

JAS - News and Prospects

LC Simulation Meeting
Norman Graf and Tony Johnson

Contents

- Quick Overview of JAS
- Interoperability
 - JAS and Linear Collider Studies
 - JAS and WIRED
 - JAS and AIDA
 - JAS and GEANT4
 - JAS and Grid
 - JAS and Root
- Future Plans

Introduction to JAS

- Pure Java Analysis Environment
 - Data Format Independent
 - Extensible via Plugins/Data Interface Modules
 - Rich Easy-to-use GUI
 - Built in editor/compiler for analysis code
 - Quick and easy to install
 - Snowmass CD – 0 to Physics Analysis in 15 minutes
 - Local and Client-Server Operation
 - Modular – individual components can be taken out and used separately
- Current production release 2.2.4

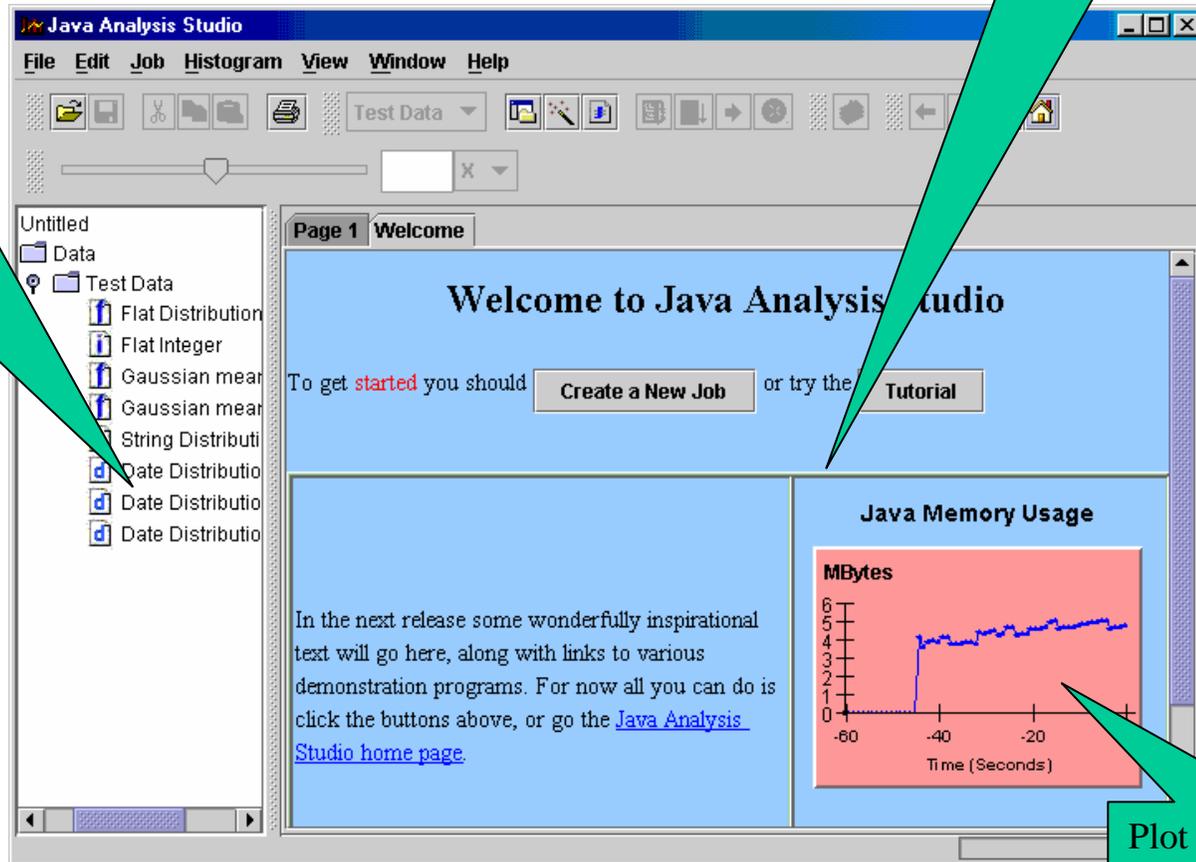
JAS GUI

Built in HTML viewer with embeddable “objects” (buttons, plots, etc).

