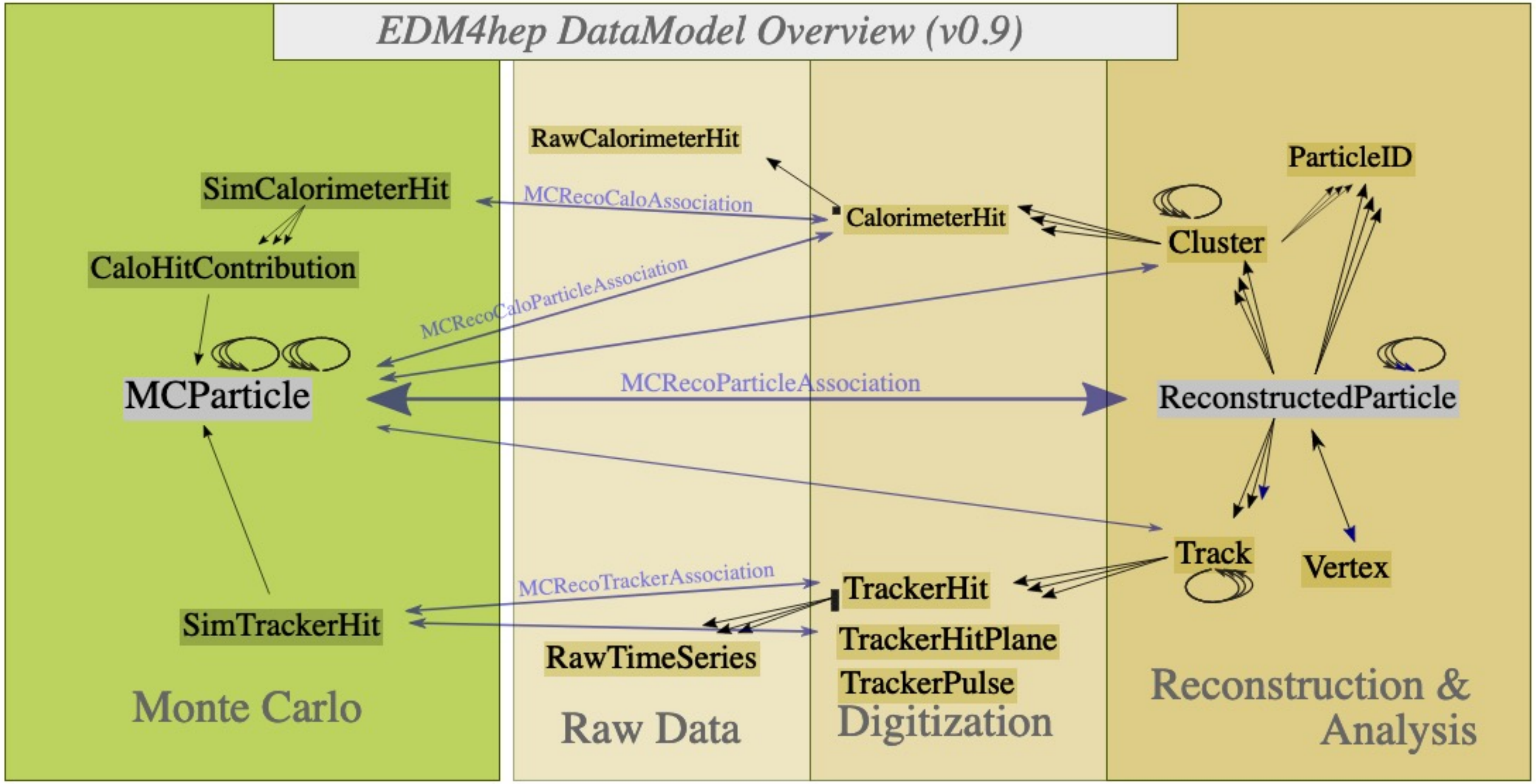
