CLAS12 Software

G.Gavalian (Jefferson Lab)
Outline

- Software organization
- Data formats
- Visualization
- Reconstruction
- Summary
Software Organization

Source Code:
- reconstruction software common tools are in JAVA
- reconstruction engines are JAVA plugins with CLARA
- code management is done using github (reconstruction plugins)
- build system MAVEN (nightly builds)

Software Distribution:
- software packages distributed via MAVEN repository
- software bundle distributed through git (reconstruction codes and utilities)

Databases:
- calibration database CCDB (Hall-D development)
- sqlite version is distributed with software bundle
- geometry database CCDB, run number and variation dependence
- nightly database dump into sqlite (included in software package build)
Common Tools

Input/Output

✓ raw data in EVIO format composite bank structure
✓ reconstruction output HIPO format (record based and compressed)
✓ common interfaces to read EVIO, BOS, HIPO files
✓ data processing interface (format agnostic) for calibration and monitoring

Databases:

✓ calibration constants and geometry definitions (CCDB)
✓ utilities to display calibration constants, compare for different sets
✓ caching algorithm of database constants for reconstruction

Geometry:

✓ unified geometry package used by simulation, reconstruction and event display
✓ ability to import CAD files (GEANT4 tessellated shapes)
✓ detector visualization package with callbacks and automated occupancy display

Reconstruction:

✓ reconstruction engine hides CLARA complexity from users
✓ automatic detector initialization from database (run dependent)
✓ run dependent calibration constants caching for reconstruction engines
## Software Structure

<table>
<thead>
<tr>
<th>IO</th>
<th>GEOMETRY</th>
<th>DATABASES</th>
<th>PLOTTING</th>
<th>RECONSTRUCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVIO Utils</td>
<td>Geometry Primitives</td>
<td>CCDB Database MySQL, SQLite</td>
<td>Histograms Package</td>
<td>Interfaced Engines</td>
</tr>
<tr>
<td>HIPO Data Format</td>
<td>Detector Definitions</td>
<td>Indexed Reading Detector Constants</td>
<td>Data Persistence</td>
<td>Plugin Based Work Flow</td>
</tr>
<tr>
<td>RAW Data Decoder</td>
<td>Detector Visualization</td>
<td>Constants Visualization</td>
<td>Tuple Trees Visual Analyzer</td>
<td>DB auto resolve Remote, Local</td>
</tr>
<tr>
<td>ROOT convertor</td>
<td>CAD (STL) import</td>
<td>Automated Constants Caching</td>
<td>Network transfer Serialization</td>
<td>CLARA</td>
</tr>
</tbody>
</table>

### Software Components

- **IO**
- **EVIO Utils**
- **HIPO Data Format**
- **RAW Data Decoder**
- **ROOT convertor**
- **GEOMETRY**
- **Geometry Primitives**
- **Detector Definitions**
- **Detector Visualization**
- **CAD (STL) import**
- **DATABASES**
- **CCDB Database MySQL, SQLite**
- **Indexed Reading Detector Constants**
- **Constants Visualization**
- **Automated Constants Caching**
- **PLOTTING**
- **Histograms Package**
- **Data Persistence**
- **Tuple Trees Visual Analyzer**
- **Network transfer Serialization**
- **RECONSTRUCTION**
- **Interfaced Engines**
- **Plugin Based Work Flow**
- **DB auto resolve Remote, Local**
- **CLARA**

### Additional Components

- **USER CODE**
- **CALIBRATION**
- **MONITORING**
User Code
Detector Visualization
- callback on clicks
- coloring by occupancy

Data Canvas
- callback for detector
- callback for table

Database Table
- component callback
- constraint coloring
- comparison utils

Event Processing Pane
- event by event, or whole
- opens: File, ET ring, CLOUD

Detector Visualization
- Layers
Online Monitoring
Data Visualization Package:

- pure Java implementation of plotting
- histograms 1D, 2D and GraphErrors
- functions and MINUIT fitting
- interactive styles and property editors
- tuple tree implementation
- saves data to HIPO files (compressed)
- data serialization for network transfer

Studio UI

- analysis studio for visual data analysis
- interactive fitting, custom function builder
- interactive data set comparison algorithms
- ASCII tuple import/export
- serialized data export with analysis procedure
Data Analysis Studio

Development: G. Gavalian, W. Phelps
Event Analysis
Event Analysis

![GROOT Studio interface with a graph showing mass distribution](image)

- EventTree
  - INCLUSIVE
  - mxekaon
    - mass
    - p
    - theta
- Cuts
- Dataset Descriptors