Tree provides access to analysis objects:

- Histograms
- Plots
- Data Sets
- Analysis Routines

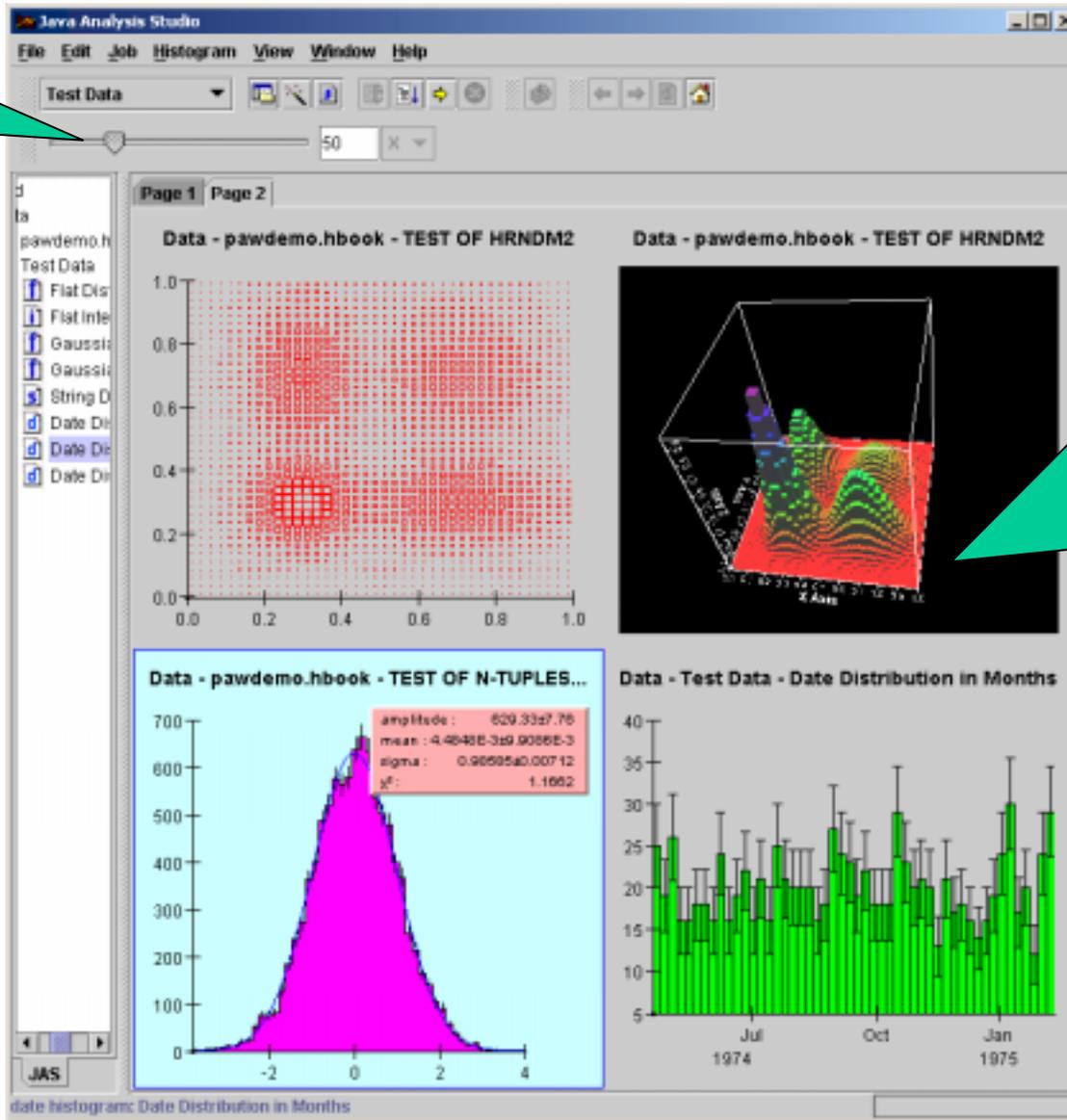
In principle any object. Each item has popup menus and double click action.