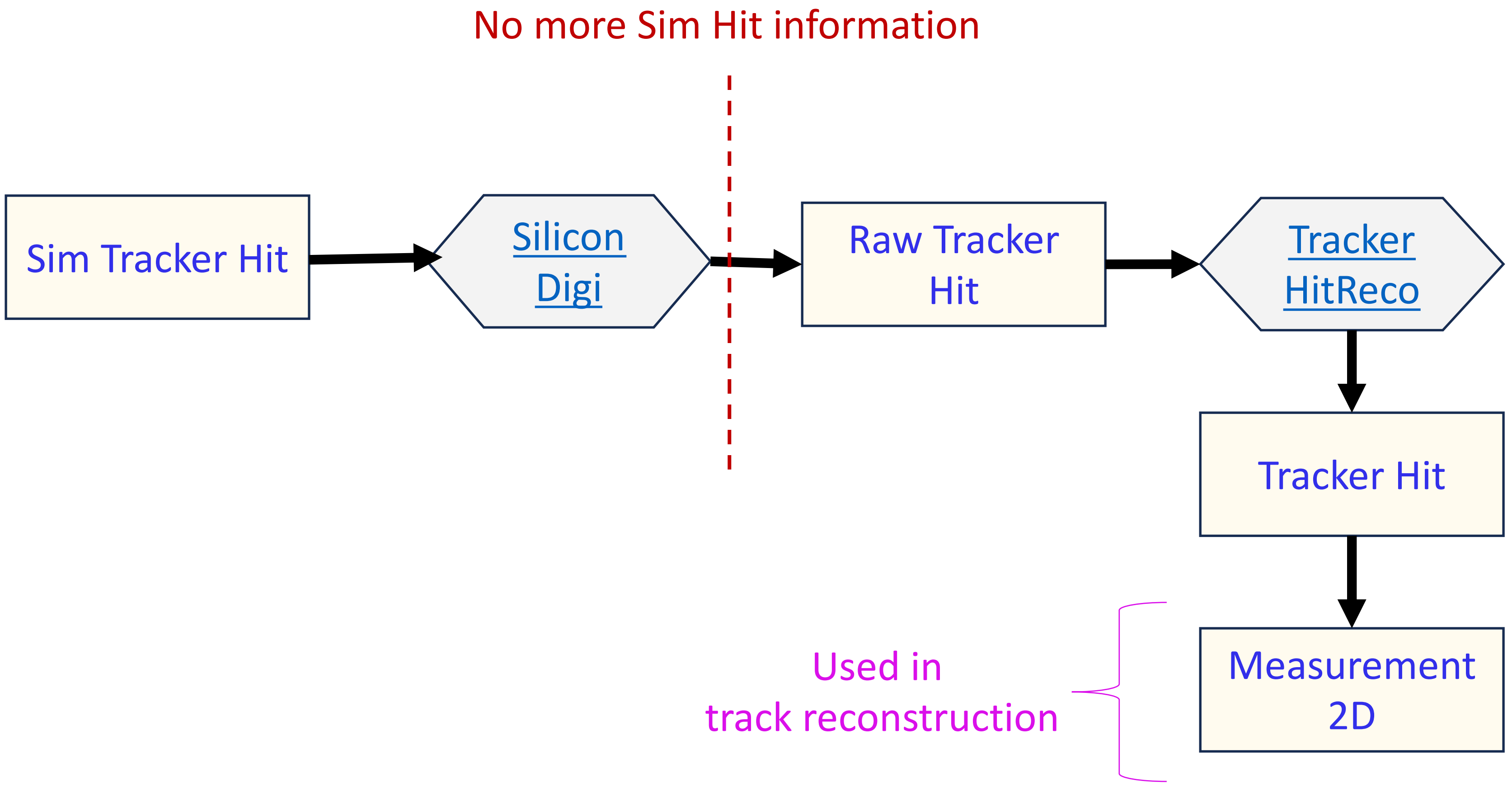
ePIC Data Model

