Update on SoLID Software

Ole Hansen

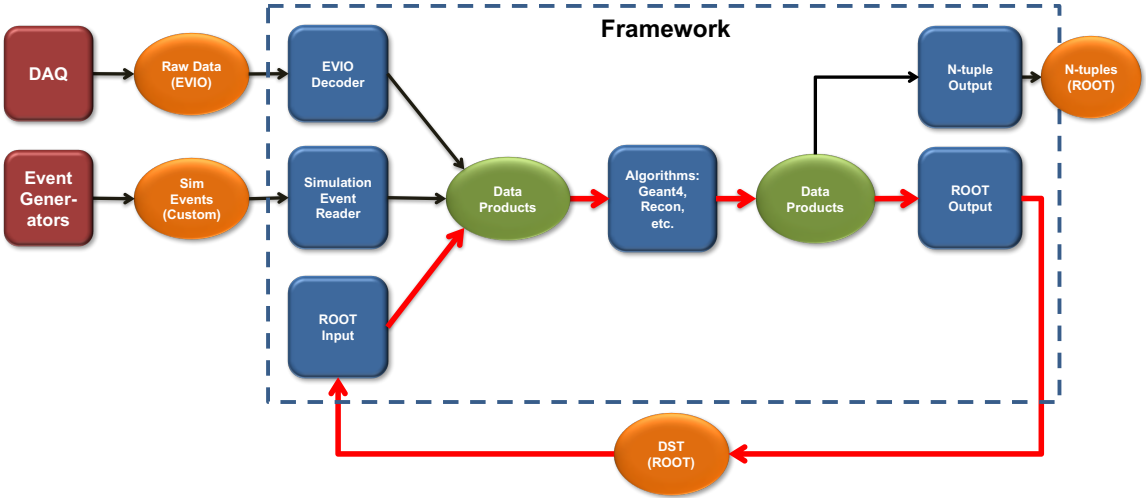
Jefferson Lab

SoLID Collaboration Meeting
October 9, 2020

Mid-Term Goals

- **Infrastructure:** Put in place an end-to-end simulation and reconstruction chain
  - Integrated software environment for (almost) all parts of data processing
  - Standardizes development environment, file format, database system etc.
  - Addresses recommendations from Director's Reviews
- **Simulation Migration:** Port existing GEMC simulation to new framework
- **Algorithm Prototypes:** Write SoLID-specific reconstruction algorithms
  - Focus on track reconstruction (highest priority, most work)
  - Port or adopt existing algorithms as much as possible

Integrated Software Environment



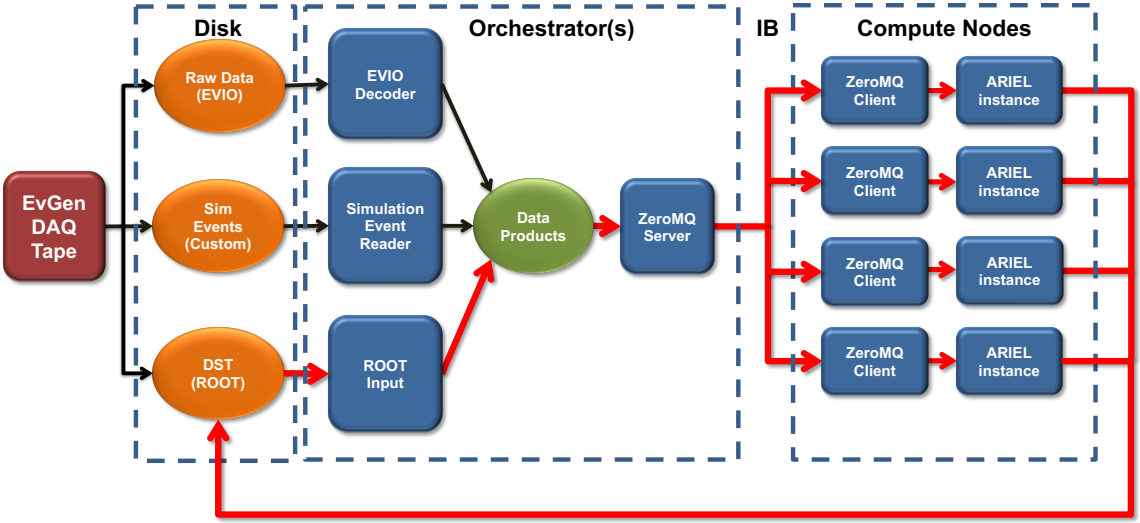
Emerging Software & Computing Challenges & Trends