Status:  
Processed:  
Memory:
Event Analysis
Reconstruction (CLARA 4.3)

- **xMsg CLARA service bus**
  - general purpose public subscribe MPI
  - utilizes zeroMQ socket libraries

- **Sockets that carry messages across various transports**
  - In-process
  - Inter-process
  - TCP
  - Multicast

- **Sockets can be connected N-to-N with patterns**
  - Fan-out
  - Pub-sub
  - Task distribution
  - Request-reply

- **Java, C++, Python bindings**
Clas12 Reconstruction Application
Reconstruction

- **Reconstruction Code:**
  - ✓ written in JAVA (calibration and reconstruction engines)
  - ✓ runs in CLARA environment (multi-threaded) with dynamic configuration

- **Reconstruction package:**
  - ✓ software comes in one package which includes:
    - ✓ descriptors for data banks
    - ✓ local copy of calibration database (sqlite)
    - ✓ magnetic map definitions and swimmers
    - ✓ scripts specifying different run configurations (YAML files describing Engines)

- **Software Structure:**
  - ✓ uber JAR containing compiled common tools
  - ✓ plugin directory for reconstruction engine codes
  - ✓ clara environment running tools (multi-threaded)

- **Tools:**
  - ✓ interactive data format conversion tools (EVIO to HIPO)
  - ✓ interactive analysis studio UI, allows low level tuple analysis
  - ✓ data event viewer with bank filtering tools
  - ✓ Raw data viewer with event decoder and ADC pulse visualization
Amdahl's law
gives the theoretical speedup in latency of the execution of a task at fixed workload that can be expected of a system whose resources are improved.

\[
S_{\text{latency}}(s) = \frac{1}{(1 - p) + \frac{p}{s}}
\]

---

**CLARA Performance**

**CLARA Reconstruction Scaling**

| p_broadwell  | 0.972 |
| p_haswell   | 0.976 |
| p_broadwell_affinity | 0.989 |

---

**Rate**

![Graph showing speedup and Event Rate vs. Number of Threads](image)
Scaling

**DCHB, DCTB and Factorial Engine Execution Times**

![Graph showing engine execution times for DCHB, DCTB, and Factorial with different event processing threads for Broadwell and Haswell]

- **Y-axis**: Engine execution time in milliseconds
- **X-axis**: Event processing threads

- **Graph Lines**:
  - DCHB-Broadwell
  - DCTB-Broadwell
  - DCHB-Haswell
  - DCTB-Haswell
  - Factorial-Broadwell
  - Factorial-Haswell

---

G. Gavalian (NOV 10 2016)
Multi Socket Machines

14-18 Core (HCC)
CLARA Vertical Scaling (Thread Affinity)

CLAS12 Reconstruction Application (CLARA)

One JVM  | Two JVM  | Two JVM 2 NUMA  | Four JVM 4 NUMA

0.045    | 0.09     | 0.135          | 0.18
Summary

Common Development Environment:
• common tools library aids users for fast application development
• unified interface for database constants access and comparison
• unified detector visualization (common detector enumeration scheme)
• efficient data format for storing DST and intermediate results
• reconstruction framework for multi-threaded engines for detectors

Reconstruction:
• reconstruction software complete in CLARA 4.3 (with xMsg)
• multithreading tests show smooth vertical scaling (no thread contention)
• plugin driven multithreaded reconstruction package is in place
• all in one package includes magnetic maps, local database copy

Software Distribution:
• github repository for the common CLAS12 package with build scripts
• detector packages (plugins) have separate github repositories
• versioned reconstruction package distributed via web download
Backup

BACKUP SLIDES
CLAS12 Reconstruction Application Performance for Different CLARA Configurations

Average processing rate in kHz

- One JVM
- Two JVM
- Two JVM: 2 NUMA Nodes
- Two JVM: 4 NUMA Nodes

Single-IO-Services
Vertical Scaling (Thread Affinity)

**CLAS12 Reconstruction Application Performance for Different CLARA Configurations**

Average processing rate in KHz

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Multi-IO-Services</th>
<th>Single-IO-Services</th>
</tr>
</thead>
<tbody>
<tr>
<td>36P36H</td>
<td>0.09</td>
<td></td>
</tr>
<tr>
<td>18P18H*2</td>
<td>0.135</td>
<td></td>
</tr>
<tr>
<td>(8P8H+10P10H)*2</td>
<td>0.18</td>
<td></td>
</tr>
<tr>
<td>9P9H*4</td>
<td>0.18</td>
<td></td>
</tr>
<tr>
<td>Any36*2</td>
<td>0.135</td>
<td></td>
</tr>
</tbody>
</table>