- ❑ ePIC's data model (EDM4eic) is based on EDM4hep

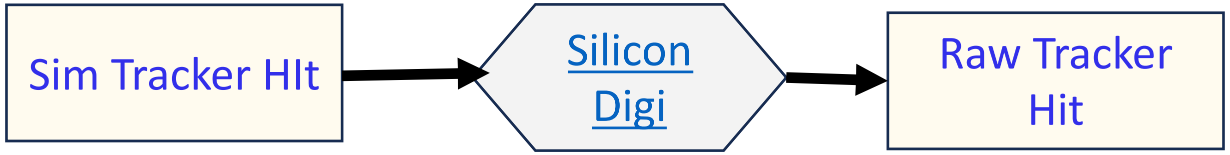


Hit Reconstruction Workflow

- ePIC workflow from simulation hit to detector measurement



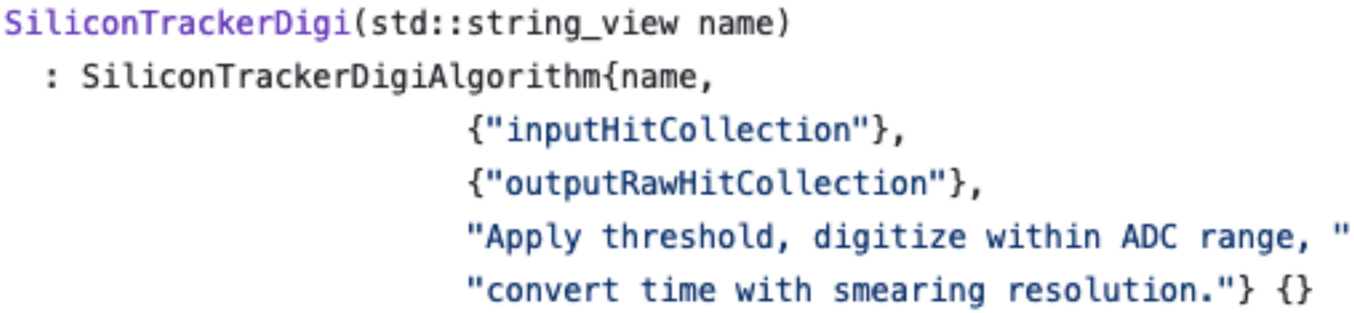
Current Digi Algorithm


Sim Tracker HIt

Silicon Digi

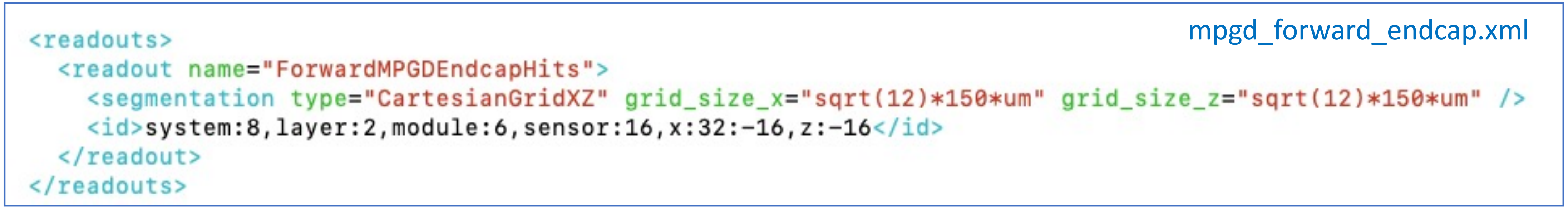
Raw Tracker Hit

- ❑ Digitization in EICrecon
  - Reads in SimHit info (**cell ID**, edep, time)
  - Apply threshold (0.25 keV)
  - Put hit at center of each cell



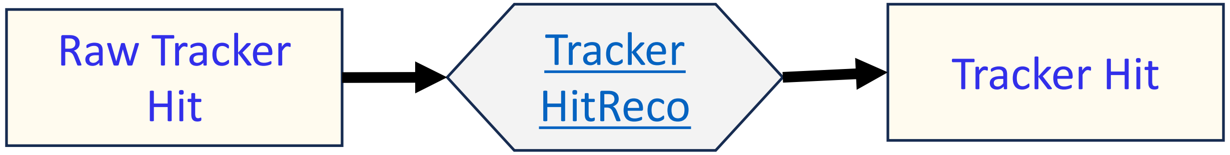
```
SiliconTrackerDigi(std::string_view name)
  : SiliconTrackerDigiAlgorithm{name,
                      {"inputHitCollection"},
                      {"outputRawHitCollection"},
                      "Apply threshold, digitize within ADC range, "
                      "convert time with smearing resolution."} {}
```

❑ **Cell ID** defined from readout segmentation of sensitive volume defined in DD4hep Segmentations


