GEANT4 for Future Linear Colliders

Norman Graf

(SLAC)

Geant4 User's Workshop
February 21, 2002

Linear Collider Environment

- Detectors designed to exploit the physics discovery potential of e^+e^- collisions at $\sqrt{s} \sim 1$TeV.
- Will perform precision measurements of complex final states.
- Require:
  - Exceptional momentum resolution
  - Excellent vertexing capabilities
  - “Energy Flow” calorimetry
  - Hermeticity

Mission Statement

- Provide full simulation capabilities for Linear Collider physics program:
  - Physics simulations
  - Detector designs
  - Include machine backgrounds
- Need flexibility for:
  - New detector geometries/technologies
- Limited resources demand efficient solutions, focused effort.

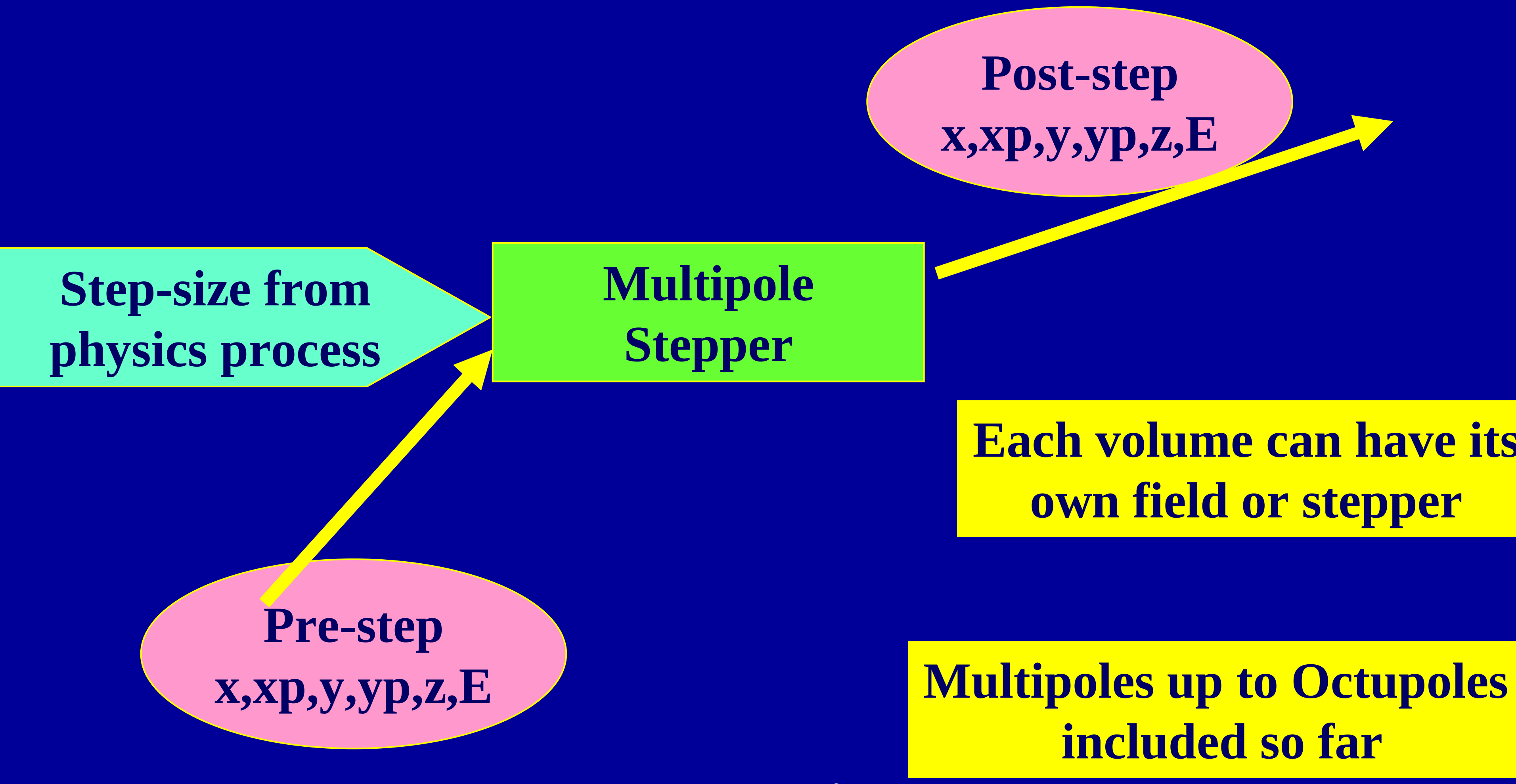
Beam Delivery System

Beamlines are built up out of modular accelerator components

Full simulation of em showers

All secondaries tracked

G4 Stepper


Post-step
x,xp,y,yp,z,E
Step-size from physics process
Multipole Stepper
Each volume can have its own field or stepper
Pre-step
x,xp,y,yp,z,E
Multipoles up to Octupoles included so far

