MICE Event viewer status

MICE CM 45 July 2016

Mihailo Savic, Institute of physics Belgrade

Overview

- Event viewer is not one unique application due to multiple functionalities it is required to provide but rather utilizes few different interfaces that share the same core code
- Has two modes of operation: offline and online
- Here we will discuss:
  - Offline EV current status
  - Online EV current status
  - Some examples
  - Issues and ToDo list

Offline EV status

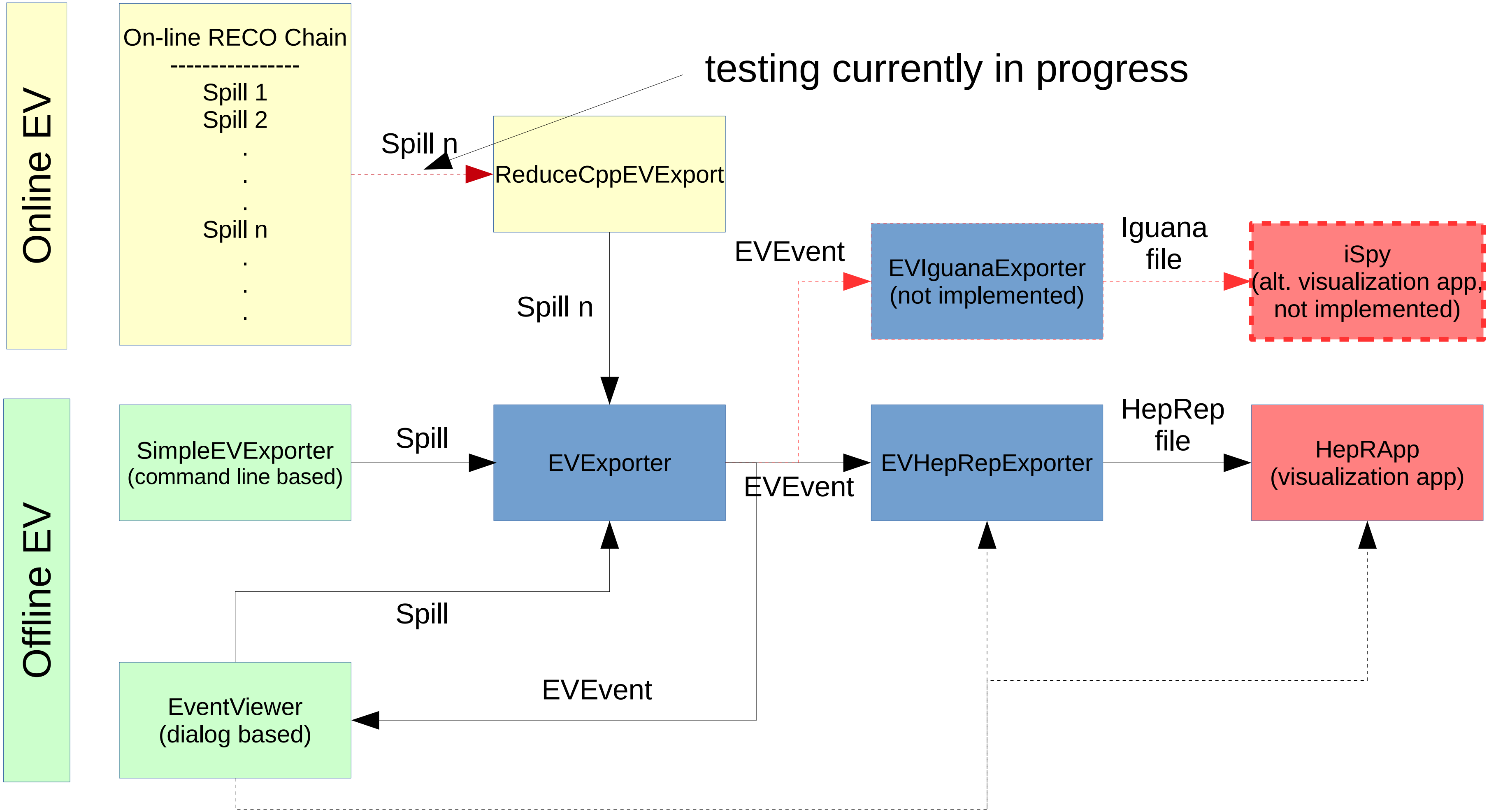
- Offline EV is designed to work with reconstructed root files and comes in two flavours:
  - Victoria's **Event viewer** (Qt dependent dialog based application). Displays MICE events in xz and yz planes (detector geometry not supported). Provides momentum and cluster info. Offers navigation through events and supports event selection (combination of detectors hit). Also, it is possible to export chosen events to heprep format and even display them (by running HepRApp) from within the application
  - **Simple EV exporter** (command line based). Exports MICE events to heprep format and combines them with detector geometry. They are then visualized by HepRApp, a java app developed at SLAC (JRE dependent).
  - Since MAUS version 2.4.0 Event viewer is distributed with MAUS (src/utilities/event-viewer) but is built separately
  - Most recent upgrade involves adding momentum (straight tracks) and cluster (space points) info to heprep output (committed to dev branch, not yet pushed or merged)