Plot Widget, shows data in real time, optimized for fast refresh performance

JAS Plotter

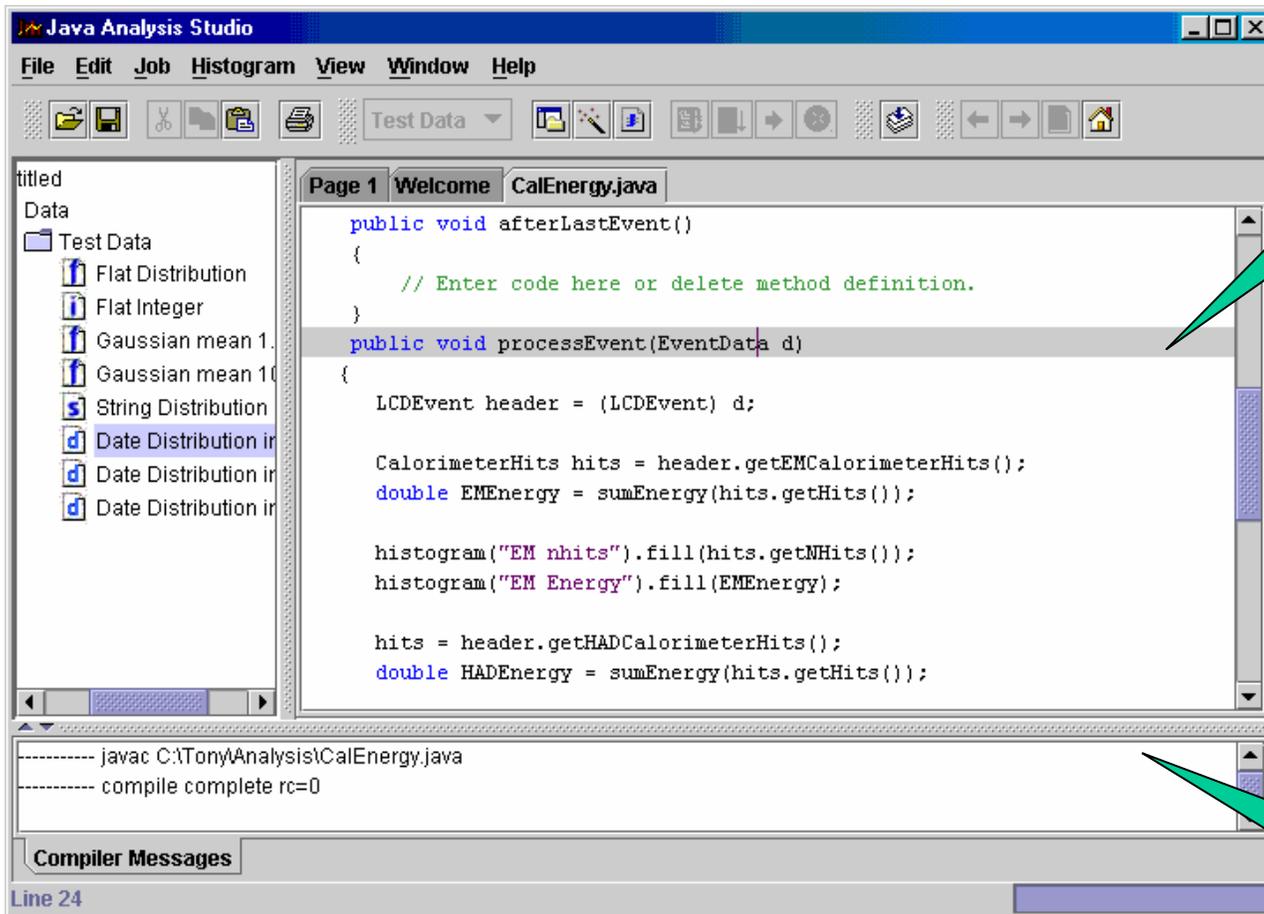
Rebin slider can be used to dynamically change # of bins.



Pages can display histograms. User can control layout, add remove plots, etc.

Plots are highly interactive, can be manipulated by the user by dragging on the axis, or bounding box. Labels (title, legend, axis labels) can be updated by clicking and typing.