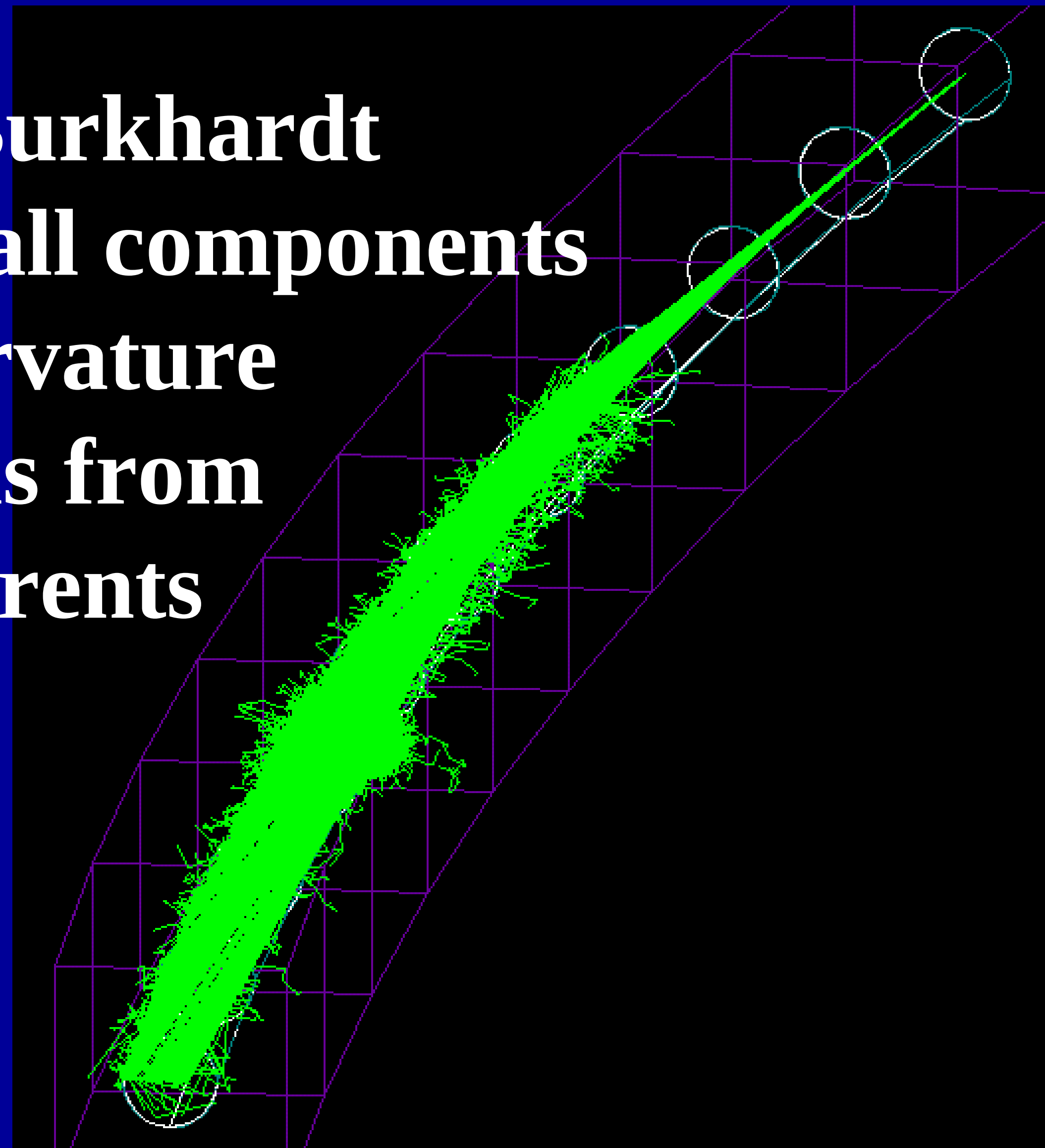
Synchrotron Radiation

Generator of H. Burkhardt

Implemented for all components

Based on local curvature

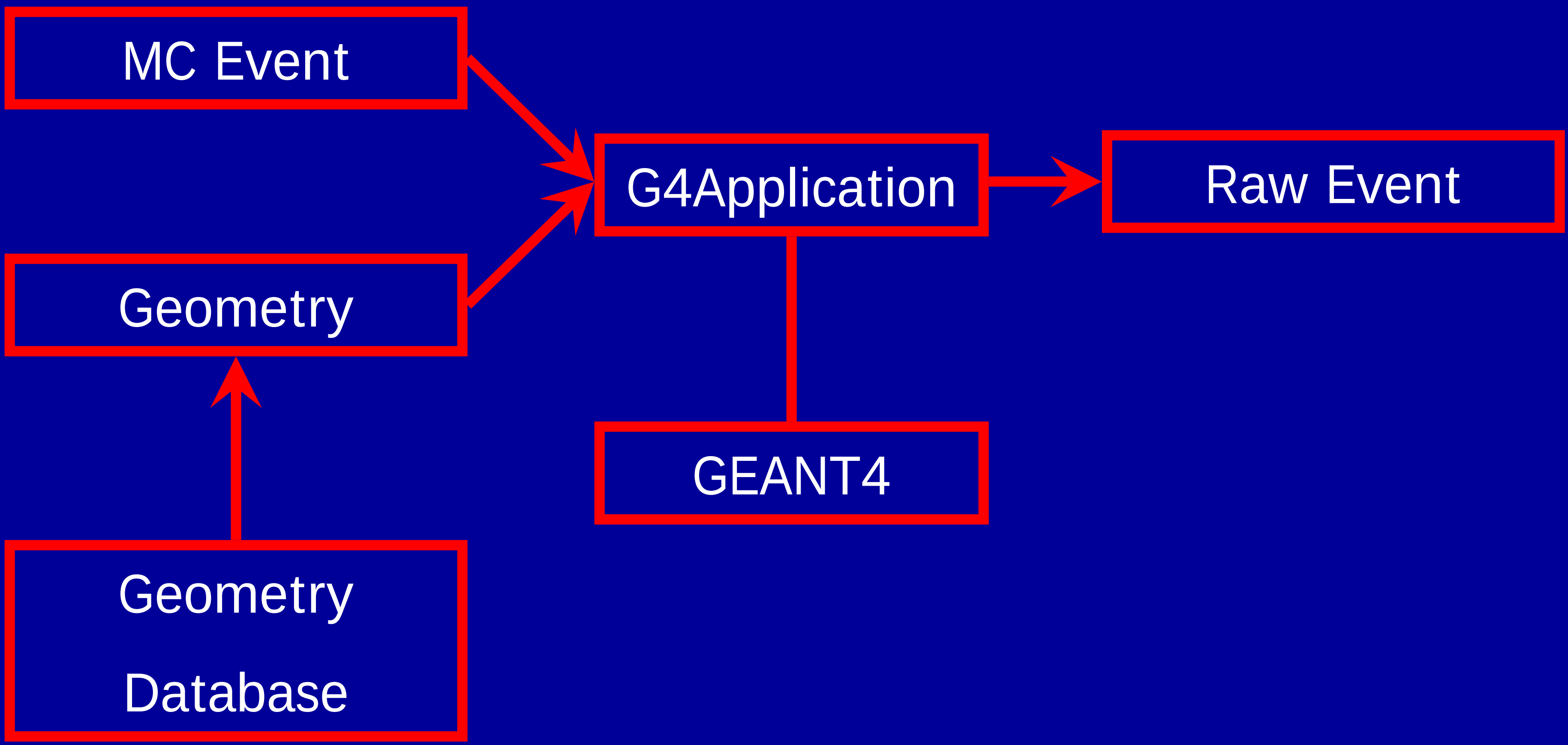
Individual photons from individual parents



BDS Summary/Future Plans

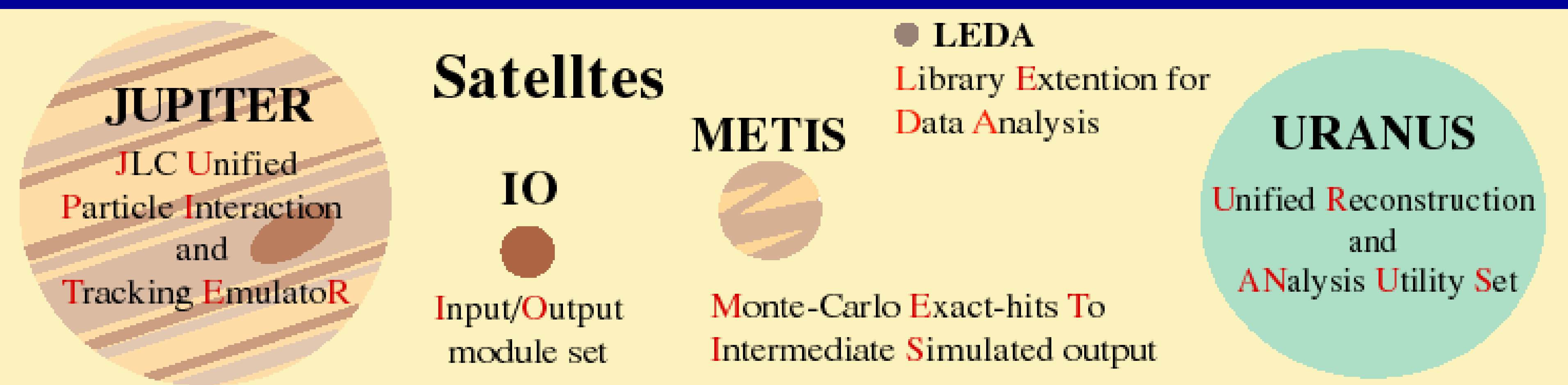
- **Accurate accelerator tracking within Geant4.**
- **Some modification of G4.4.0 is needed.**
- **Interaction with the G4 experts at CERN.**
- **Soon will be fully consistent with standard G4.**
- **Code at the status of an alternative tracker.**
- **Results on SR need to be checked.**
- **New processes incorporated - eg Planck scattering, Laser wire…**
- **Serious collimation studies now possible ...**

LC Detector Full Simulation



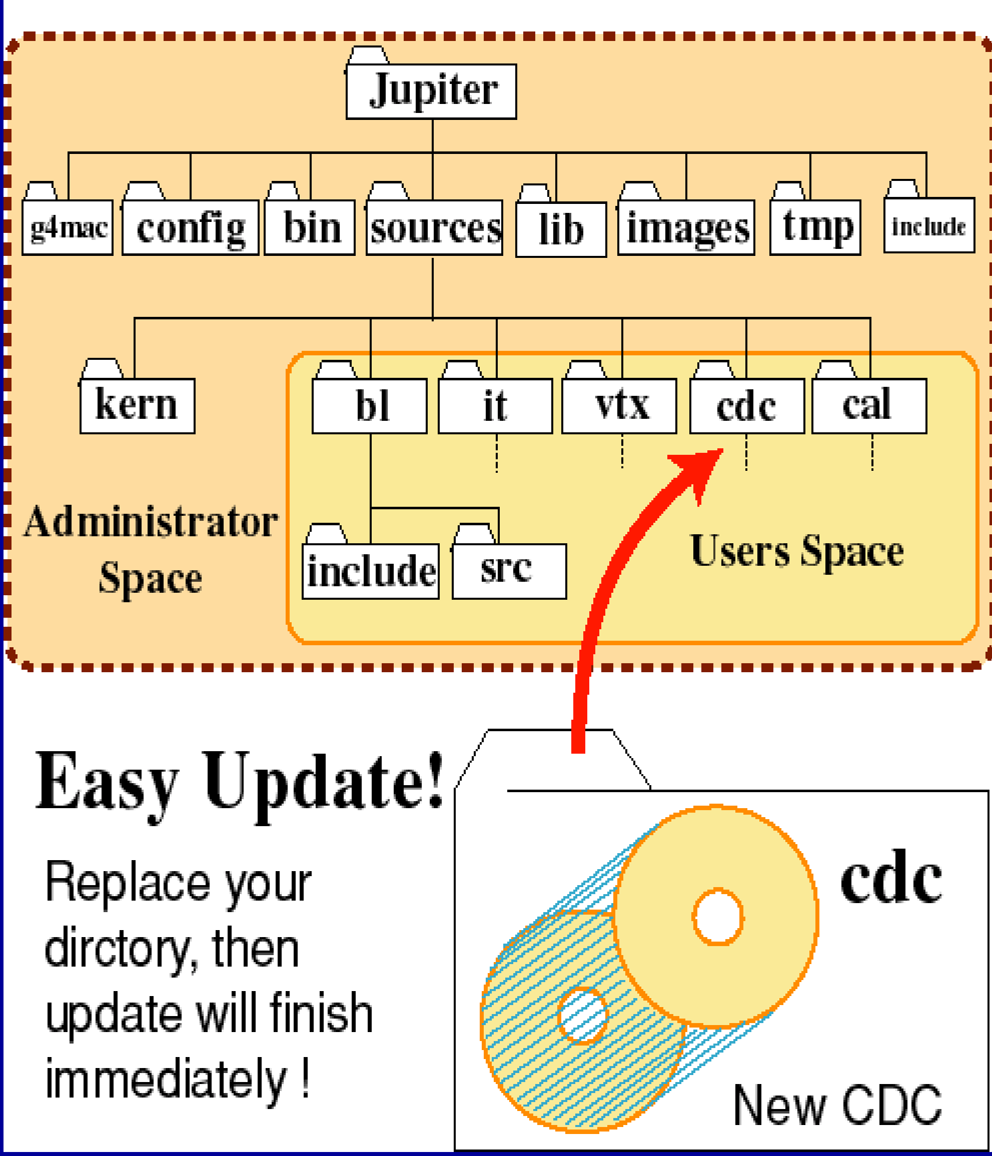
JLC Geant 4 (JUPITER)