- **PB-size data sets** (biggest challenge)
  - Need highly scalable software
  - Efficient data management crucial (*e.g.* analysis trains)
- **Software Complexity**
  - Need dedicated software experts/teams (like HEP)
  - Users won't be expected/able to write reconstruction code
  - *Using* the software should be easy
- **Resources moving into the cloud**
  - Consider web-based interfaces (for everything!)
  - Not always efficient ...

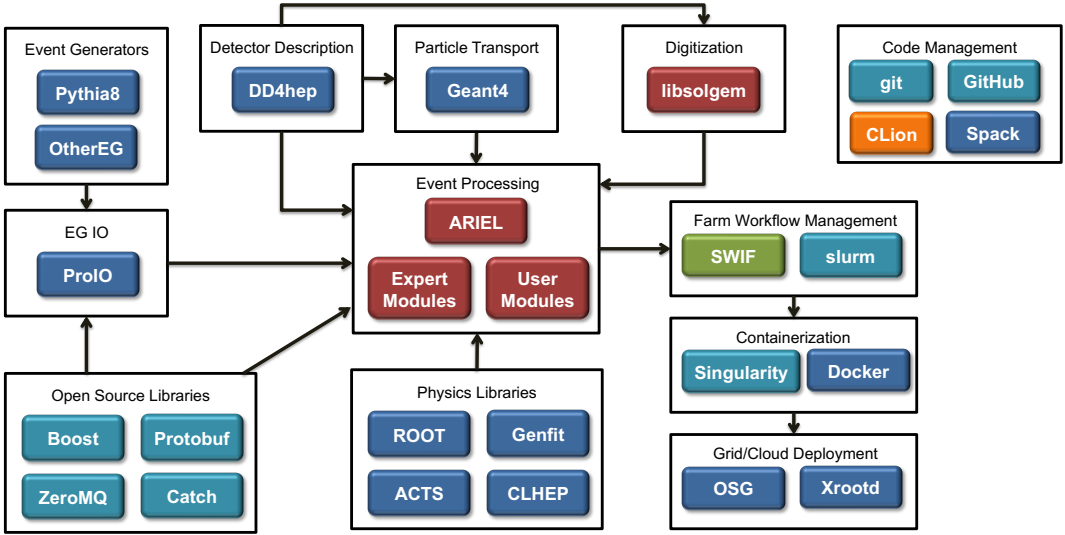
Scaling Models

- Vertical (multi-threading)
- Horizontal (locally distributed on cluster)
  - Critical for HLT and online
  - Offers best tape bandwidth and disk usage efficiency
- Grid/Cloud
  - Good for event generation, detector simulation and digitization
  - Less optimal for reconstruction and analysis (network is bottleneck)

Locally Distributed Processing



Larger SoLID Software Ecosystem

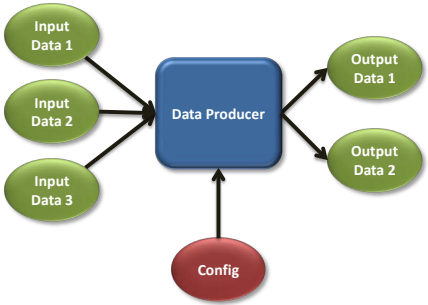


Core Infrastructure: Event Processing Framework

Several good frameworks exist. Probably wasteful to re-invent, esp. if from scratch

Successful approach in HEP frameworks:

- "Separation of concerns" — data products and algorithms
- Data products are stored in files
- 20+ years track record of handling PB of data efficiently

