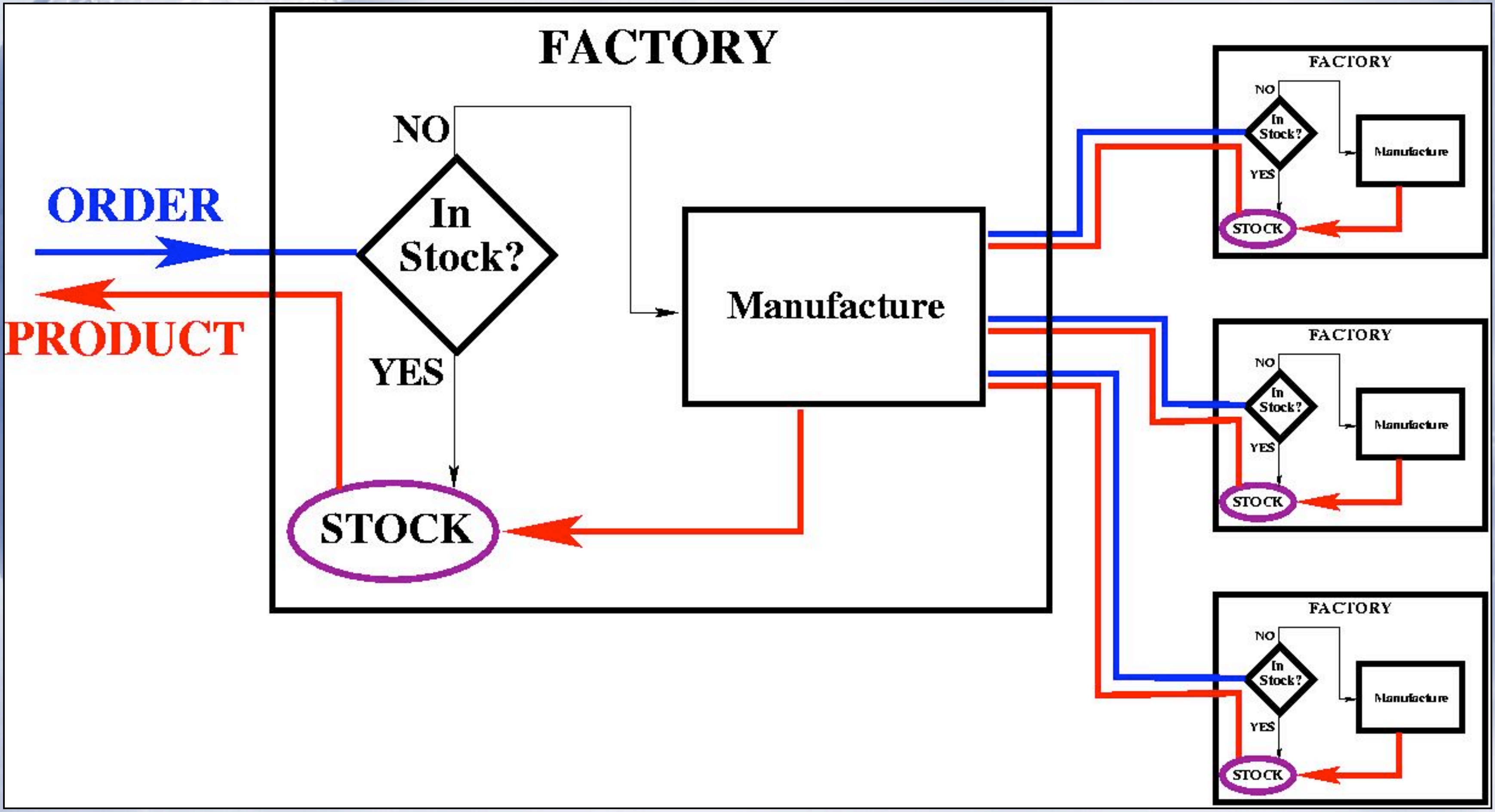
JANA

C++ Event Reconstruction Framework

David Lawrence JLab

Feb 2, 2007

JANA Factory Model



JANA Factory Model

- The factory model causes data to be produced only “on-demand”

- CPU time is not wasted on reconstruction that is not needed for that event

- Particularly beneficial for event filters. e.g. Level 3 trigger, monitoring programs