- Simulator for JLC Detectors based on Geant4 and ROOT (JSF)
- JLC Unified Particle Interaction and Tracking EmulatoR

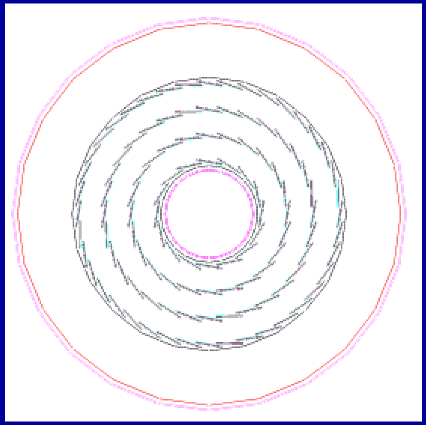


JUPITER (2)

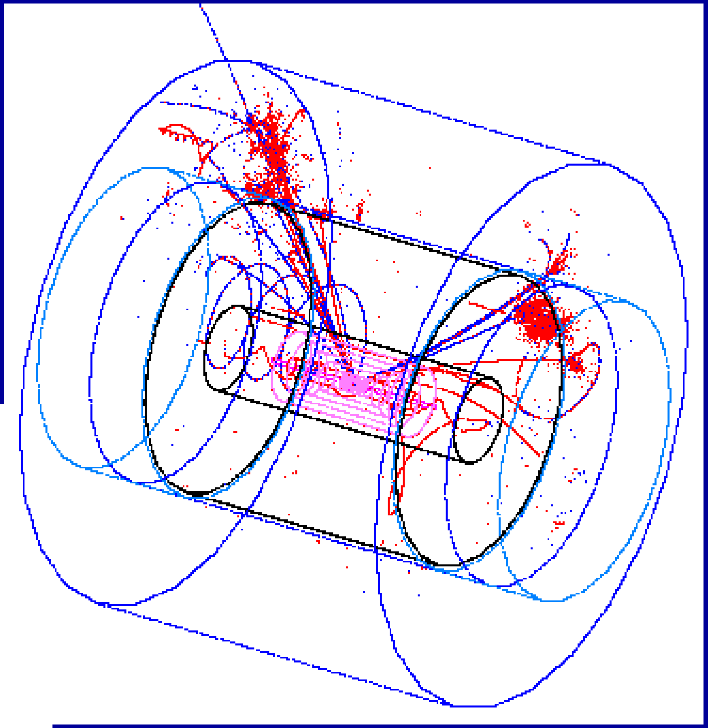
- A set of base abstract classes provide methods for installation and data-output.
- Specific parameters are run-time definable, but geometry structure hard-coded.
- ASCII output → ROOT
- Future use of XML envisioned.

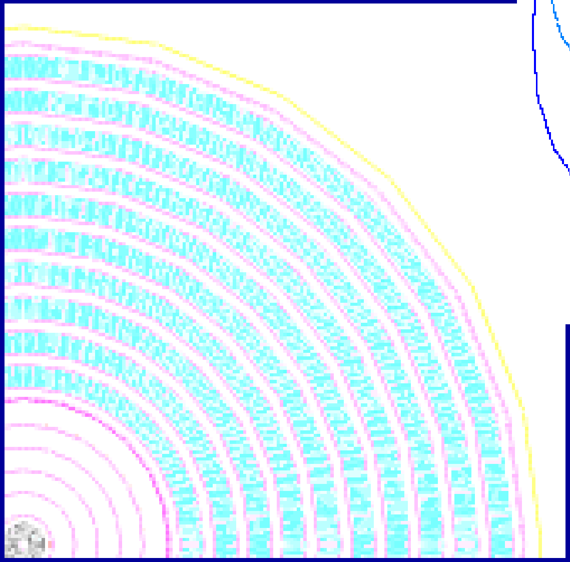


JUPITER(3)



R-phi section of VTX (installed by Aso-lab)
@Toyama National College of Maritime Technology