mpgd_forward_endcap.xml

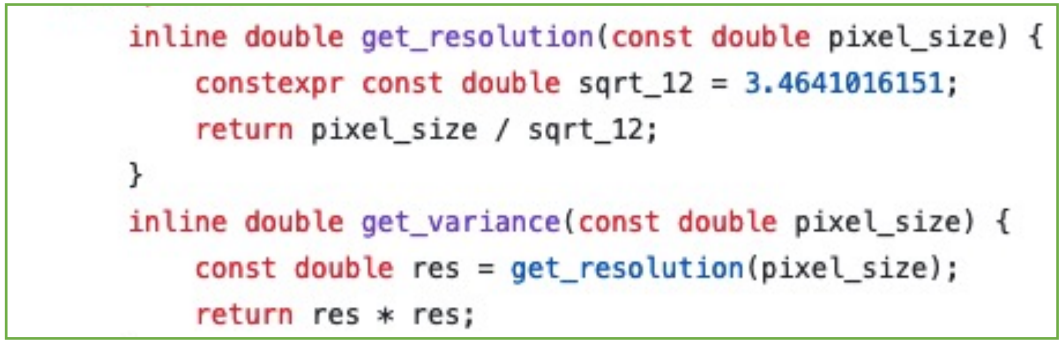
```
<readouts>
  <readout name="ForwardMPGDEndcapHits">
    <segmentation type="CartesianGridXZ" grid_size_x="sqrt(12)*150*um" grid_size_z="sqrt(12)*150*um" />
    <id>system:8,layer:2,module:6,sensor:16,x:32:-16,z:-16</id>
  </readout>
</readouts>
```

Current Hit Reconstruction Algorithm

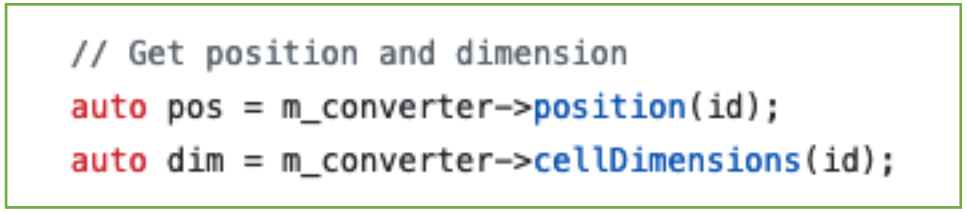


❑ Hit reconstruction in EICrecon

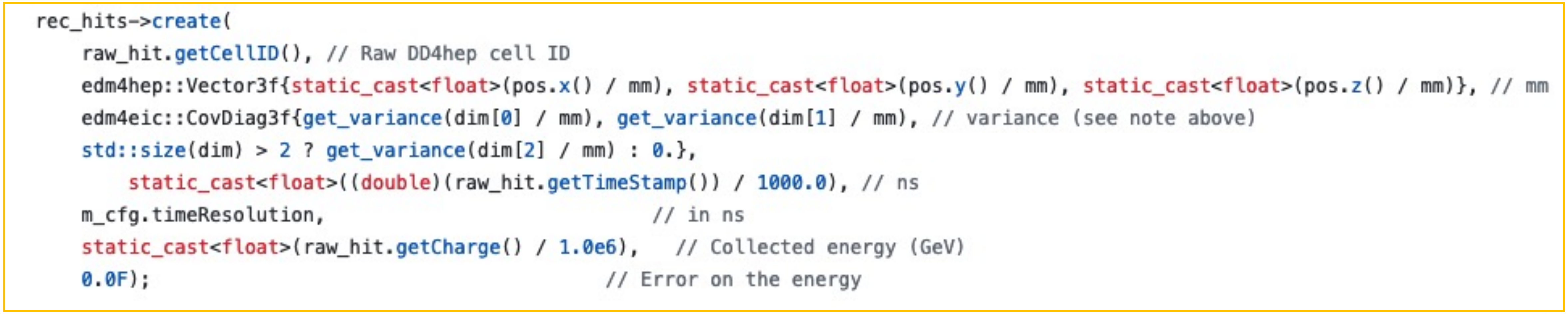
- Reads in Raw Tracker Hit
- Resolution and variance set by cell dimensions



```
    inline double get_resolution(const double pixel_size) {
        constexpr const double sqrt_12 = 3.4641016151;
        return pixel_size / sqrt_12;
    }
    inline double get_variance(const double pixel_size) {
        const double res = get_resolution(pixel_size);
        return res * res;
```

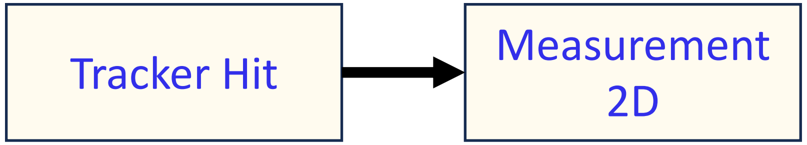


```
// Get position and dimension
auto pos = m_converter->position(id);
auto dim = m_converter->cellDimensions(id);
```