Mutli-threading

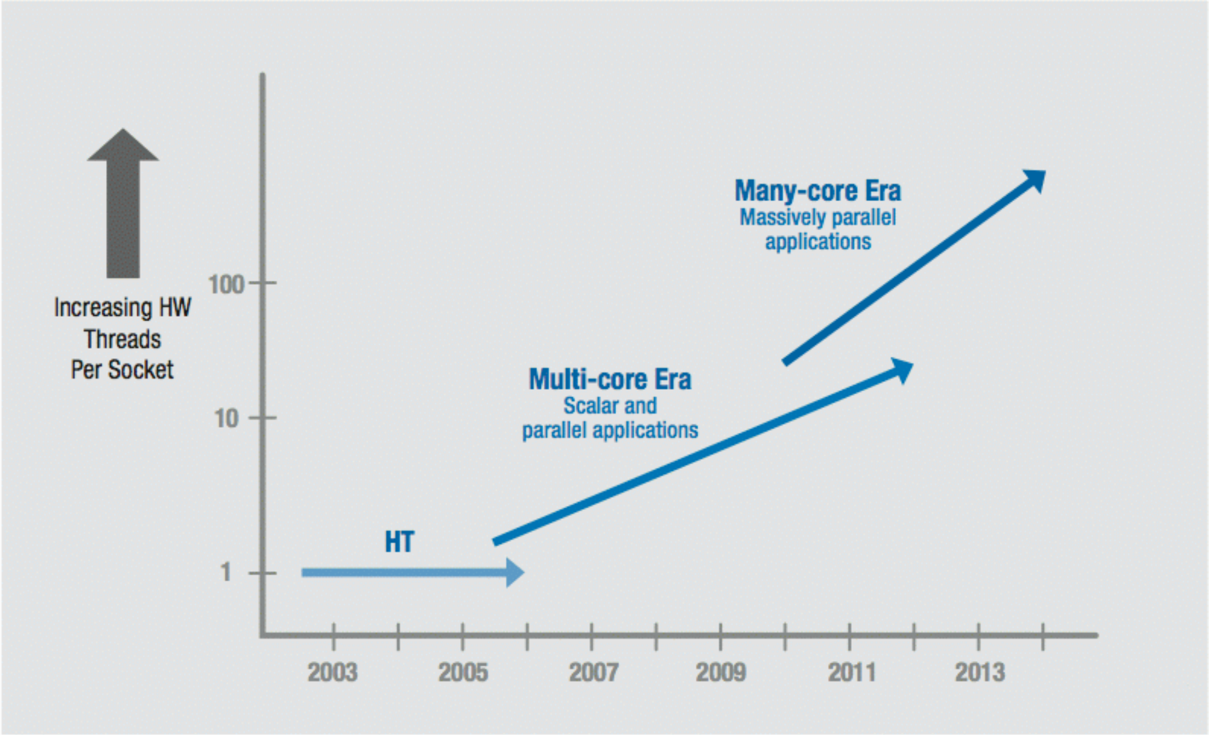
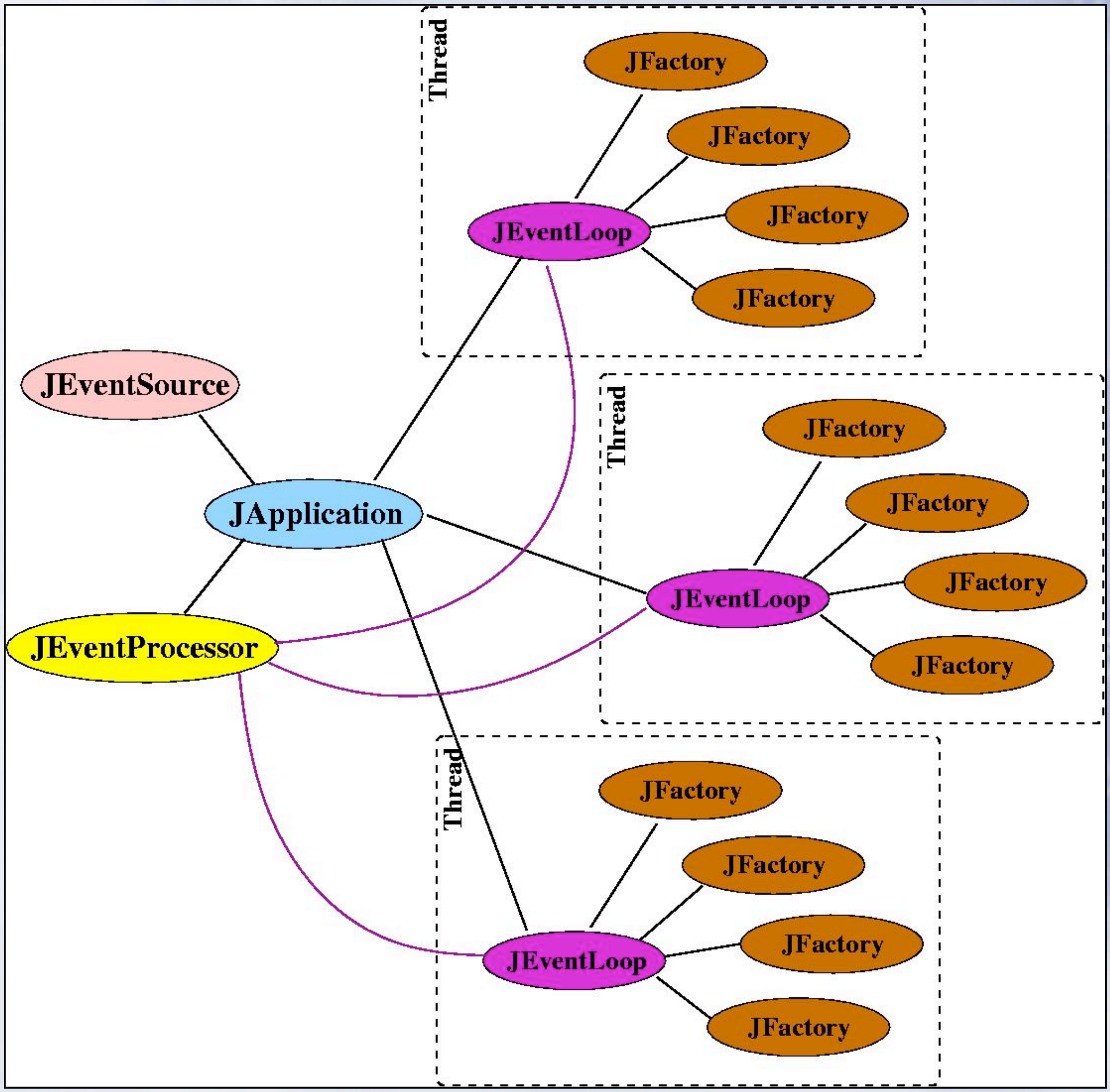