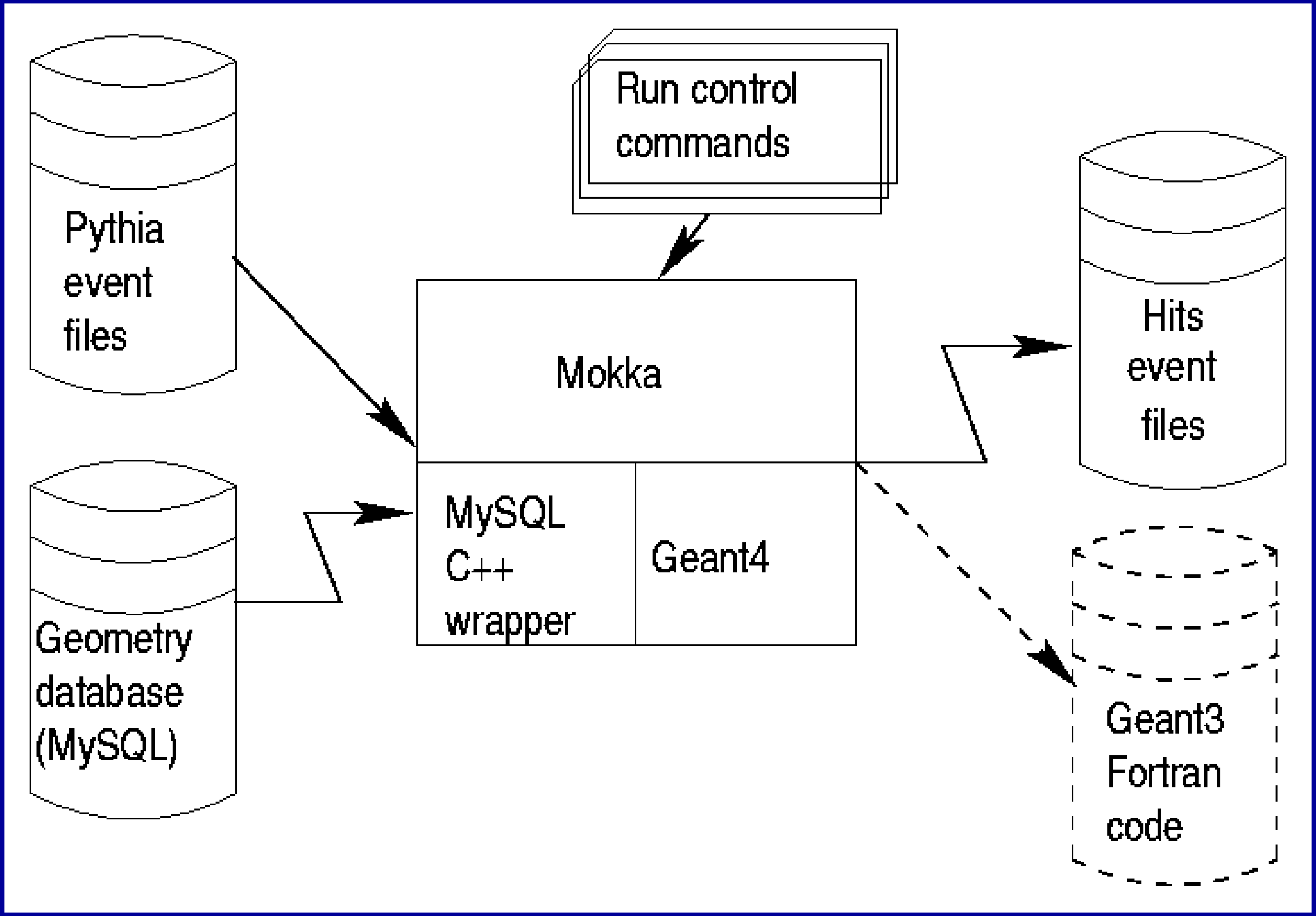




CDC:
Layer 10
Cell   36~108/Layer
Wire 5/Layer

Event display of $e+e- \to Z^0H$

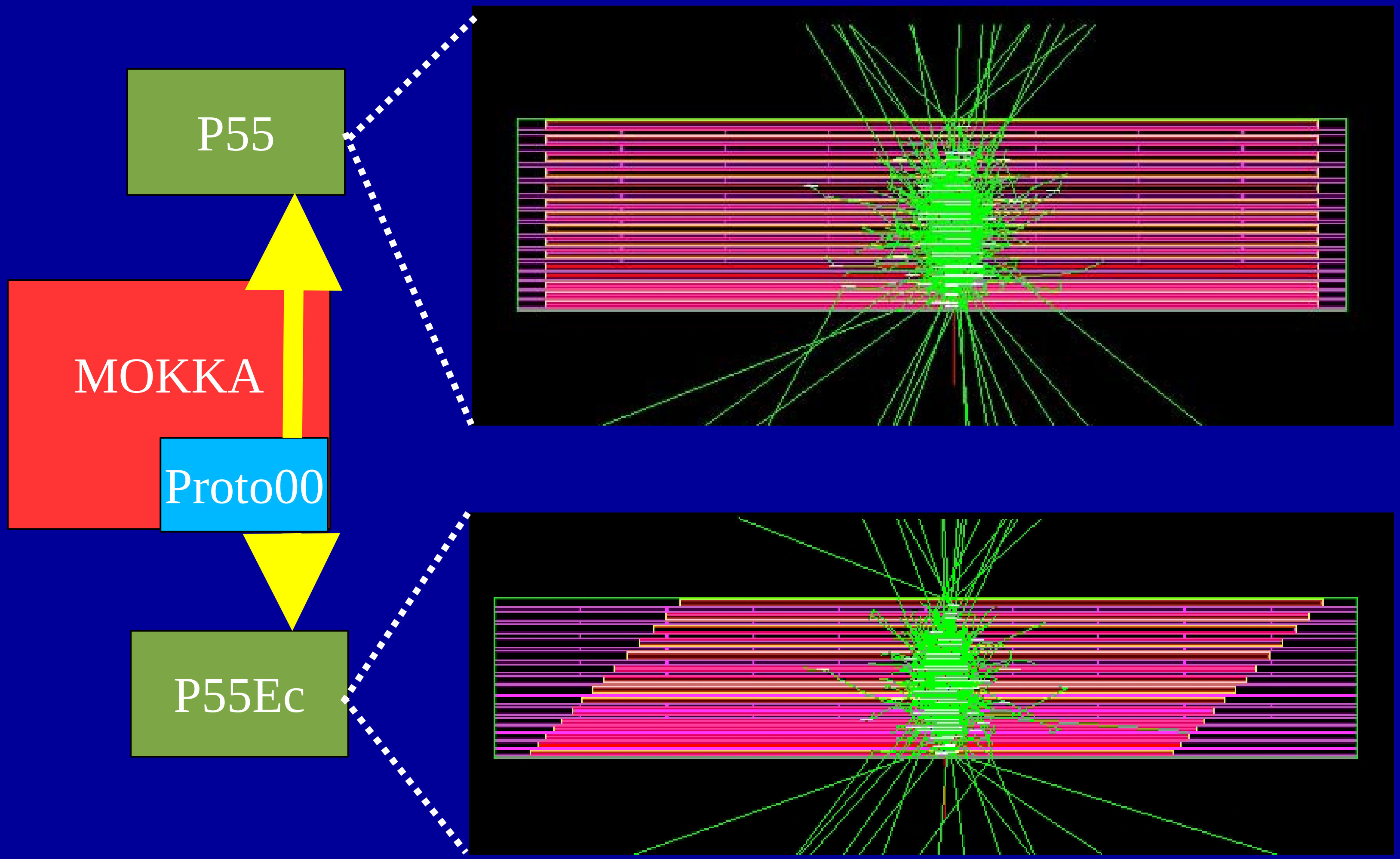
TESLA EM Calorimeter



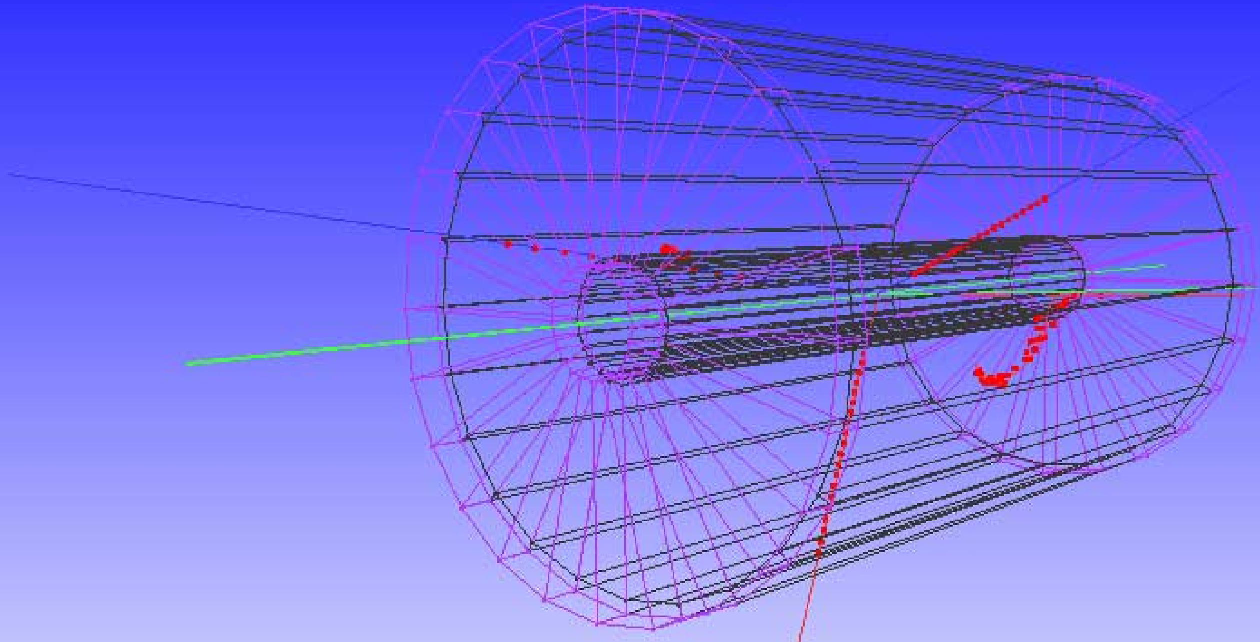
Mokka

- A Geant4 full simulation for the Tesla detector calorimeters, started in December 1999.
- Uses MySQL database plus subdetector-specific geometry drivers.
- Uses unix directory structure for output.
  - run/event/subdetector/hit
- In use for calorimeter optimisation studies.
- ~ 12,500 lines of C++ code.
- Ready for a collaborative development.