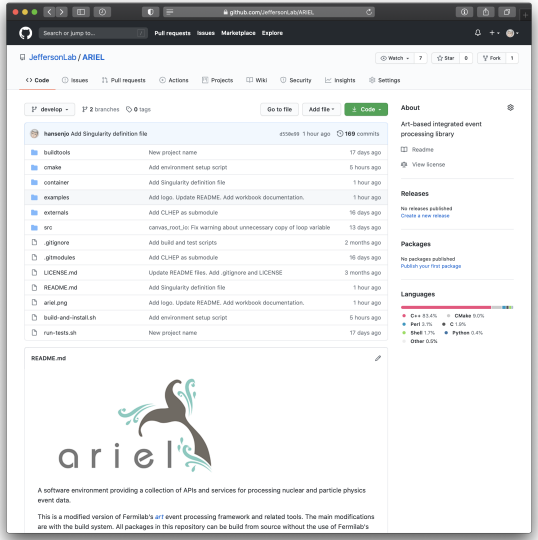


Good early choice will minimize development costs.
No currently available framework offers truly full feature set.
Possible candidates:

- art (FNAL). Widely adopted by FNAL neutrino program, including DUNE. C++. Multithreaded.
- JANA2 (JLab, in-house). Updated version of existing JANA software from GlueX. C++. Multithreaded.
- Gaudi (CERN). Old, but being modernized. Widely used in HEP. C++. Multithreaded. Documentation?
- CLARA (CLAS12). Java-based. Multithreaded & distributed.

Prototype Framework Implementation: ARIEL

- Continuation of my 2018 `art-jlab` project
- Most recent *art* version 3.06.03 (Aug 2020) plus dependent packages, integration tests, toyExperiment, art-workbook
- Task-based event-level multithreading (TBB)
- Supported on Linux & macOS w/C++17
- Singularity build available (see next)
- `https://github.com/JeffersonLab/ARIEL`



ARIEL Containers

Singularity Container on ifarm

```
ifarm1801> module load singularity
ifarm1801> singularity shell /group/solid/apps/ARIEL_0.5.0.sif
Singularity> art --version
art 3.06.03
Singularity> ^D
ifarm1801>
```

- Ubuntu 20.04 LTS base w/gcc 9.3.0
- ROOT 6.22.02 w/C++17 support
- Intended as SDK for user code (work in progress)
- Feel free to try. See container help text for instructions:

```
singularity run-help ARIEL_0.5.0.sif
```

- Plan to create Docker version

High-Level Software Tasks (Infrastructure & Algorithms)

Component	Notes
Databases (geometry, conditions, mapping, materials, alignment)	Choose from several existing systems
Data model	Will evolve, need version tracking
Integrate Geant4 into framework	Reference exists *(artg4)*, test and tune
Flat n-tuple ROOT output module	Universal runtime-configurable module
Event display	ROOT/Eve reference implementation exists
Raw data format decoder	Several C++ implementations exist

Component	Notes
Physics event generators	Several already in use for design studies
Digitization	Early versions exist. Evolve over time
Trigger emulation	Handle DAQ timing effects etc.
Track reconstruction (requires different configurations for SIDIS and PVDIS)	Prototype codes developed
Calorimeter cluster reconstruction	Pick off-the-shelf algorithm
Cherenkov amplitude summation	Pick off-the-shelf algorithm
Particle ID analysis	Traditional algorithms readily available. Good candidate for ML approach.

Collaboration Opportunities

- ANL MEP (already SoLID collaborators)
  - `DD4hep`/`NPDet` (geometry)
  - Overall project primarily aimed at EIC
- EIC group at JLab
  - ESCalate framework: seems too EIC-centric
  - Possible overlap with some tools & ideas (*e.g.* Python interfaces)
- EPSCI group at JLab
  - Support a number of useful/essential common tools, *e.g.* CODA, EVIO, CCDB
  - Want us to use `JANA2`. Some interesting features:
    - ⋆ Multiple processing queues (for handling streaming readout)
    - ⋆ Dedicated tasks for GPU resources
  - Good central resource for Scientific Computing support (*e.g.*, CVMFS, GPU nodes, analysis workflows, ML, OSG etc.)

Next Steps

- Further evaluate `ARIEL`: Write test modules & benchmark
- Evaluate `JANA2` features
- Develop plan to port simulations to `ARIEL` with input from `GEMC`, `artg4` and ANL's `NPDet`
- Investigate deployment on grid for future simulations (GlueX experience)

Summary

- Design & development of SoLID software ecosystem in progress
- Actively evaluating prototype implementations
- Considering collaboration with other groups where useful overlaps exist