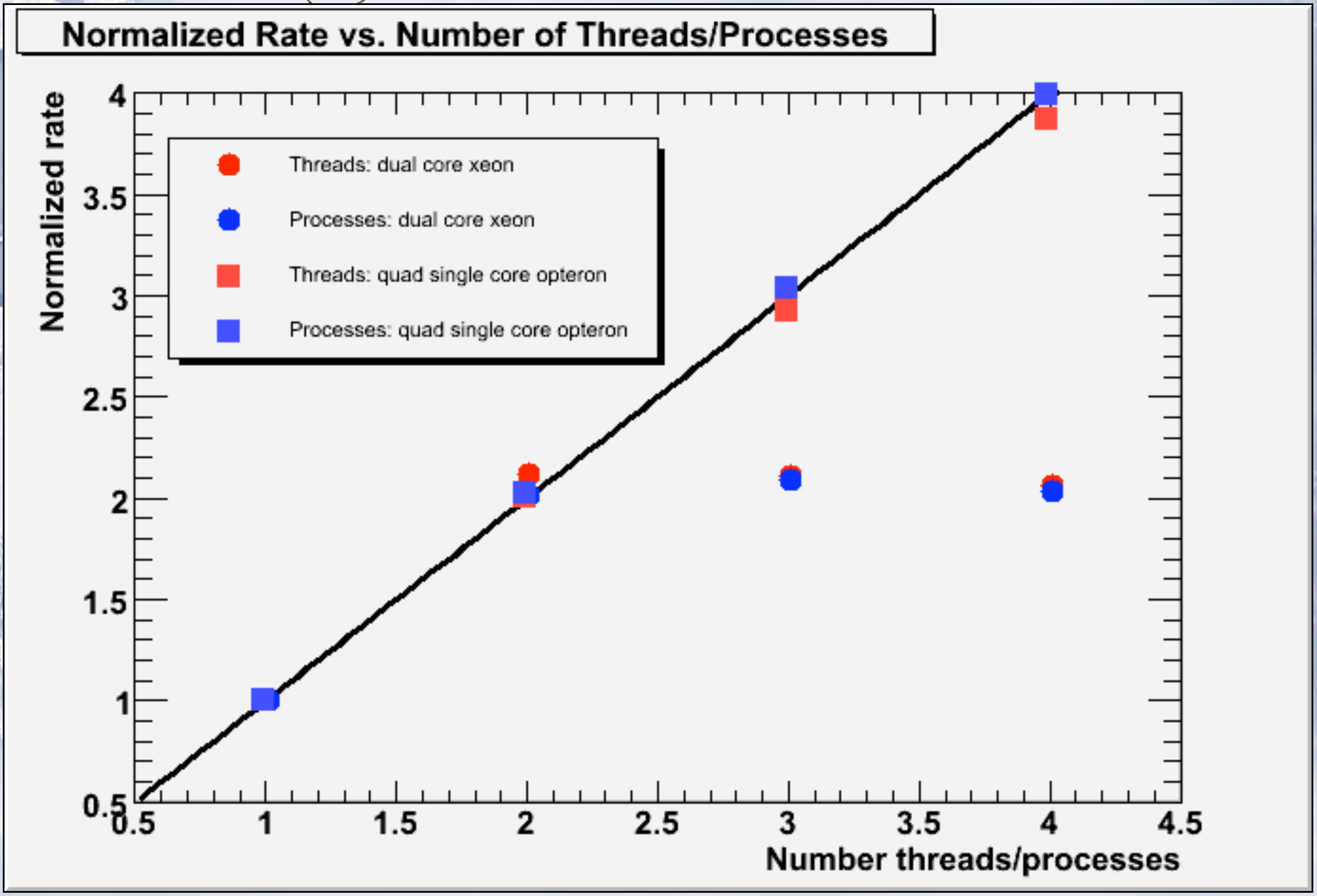
Figure 1: Current and expected eras of Intel® processor architectures