The Proto00 geometry driver



TESLA TPC



simple geometry model
simple hit reconstruction
no interface to reconstruction yet

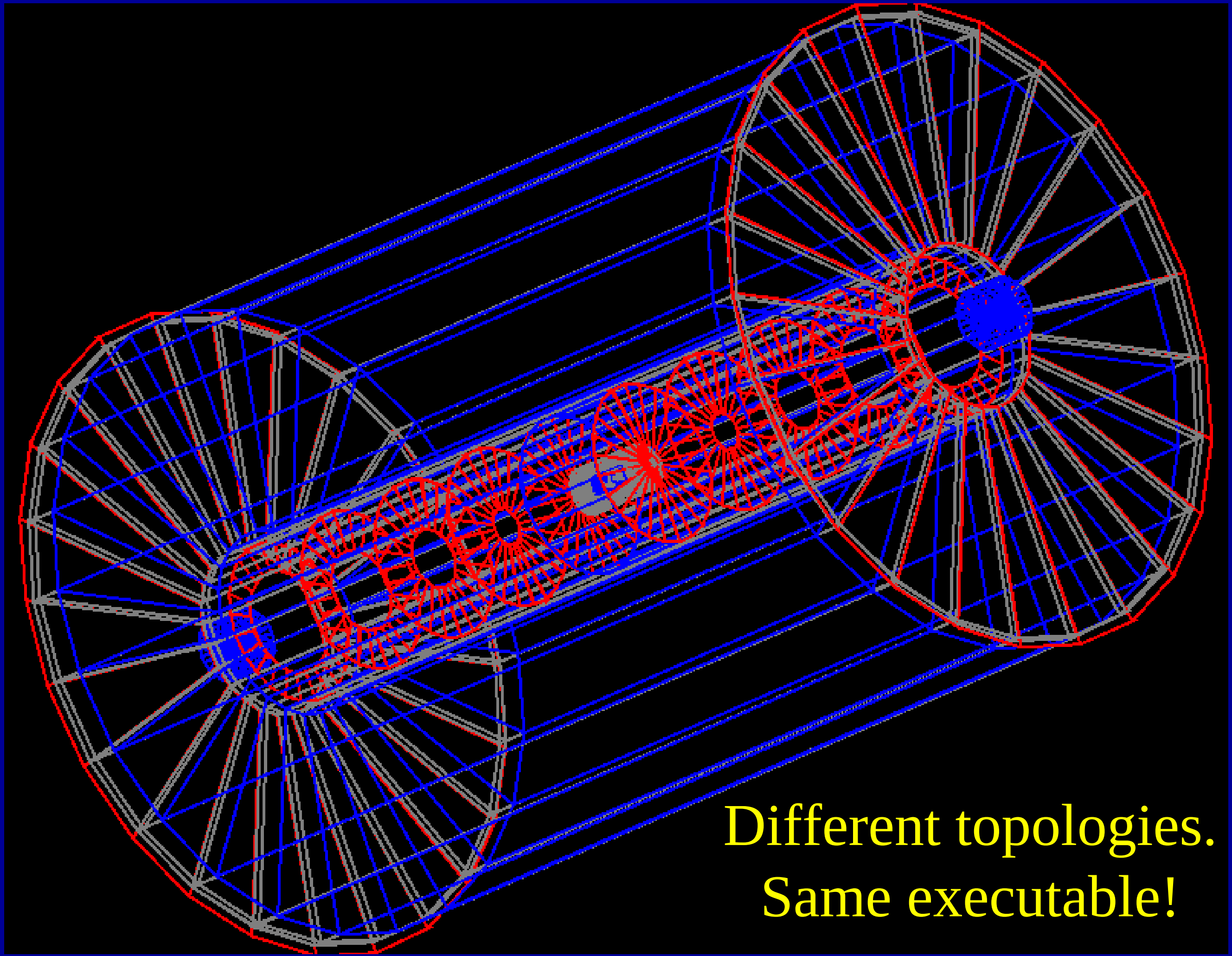
LCD GEANT 4

- GEANT4 geometry defined in XML.
- Dynamic topology, not just parameters.
- Newer parsers (xerces) supports XML Schema. Very useful for “compile-time” type safety and bounds checking.
- Prefer a common G4 XML-based solution.
- Have defined generic hit classes for sensitive tracker and calorimeter hits.
- Details in Dr. Abe’s talk (following)

Why XML?

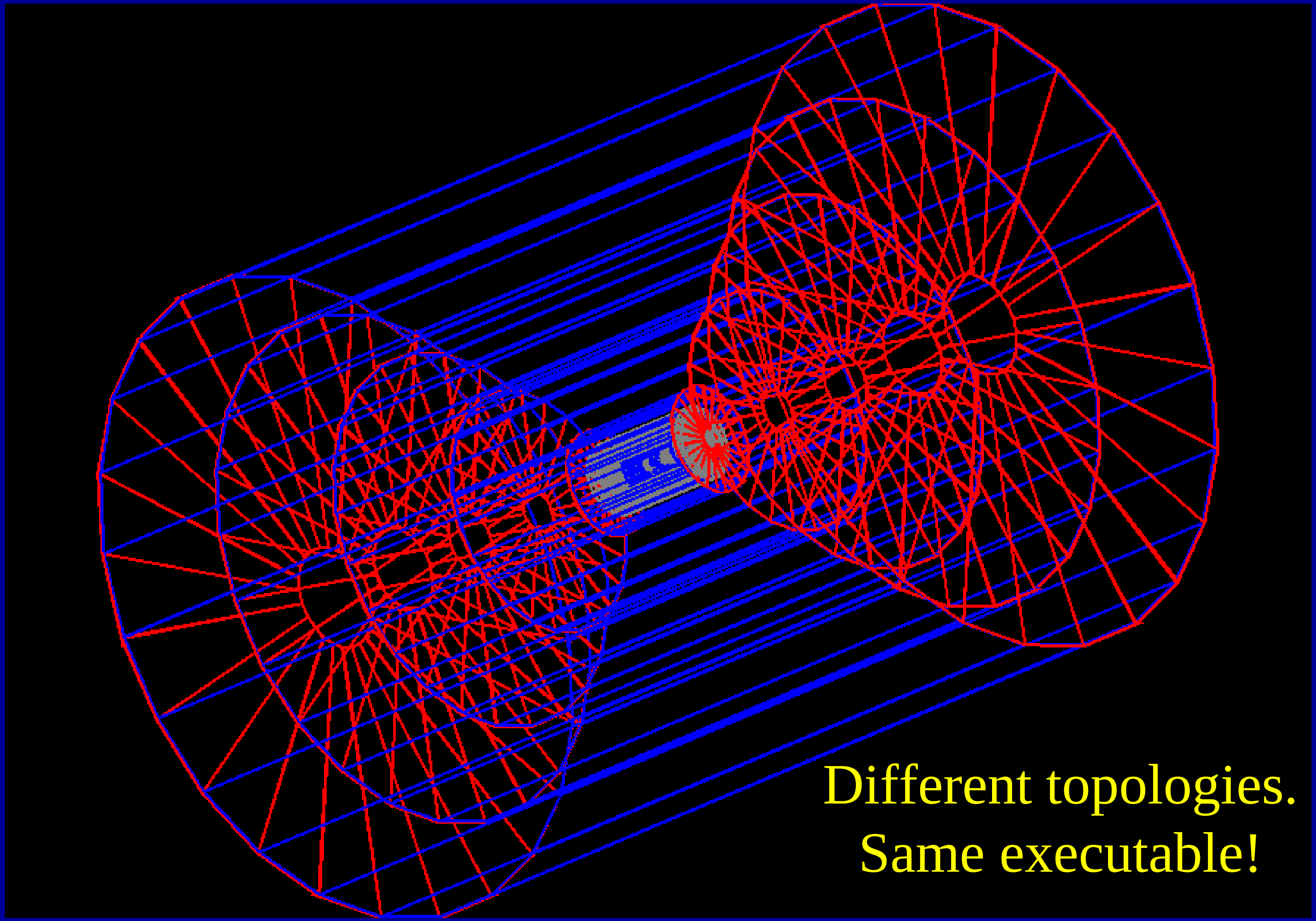
- Simplicity: Rigid set of rules, plain text
- Extensibility: Add custom features, data types
- Interoperability: between OS and languages
- Self-describing data
- Hierarchical structure $\leftrightarrow$ OOP
- Open W3 standard, lingua franca for B2B
- Many tools for validating, parsing, translating
- Automatic code-generation for data-binding

TPC Tracker, Si Disks, CCD VTX



Different topologies.
Same executable!