JAS Editor/Compiler



Built in code editor
with syntax
highlighting (based
on open-source
Jedit editor)

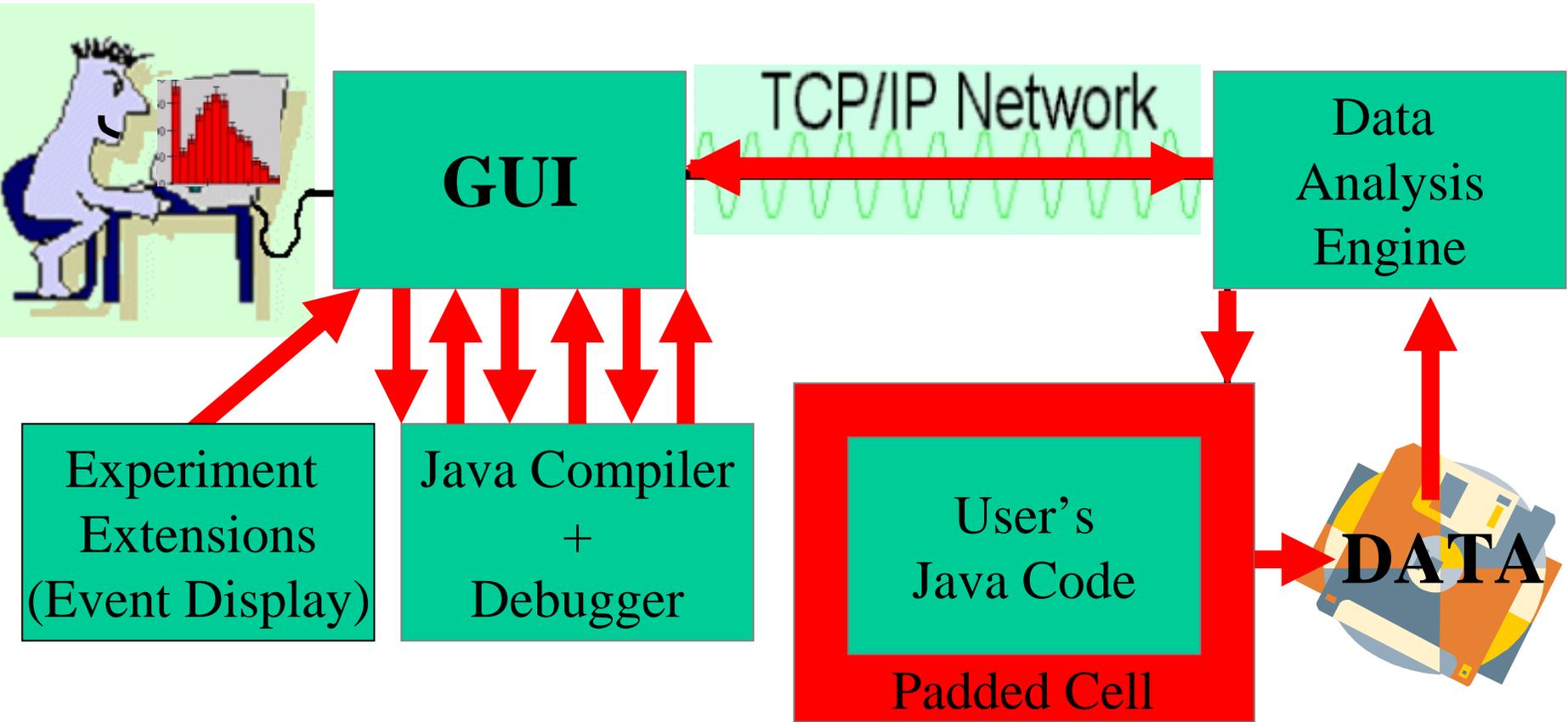
Built-in Java compiler.
Can dynamically load
(and unload) analysis
code.