From *“Platform 2015: Intel Platform Evolution for the Next Decade”*

JANA Objects



Mutli-threading



JANA Uses Modern C++ Tools

- Use STL ***vector***

  STL=Standard Template Library
  vector = “smart” array

- Use *const* pointers

  Factories deliver *read-only* data eliminating cross-factory data corruption

- Use templates

  Provides type safety removing need to “cast” pointers which can lead to difficult errors.

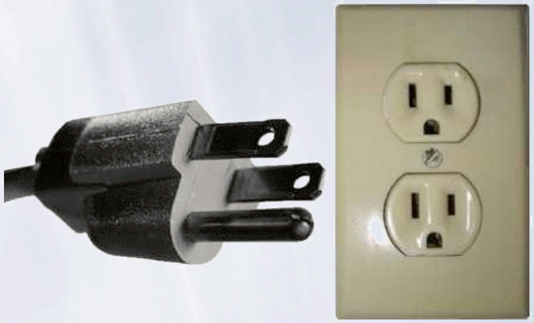
Event Sources in JANA

- JANA programs can read from multiple file types without recompiling
- In Sept. 2006 beam test in Hall-B, JANA programs could read from:
  - ROOT files (Monte Carlo)
  - EVIO files
  - ET system

Event Source API

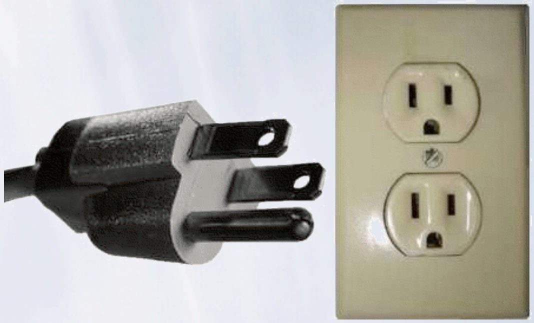
```
JEventSourceGenerator
        const char* Description(void)
        double CheckOpenable(string source)
        JEventSource* MakeJEventSource(string source)
```

```
JEventSource
        jerror_t GetEvent(JEvent &event)
        void FreeEvent(JEvent &event)
        jerror_t GetObjects(JEvent &event
                                    , JFactory_base *factory)
```