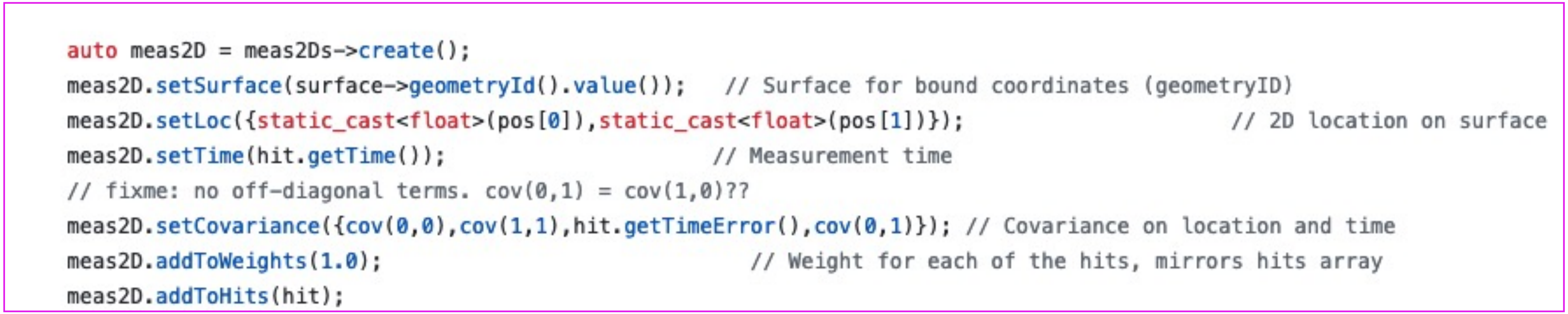
```
rec_hits->create(
    raw_hit.getCellID(), // Raw DD4hep cell ID
    edm4hep::Vector3f{static_cast<float>(pos.x() / mm), static_cast<float>(pos.y() / mm), static_cast<float>(pos.z() / mm)}, // mm
    edm4eic::CovDiag3f{get_variance(dim[0] / mm), get_variance(dim[1] / mm), // variance (see note above)
    std::size(dim) > 2 ? get_variance(dim[2] / mm) : 0.},
        static_cast<float>((double)(raw_hit.getTimeStamp()) / 1000.0), // ns
    m_cfg.timeResolution,                            // in ns
    static_cast<float>(raw_hit.getCharge() / 1.0e6),   // Collected energy (GeV)
    0.0F);                                       // Error on the energy
```