Data Format Independent



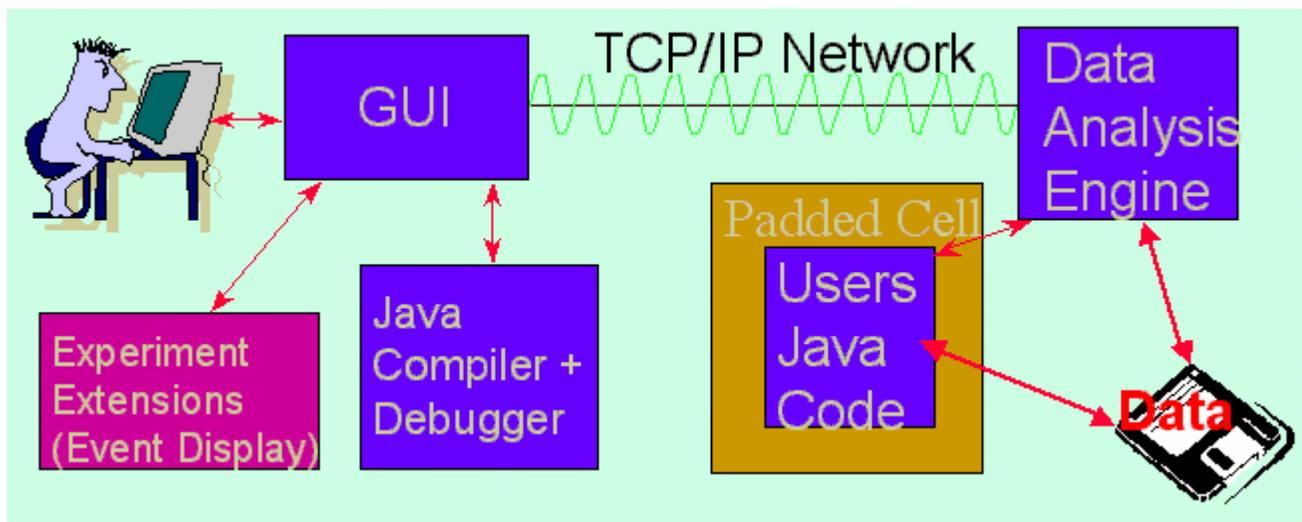
- JAS uses a simple interface (DIM) to isolate it from any particular data format. There are many DIM implementations which make different data formats accessible to the JAS client.
- Flexible design works with many different types of data, from N-Tuples, Database tables (via JDBC) to arbitrary trees of objects.

JAS Remote Data Access



Remote Data Access

- Rather than transporting peta-bytes of data to the physicist
 - Transport the physics analysis code to the data
 - Transparently - so that it feels just like local data access
 - Just ship histogram contents back to the physicists desktop (on demand)
- Allows remote analysis with modest network bandwidth
- Allows user to “feel” as if using local machine even when accessing remote data.



Extensible via Plugins

- Plugins can:
 - Define experiment specific utilities (event display, analysis utilities, specialized tables).
 - Define data interfaces to handle new types of data.
 - Define new plotting routines (e.g. to display special display).
 - Add menus, create control areas, consoles, and output pages.
 - Plugins will be even more flexible in JAS 3.0
 - Entire application will become a set of cooperating plugins in the “FreeHEP application framework”

JAS and Linear Collider

- History
 - JAS has been used for Linear Collider Physics studies in US for 3+ years
- Usage
 - Reconstruction
 - Entire LCD reconstruction program written in Java
 - Track finding+fitting
 - Cluster finding
 - Vertex finding (ZVTop)
 - Fast (parameterized Monte-Carlo)
 - Can be run standalone – or inside JAS
 - No explicit dependence on JAS

JAS + Linear Collider

- Usage cont.
 - IO Formats
 - DIM's created for 2 different data formats
 - SIO – Serial IO – simple XDR based format ideal for interchange with other programs (Geant4, Gismo, Root etc)
 - LCD – random access format optimized for high-performance data analysis
 - Analysis/Reconstruction works equally well with either format (it doesn't see any difference)
 - Remote Data Access
 - Used to allow remote access to large volumes of MC data stored at SLAC and elsewhere.
 - Tools
 - MC particle viewer, event display, ...

LCD Plugins

The screenshot shows the 'MC Tree' view in Java Analysis Studio. The tree structure represents a particle decay event. At the top is 'Run 31752 Event 1', which contains a 'HepEvt' object. This event decays into several particles, including 'sel-' particles and 'e-' particles. Each particle is represented by a small icon and a text label containing its mass, id, charge, energy, and status. The status of each particle is shown in parentheses at the end of the label.

```
Run 31752 Event 1
├── HepEvt (mass=0.0 id=9999999 charge=0(E=0.0 status=3)(gismoStatus=8)
│   ├── sel- (mass=100.0 id=1000011 charge=-1)(E=278.8999938964844 status=2)(gismoStatus=1)
│   │   ├── sel- (mass=100.0 id=1000011 charge=-1)(E=278.8999938964844 status=2)(gismoStatus=1)
│   │   │   ├── e- (mass=5.11E-4 id=11 charge=-1)(E=65.39600372314453 status=1)(gismoStatus=4)
│   │   │   └── e+ (mass=100.0 id=2000011 charge=1)(E=221.10000610351562 status=2)(gismoStatus=1)
│   │   └── sel+ (mass=100.0 id=2000011 charge=1)(E=221.10000610351562 status=2)(gismoStatus=1)
│   │       ├── sel+ (mass=100.0 id=2000011 charge=1)(E=221.10000610351562 status=2)(gismoStatus=1)
│   │       └── e+ (mass=5.11E-4 id=-11 charge=1)(E=16.00099450883594 status=1)(gismoStatus=4)
└──
```