All Si Tracker, CCD VTX



Generic Hits Problem Statement

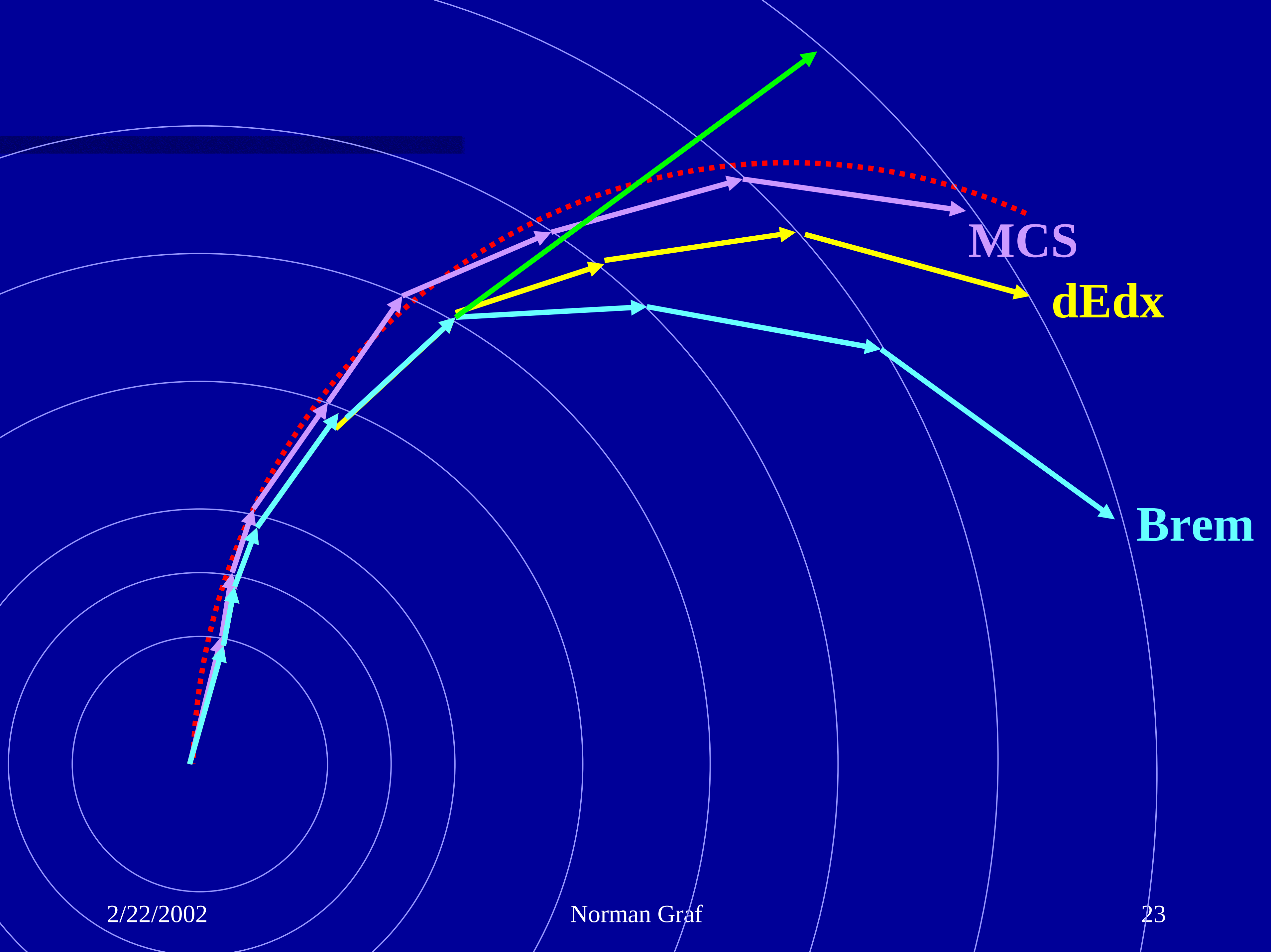
- We wish to define a generic output hit format for full simulations of the response of detector elements to physics events.
- Want to preserve the “true” Monte Carlo track information for later comparisons.
- Want to defer digitization as much as possible to allow various resolutions, etc. to be efficiently studied.

Types of Hits

- “Tracker” Hits
  - Position sensitive.
  - Particle unperturbed by measurement.
  - Save “ideal” hit information.
- “Calorimeter” Hits
  - Energy sensitive.
  - Enormous number of particles in shower precludes saving of each “ideal” hit.
  - Quantization necessary at simulation level.

Track Definition

- Particles suffer various indignities while traversing the detector.
- Knowing track parameters at a single point (*e.g.* the point of generation) is insufficient for precision fits due to material effects (dE/dx, MCS, bremsstrahlung) and field inhomogeneities.
  - No global functional form for the fit.
- Store track information at each volume.


MCS
dEdx
Brem

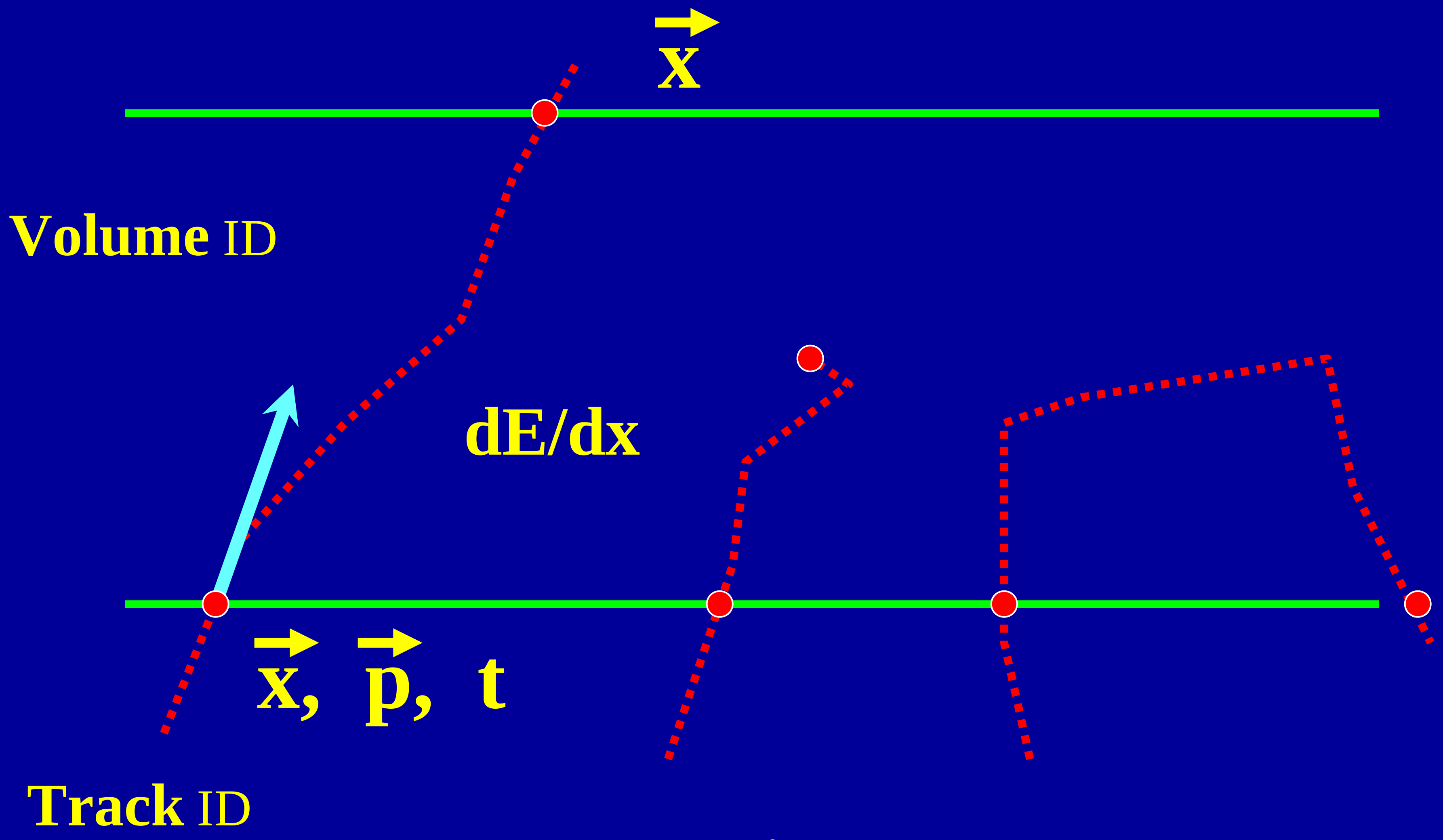
Tracker Hit

- **MC Track Number**
- Encoded **detector ID** (detector dependent)
- Global **hit position at entrance** to sensitive volume
- Global **hit position at exit** of sensitive volume
- **Track momentum** at entrance to sensitive volume
- **Energy deposited** by track in sensitive volume
- **Time** of track's crossing