Current Measurement Algorithm


Tracker Hit

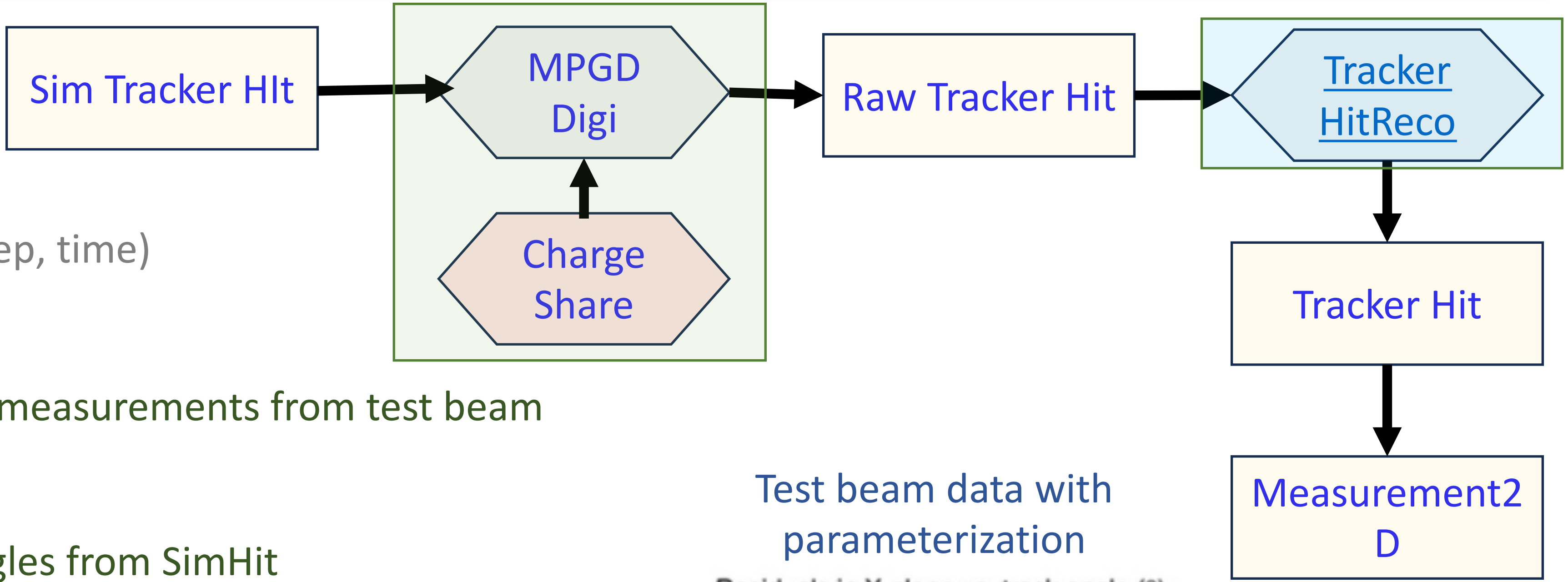
Measurement 2D

- □ Tracker Hit to 2D Measurements
  - Measurements are used in ACTS track fitting
  - No track clustering algorithms defined
    - One Tracker Hit = one 2D Measurement



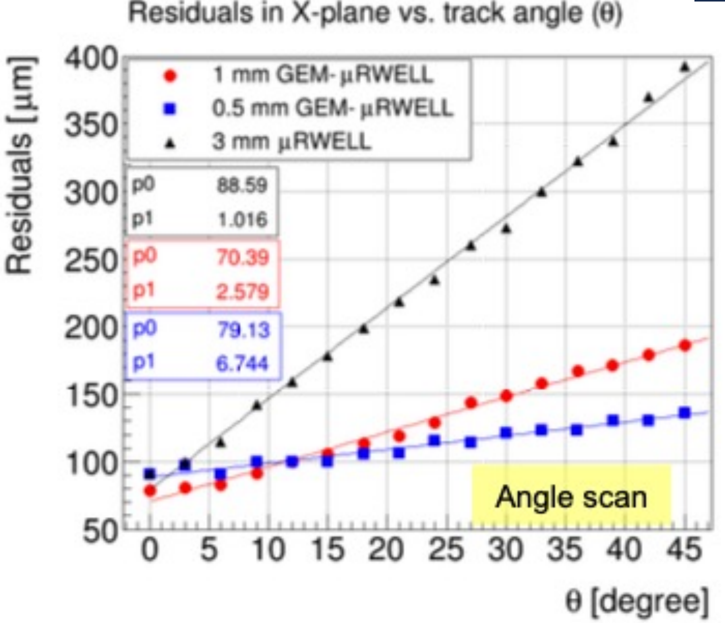
```
auto meas2D = meas2Ds->create();
meas2D.setSurface(surface->geometryId().value());   // Surface for bound coordinates (geometryID)
meas2D.setLoc({static_cast<float>(pos[0]),static_cast<float>(pos[1])});                // 2D location on surface
meas2D.setTime(hit.getTime());                                 // Measurement time
// fixme: no off-diagonal terms. cov(0,1) = cov(1,0)??
meas2D.setCovariance({cov(0,0),cov(1,1),hit.getTimeError(),cov(0,1)}); // Covariance on location and time
meas2D.addToWeights(1.0);                                        // Weight for each of the hits, mirrors hits array
meas2D.addToHits(hit);
```

MPGD Digi Algorithm: Overview



- MPGD Digi
  - Reads in SimHit info (**cell ID**, edep, time)
  - Apply threshold (0.25 keV)
  - Parameterize residual vs. angle measurements from test beam data
  - Calculate polar and azimuth angles from SimHit
  - Apply smearing to charge and digitize several points sampling from the smear function – Gauss
- TrackerHitReco
  - Define TrackerHit position from weighted mean of N charges
  - Set variance based on spread of N charges
  - Output TrackerHit

Test beam data with parameterization


Residuals in X-plane vs. track angle (θ)
Residuals [µm]
θ [degree]
1 mm GEM-µRWELL
0.5 mm GEM-µRWELL
3 mm µRWELL
p0 88.59
p1 1.016
p0 70.39
p1 2.579
p0 79.13
p1 6.744
Angle scan