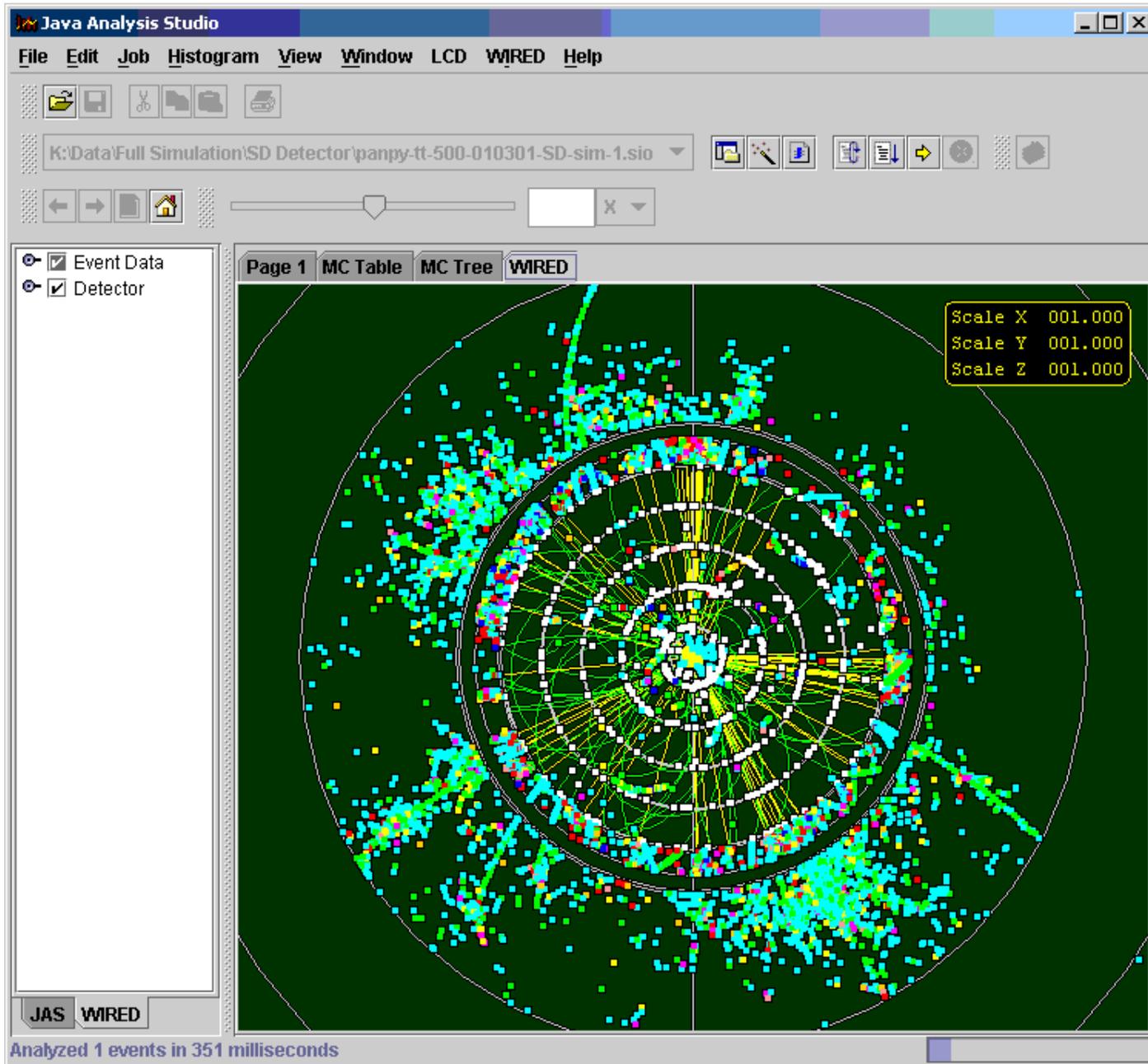
Particle decay tree

The screenshot shows the 'MC Table' view in Java Analysis Studio. It displays a table of particle properties for the event shown in the previous screenshot. The table has columns for 'N', 'Type', 'Status', 'Parent', 'PX', 'PY', 'PZ', and 'E'. The rows represent individual particles, with their properties listed in scientific notation. The status of each particle is shown in the 'Status' column.

