Designing calorimeter digitization software for the test beam

Dhiman, Guilherme

NORTHERN ILLINOIS UNIVERSITY

DHCal Meeting at NIU
November 17, 2004

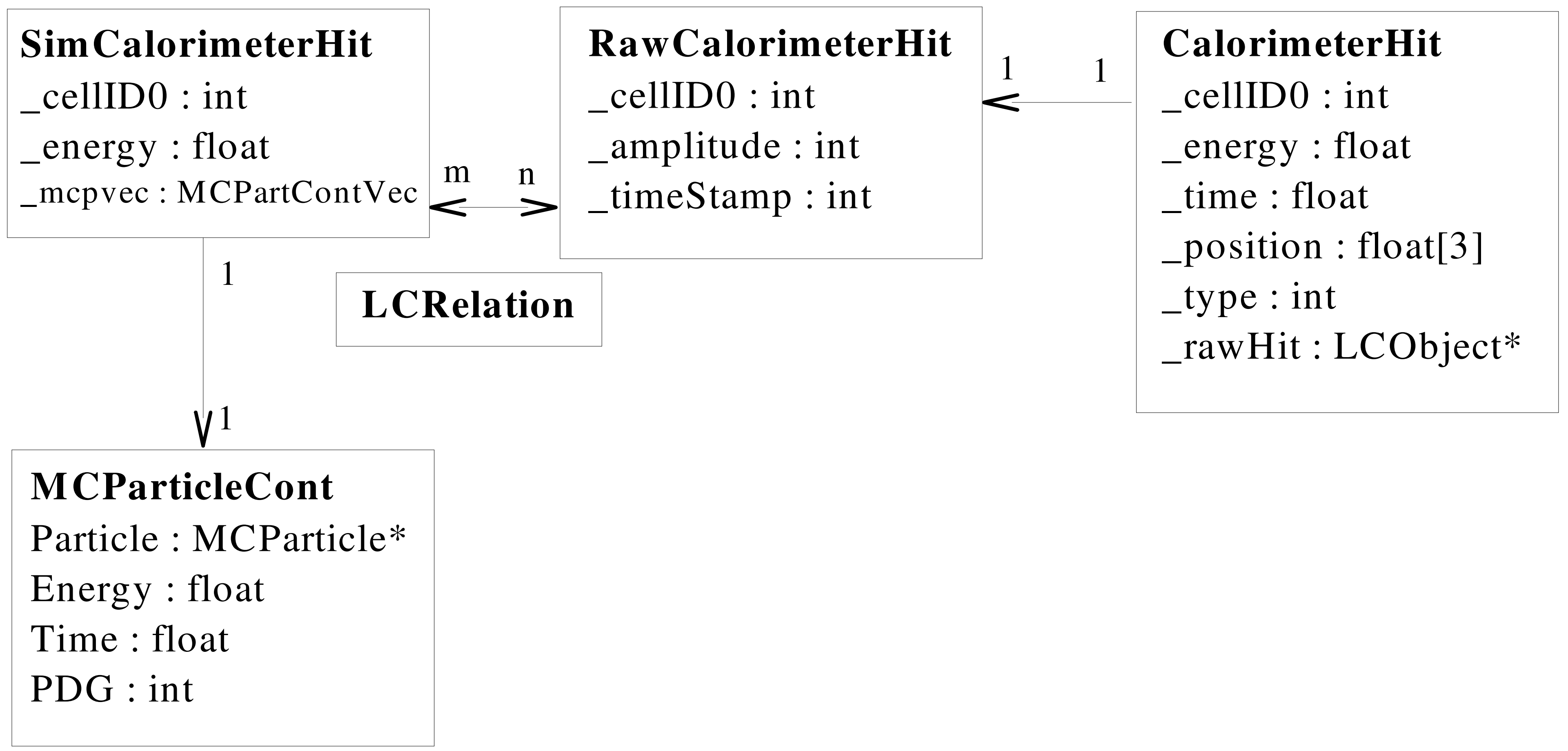
Basic requirements

- Goal: develop a digitization system to be used for the test beam. Result may be used as a basis for the digitizers of the full detector.
- All test beam code is based on C++.
- Use LCIO for persistency, and Marlin as the framework.
- Object oriented design to simplify maintenance and implementation of new functionality.