Online EV status

- Online EV is supposed to display MICE events in the control room as they are being reconstructed during data taking
- For this a cpp reducer needs to be implemented in the online reconstruction framework. This reducer would export all events from every nth (user defined) spill to heprep.
- HepRApp would run independently in autorefresh mode and iterate through events as they are being exported (every n seconds it displays next event)
- First version of the reducer (ReduceCppEVExport) is ready (dev branch) and has been tested against offline reconstruction chain
- Implementation in the online reconstruction framework is currently in progress
- To reduce the size of output heprep files new minimal detector geometry was stripped (MICEGeometryMinimal.heprep). For users that require greater detail a more detailed detector geometry is provided (MICEGeometryDetailed.heprep).

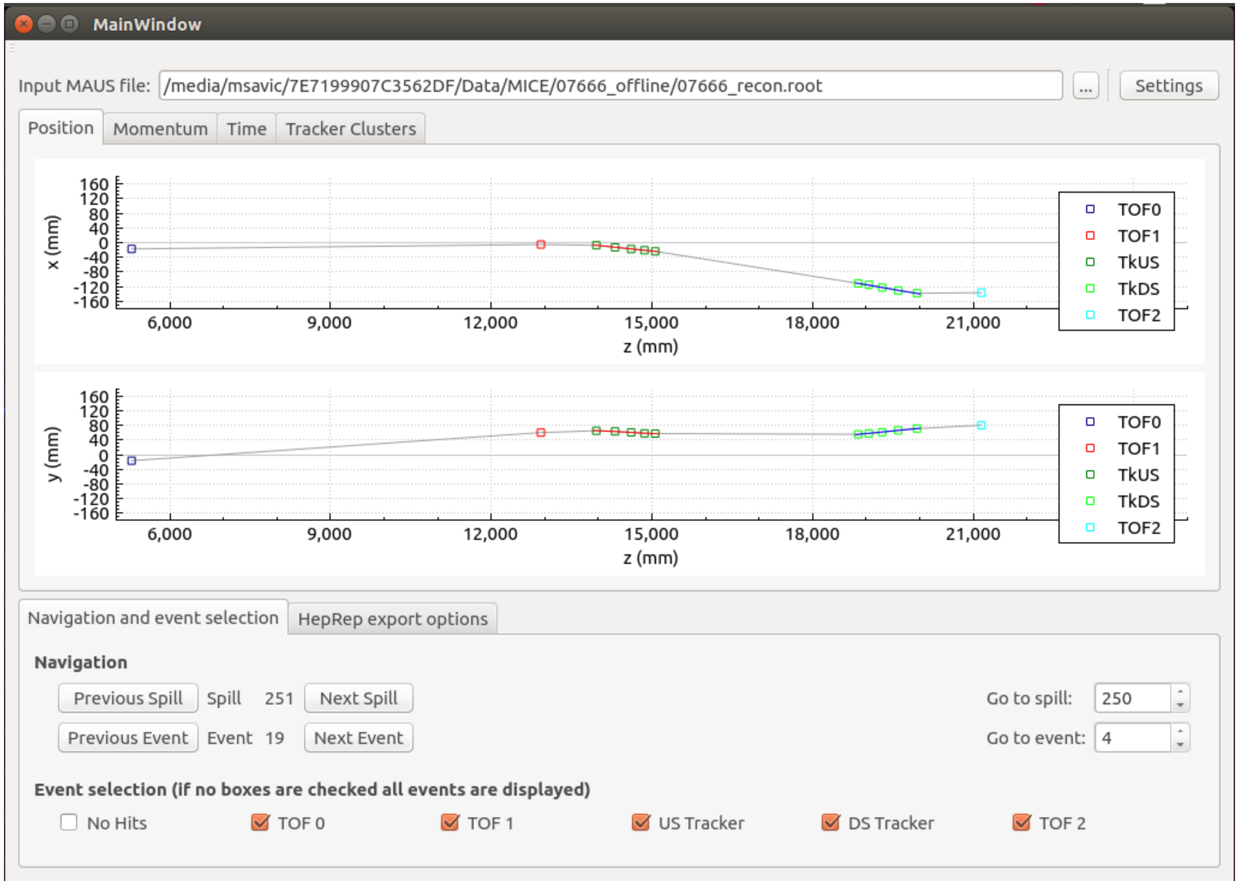
Online EV status

- Next figure illustrates relationship between EV structures (updated status):



Some examples

- Victoria's event viewer



Input MAUS file: /media/msavic/7E7199907C3562DF/Data/MICE/07666_offline/07666_recon.root