Plug-ins

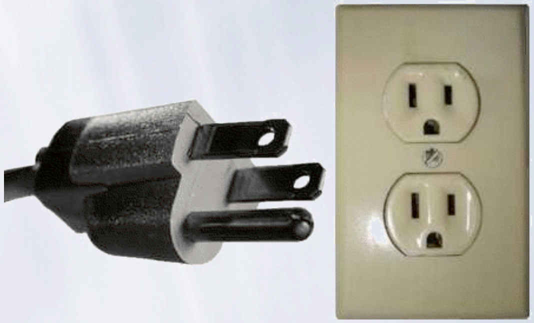
*"A **plugin** (or **plug-in**) is a computer program that interacts with a main (or host) application (a web browser or an email program, for example) to provide a certain, usually very specific, function on-demand."* -- Wikipedia



Plug-ins

JANA has a fairly generic plug-in interface:

```
void InitPlugin(JApplication *app)
```



Plug-ins

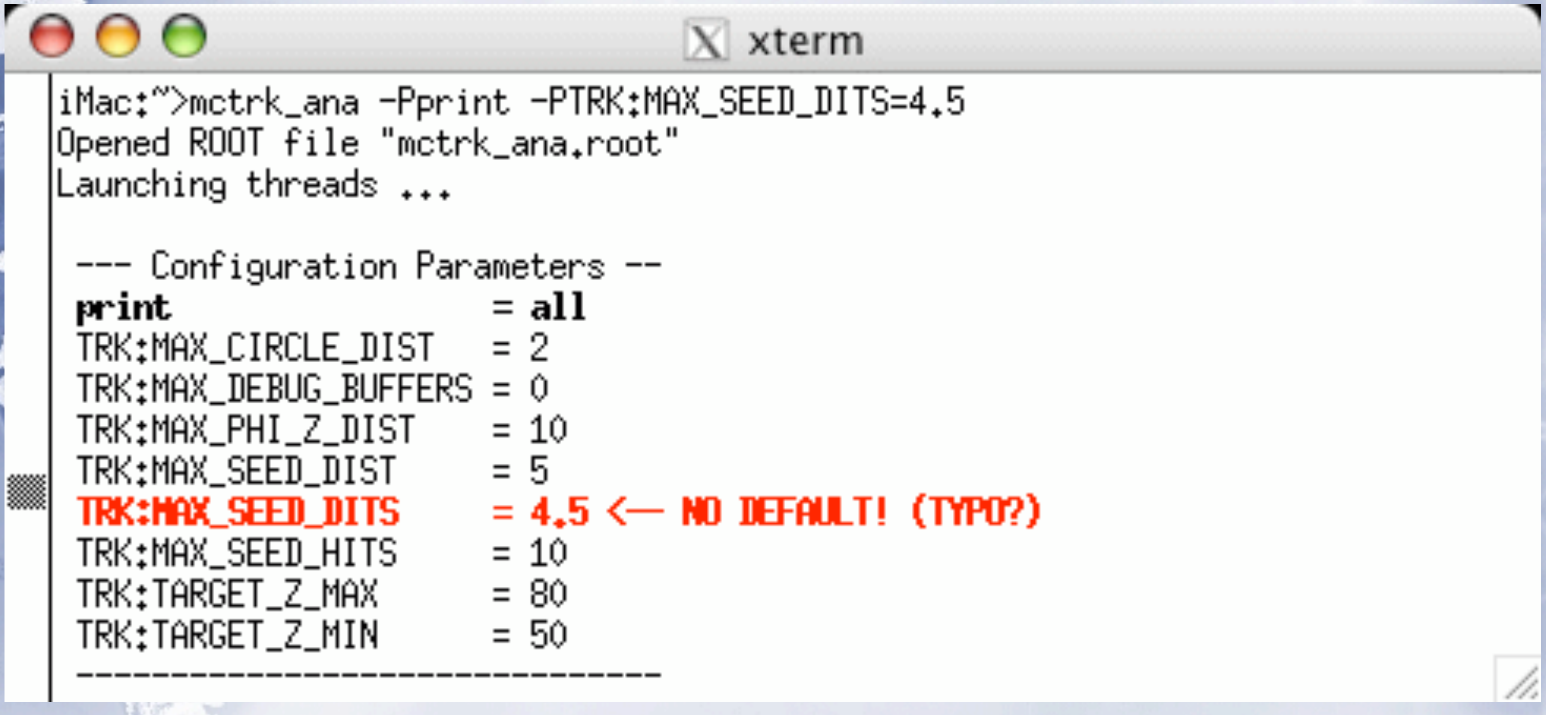
Plug-ins can be used to:

- *Add Event Processors*
- *Add Event Sources*
- *Add Factories*

Configuration Parameters

- A small, temporary database of parameters is built when the program starts
- Parameters can be changed via command line or configuration file*(future)*
- Parameters can be dumped into the DST file to keep a complete record

Configuration Parameters



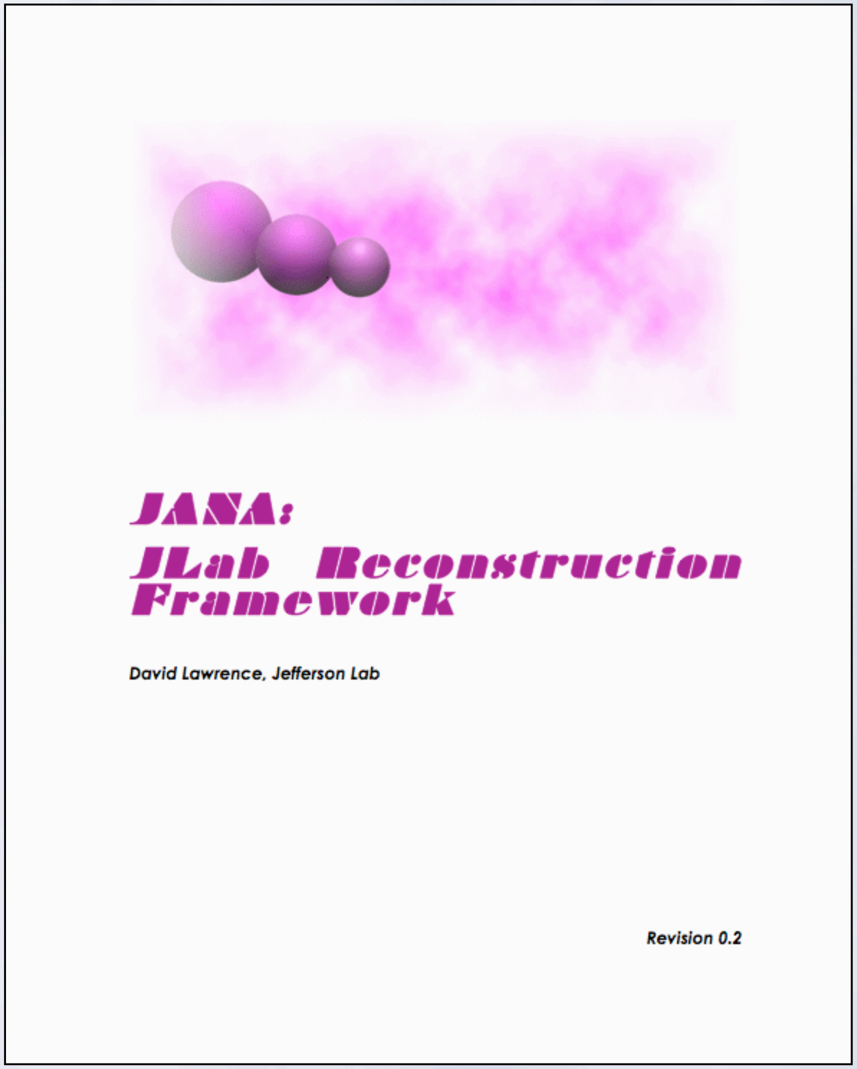
```
iMac:~>mctrk_ana -Pprint -PTRK:MAX_SEED_DITS=4.5
Opened ROOT file "mctrk_ana.root"
Launching threads ...

 --- Configuration Parameters --
 print                  = all
 TRK:MAX_CIRCLE_DIST    = 2
 TRK:MAX_DEBUG_BUFFERS = 0
 TRK:MAX_PHI_Z_DIST     = 10
 TRK:MAX_SEED_DIST      = 5
 TRK:MAX_SEED_DITS      = 4.5 <-- NO DEFAULT! (TYPO?)
 TRK:MAX_SEED_HITS      = 10
 TRK:TARGET_Z_MAX       = 80
 TRK:TARGET_Z_MIN       = 50
 -------------------------------
```

Still to do …

- Standardize exceptions and logging streams
- Configuration files and dumping of configuration parameter database
- More testing, more development, **more ideas** …

http://www.jlab.org/JANA



JANA:
JLab Reconstruction Framework

David Lawrence, Jefferson Lab

Revision 0.2