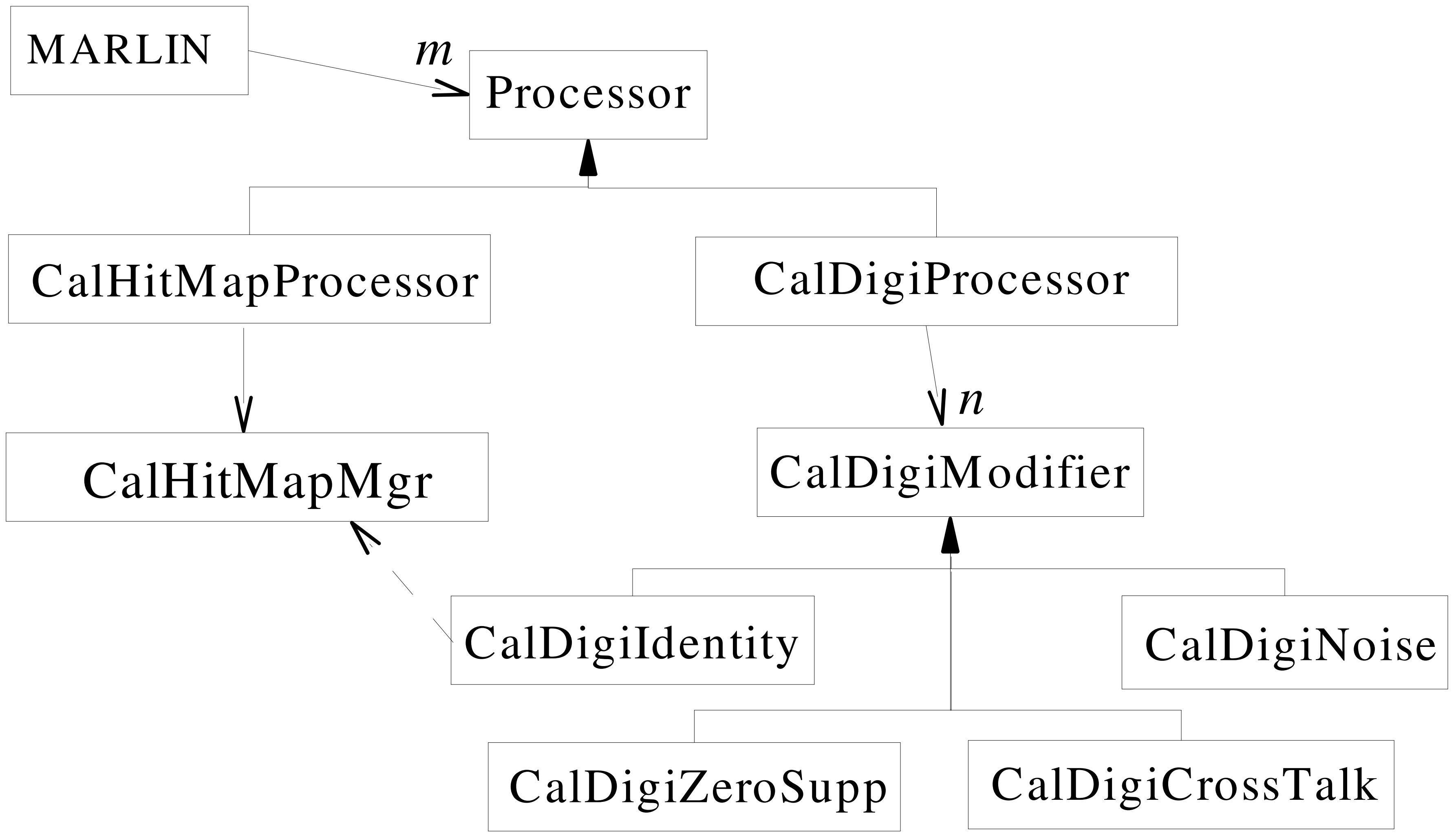
Digitizer role

- Detector simulation (Geant4) produces ideal energy depositions in detector *cells*
- Real detector produces real hits, with ADC counts and time stamps in detector *channels*
- Basic role of digitizer is convert the *ideal hits* into *real hits*, so that simulation output can be as close as possible to the detector output.
- *As close as possible* means that all known data acquisition effects should be taken into account, if possible (inefficiencies, noise, crosstalks, non-uniformities, etc.)

LCIO event model



CalDigi class diagrams



Status

- A first version (proof of concept) is implemented
  - "Identity" tool only, could modify hits in map passed as argument
  - conversion uses constant energy-to-ADC and time-to-timestamp factors
  - LCRelation: to be stored also, looking for example code
- Output LCIO files contain RawCalorimeterHit collection, while keeping simulation collections untouched.
- Creation of new modifiers should be easy, by just copying one of the existing modifier classes and implementing the desired transformation
- Crosstalk requires cell-neighborhood. Nproj code exists (java and C++), but projective geometry is available only in java.

Some open questions

- Modifiers should act on SimCalHits, RawCalHits or any of these?
- Requirements on ordering of modifiers execution (like crosstalks before hot channels)
- Any other variables the modifiers can depend on?
  - Now available: cellID, energy deposition and timing
  - Not available: space points (x,y,z) and cell neighborhood
    Need a geometry-aware class to provide missing information