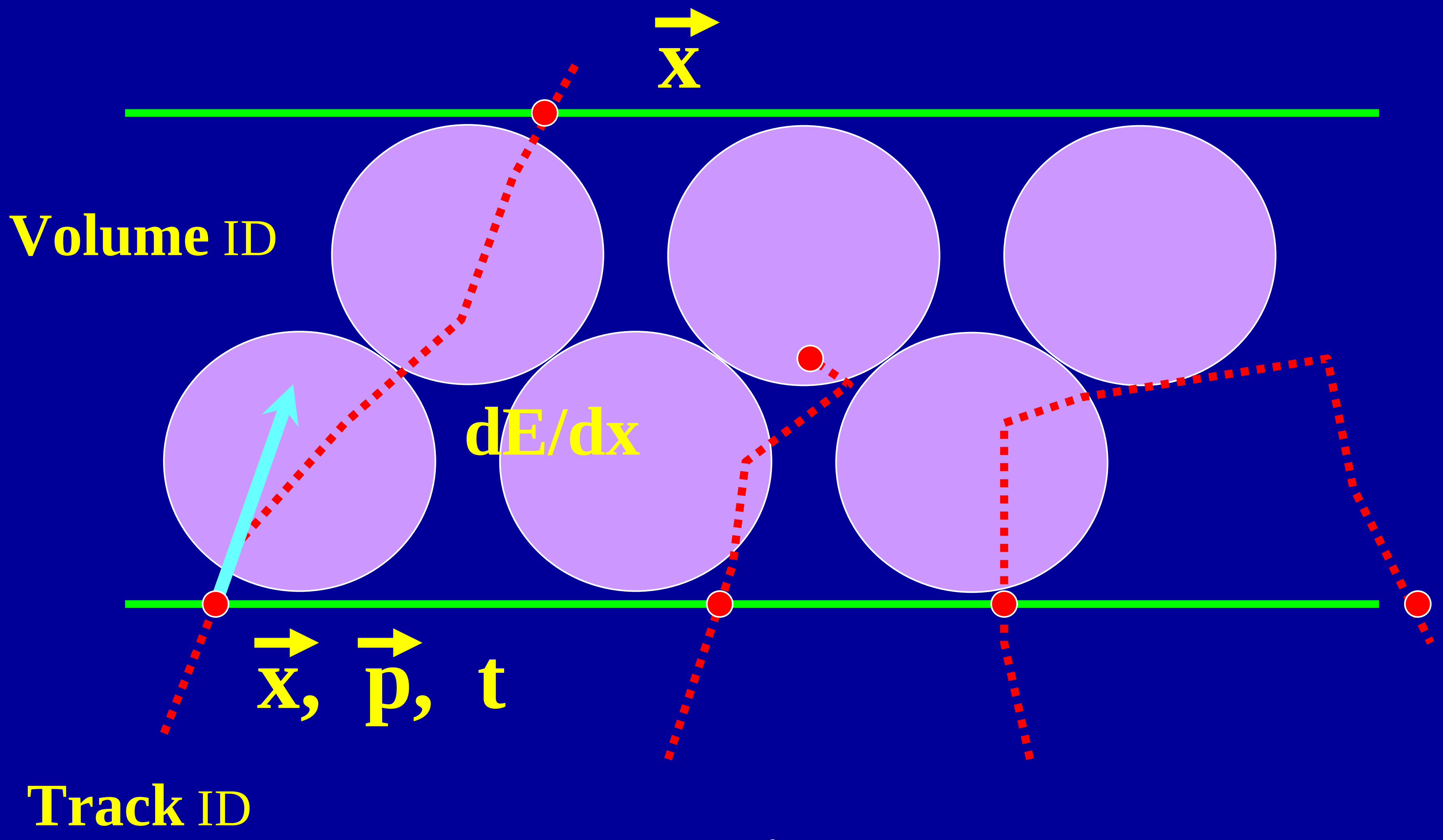
"Convenience" attributes:

- Hit number
- Local hit position at entrance to sensitive volume
- Local hit position at exit of sensitive volume
- Step size used by simulator in sensitive volume