Navigation

Spill 251

Event 19

Go to spill: 250

Go to event: 4

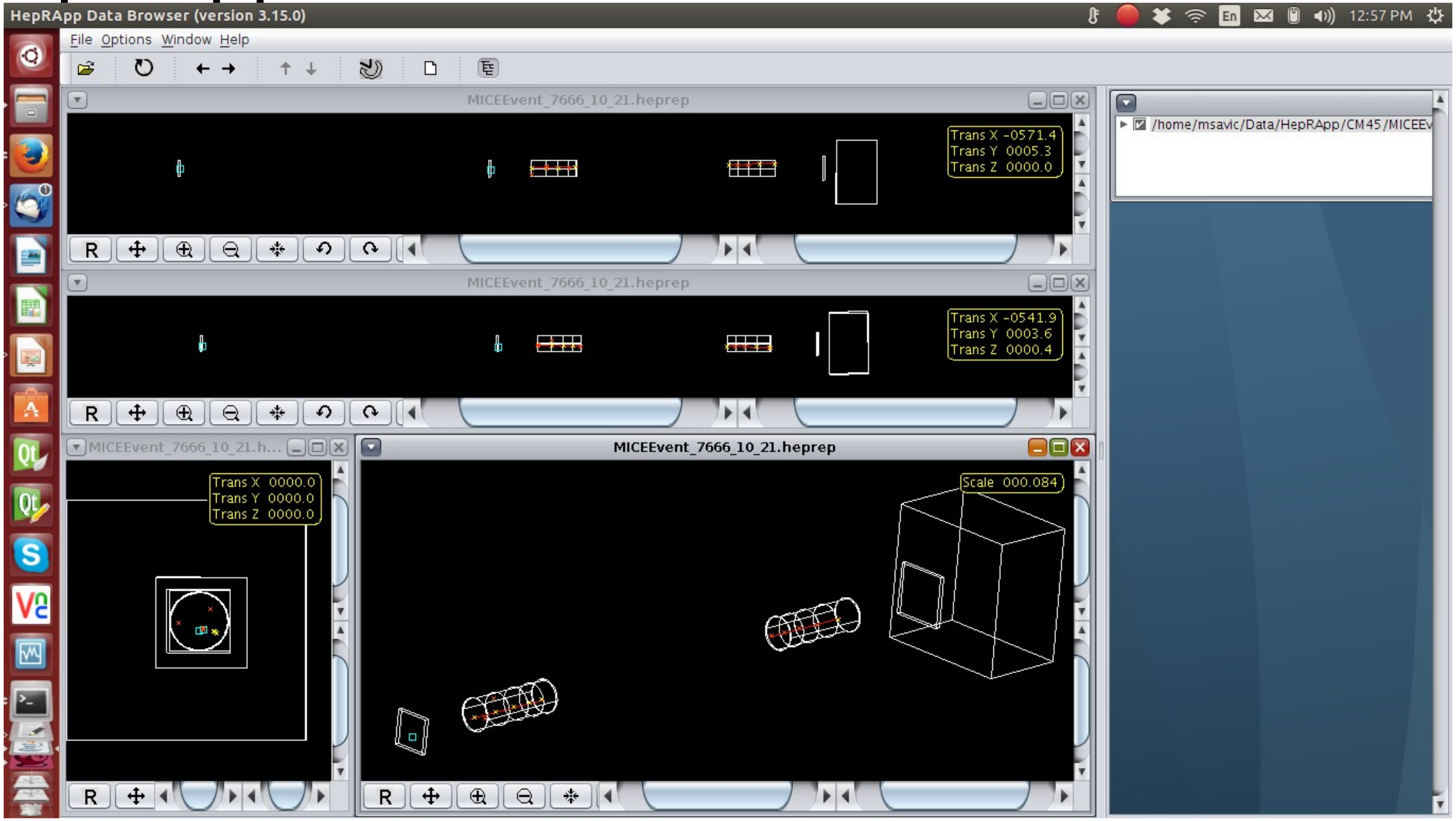
Event selection (If no boxes are checked all events are displayed)

☐ No Hits ☑ TOF 0 ☑ TOF 1 ☑ US Tracker ☑ DS Tracker ☑ TOF 2

Short demo video: https://youtu.be/3tQ4BDrJdPo

Some examples

- HepRApp



Short demo video: https://youtu.be/2uJnOze4Z-I

Issues and ToDo

- General
  - At the moment data that is being displayed includes TOF points, tracker space points (only in HepRApp), tracker reconstructed points and reconstructed straight tracks. EMR points need to be added. Any other data?
  - Helical tracks are not supported yet. In Victoria's event viewer 2D projections can be used. In HepRApp helical exists as a structure but is not parametrized. Polyline needs to be experimented with.
  - Truth MC data?
- Online EV
  - Since EV code is built independently of MAUS it is not possible to link EV to ReduceCppEVExport but code rather has to be deployed in the reducer directory. Adam generously offered to look into this
  - Even though new geometry is substantially smaller (~50kB), total size of exported files, providing every event (assuming 100 events per spill) from every 100th spill is exported, would amount to around 400Gb after two hours of running.
- HepRApp related issues (by design and operational)
  - Deleting files while in autorefresh mode not possible. Currently that would require restart after every run (session). Continuous running will require some workaround
  - A window may freeze on event update (no regularity noticed). Requires new window to be opened and set up. Could be tested against different JRE versions