS data from all important data-taking periods
- Simple structure for payload data

Use of Virtualization - An Outlook

- **Medium term**: use older releases on new operating systems with compatibility libraries or in containers (Docker, etc.)
- **Long term**: Impossible to use legacy code "as is" due to
  - Introduction of new computing hardware technologies
  - Operating system changes
  - Changes to compiler and core libraries

  → Use **virtualization to preserve the complete runtime and development environment**
- Hardware abstraction and preservation of software environment
- Introduces computational and resource overhead
- Need to foresee patch releases to adapt to changing external infrastructure services → e.g. for data input/output services or for changing database technologies

ATLAS EXPERIMENT