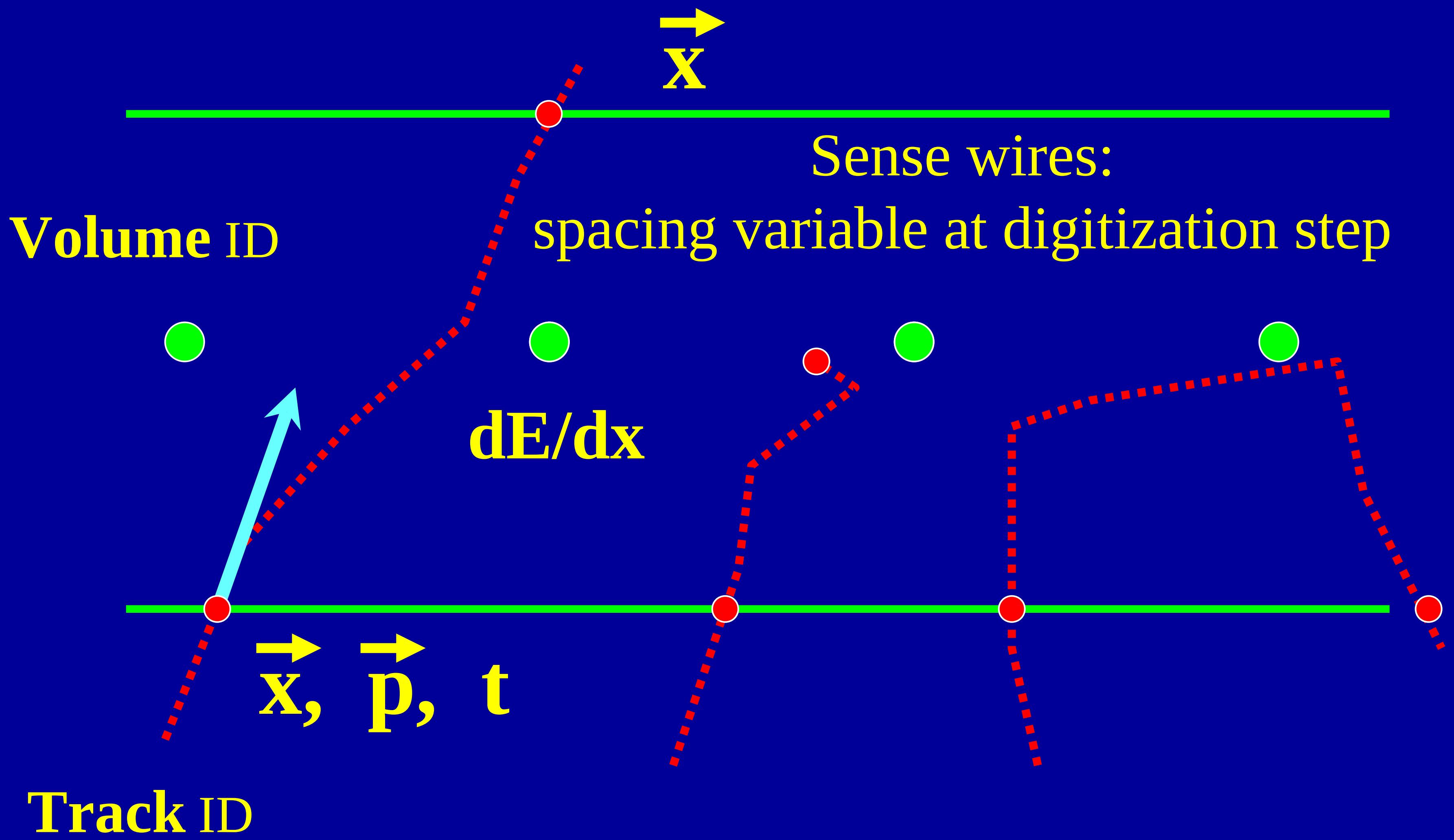
Digitization: Tracker Hit



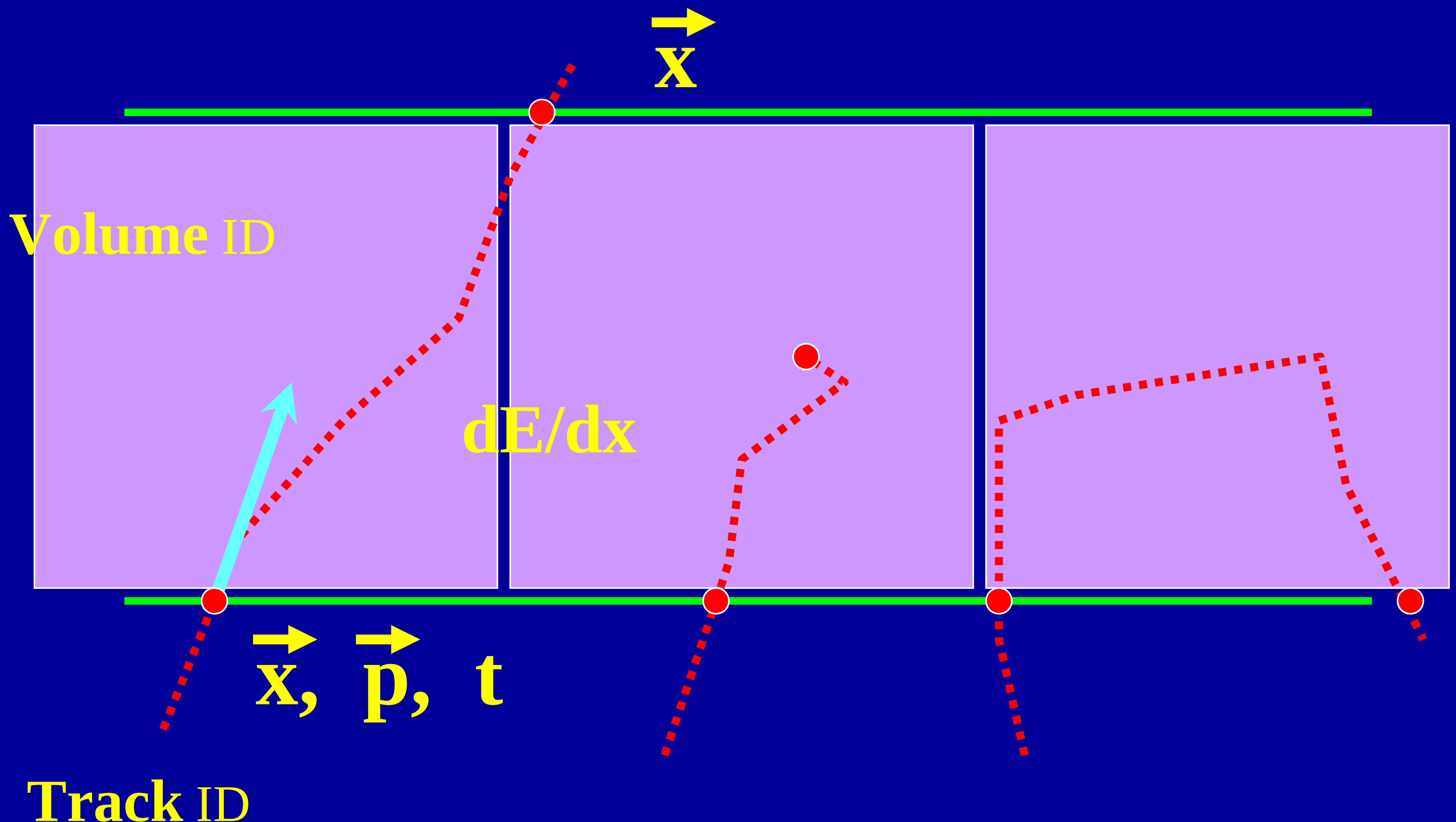
Digitization: Scintillating Fiber



Digitization: Drift Chamber



Digitization: Pixel Detector



Calorimeter Hit

- Encoded detector ID (detector dependent)
- MC ID and energy deposited by each contributing particle

"Convenience" attributes

- Hit Number
- Cell position
  - Radius, Phi, Z of cell
  - X, Y, Z of cell
- Total energy deposited in cell

Hits Summary

- Storing “ideal” hits gives detailed information about MC track.
- Deferring digitization allows studies of detector resolution to be efficiently conducted.
- Can approximate the same in calorimeter by defining small cells, then ganging later.

Main Issues

- Need flexible method to describe geometry.
  - Prefer common XML-based geometry input (GDML?)
- Need to define a "generic" output format.
- Persistence
  - ASCII flat files

    + simple, portable, - large, loss of precision, no structure
  - SIO

    + simple, portable, compact, - no large user base
  - ROOT

    + compact, structured, - more overhead

Towards Internationalization

- Suggest that Tesla, NLC and JLC full simulation groups could run a single GEANT4 executable (others, too...).
- Geometry determined at run-time (XML).
- Write out common "ideal" hits (~flat-file).
- Digitize as appropriate with plug-ins.
- Enormous savings in effort.
- Makes comparisons easy.

Full Simulations

LCD Full Sim	BRAHMS	JIM
GISMO	GEANT3	GEANT3
C++	FORTRAN	FORTRAN

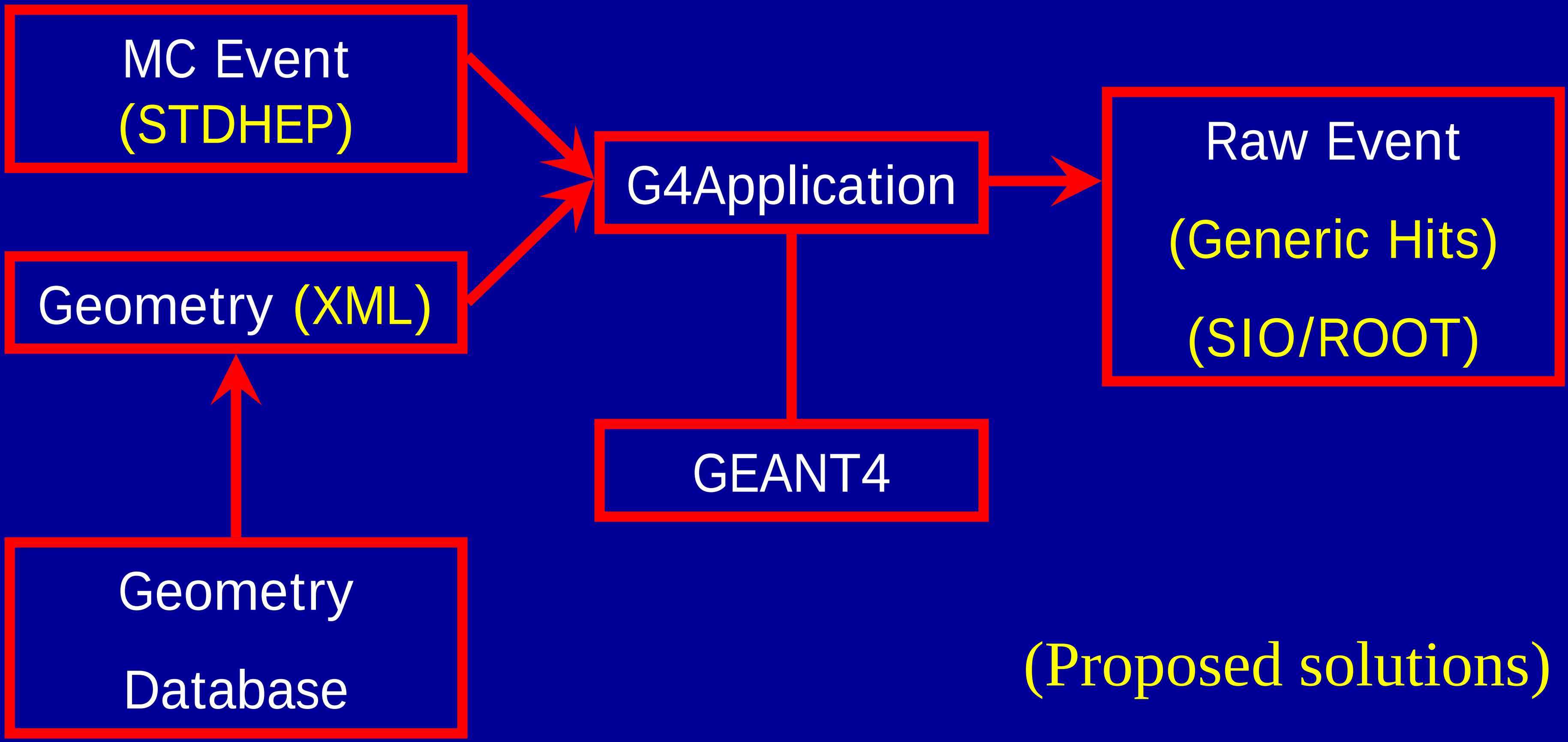
LCDG4	MOKKA	JUPITER

Common GEANT4 executable

XML-based geometry

Generic Hit output

LC Detector Full Simulation



(Proposed solutions)

Acknowledgements

- Toshi Abe (LCDG4)
- Ties Behnke (TESLA G4)
- Grahame Blair (BDS)
- Kotoyo Hoshina (JUPITER)
- Akiya Miyamoto (JLC G4)
- Paulo Mora de Freitas (MOKKA)