N	Type	Status	Parent	PX	PY	PZ	E
0	HepEvt	Document	0	0	0	0	0
1	sel-	Intermediate	-132.78	-82.886	32.989	278.89	
2	sel+	Intermediate	132.78	82.886	-32.989	221.10	
3	sel-	Intermediate 1	-132.78	-82.886	32.989	278.89	
4	sel+	Intermediate 2	132.78	82.886	-32.989	221.10	
5	sel-	Intermediate 3	-132.78	-82.886	32.989	278.89	
6	sel+	Intermediate 4	132.78	82.886	-32.989	221.10	
7	e+	Final State	-10.834	83.094	37.583	65.388	
8	e-	Final State	-4.4884	4.8206	14.884	16.931	
9	gamma	Final State	-1.2188	1.8258	5.9488	5.5493	
10	gamma	Final State	1.0000E-4	-1.8888E-4	5.0000E-4	1.0000E-3	
11	gamma	Final State	-1.0000E-4	1.0000E-4	-5.0000E-4	1.0000E-3	
12	gamma	Final State	-4.0000E-4	1.0000E-4	3.0000E-4	1.0000E-3	
13	gamma	Final State	4.0000E-4	-1.0000E-4	-3.0000E-4	1.0000E-3	
14	gamma	Final State	4.0000E-4	2.0000E-4	-3.0000E-4	1.0000E-3	
15	gamma	Final State	-4.0000E-4	-2.0000E-4	3.0000E-4	1.0000E-3	
16	gamma	Final State	2.0000E-4	2.0000E-4	4.0000E-4	1.0000E-3	
17	gamma	Final State	2.0000E-4	-2.0000E-4	4.0000E-4	1.0000E-3	
18	gamma	Final State	0	3.0000E-4	4.0000E-4	1.0000E-3	
19	gamma	Final State	0	3.0000E-4	-4.0000E-4	1.0000E-3	
20	gamma	Final State	-2.0000E-4	4.0000E-4	-2.0000E-4	1.0000E-3	
21	gamma	Final State	2.0000E-4	-4.0000E-4	2.0000E-4	1.0000E-3	
22	gamma	Final State	-4.0000E-4	2.0000E-4	1.0000E-4	1.0000E-3	
23	gamma	Final State	4.0000E-4	-2.0000E-4	-1.0000E-4	1.0000E-3	
24	gamma	Final State	4.0000E-4	2.0000E-4	1.0000E-4	1.0000E-3	
25	gamma	Final State	-4.0000E-4	-2.0000E-4	-1.0000E-4	1.0000E-3	
26	gamma	Final State	-3.0000E-4	-2.0000E-4	-4.0000E-4	1.0000E-3	
27	gamma	Final State	3.0000E-4	2.0000E-4	4.0000E-4	1.0000E-3	

Particle tabulation

Wired Plugin for LCD



JAS + Wired

- **WIRED**
 - Excellent toolkit for developing very flexible event displays
 - **WIRED** used as LCD event display
 - took only a few days work
 - **WIRED** can now run as a JAS plugin
- Work is well along to merge JAS+WIRED base functionality into FreeHEP Java Library
- JAS (analysis, plotting) + WIRED (event display) will become plugins for “FreeHEP Analysis Studio”.

JAS + AIDA

- AIDA = Abstract Interface for Data Analysis
 - Joint effort between developers of Lizard/Anapne, Open Scientist, JAS, FreeHEP, and others.
 - Aims to provide common set of abstract interfaces for:
 - Histogramming
 - N-Tuples
 - Fitting
 - Plotting
 - Will allow analysis code to run unaltered with different analysis tools
 - Will allow interchange of analysis objects (eg histograms) between tools
 - JAS 3 (future) aims to be completely “AIDA compliant”
 - JAS 2 (present) is phasing in AIDA support
 - Import/export of AIDA Histograms, Ntuples via XML
 - N-Tuple tools based on AIDA interfaces

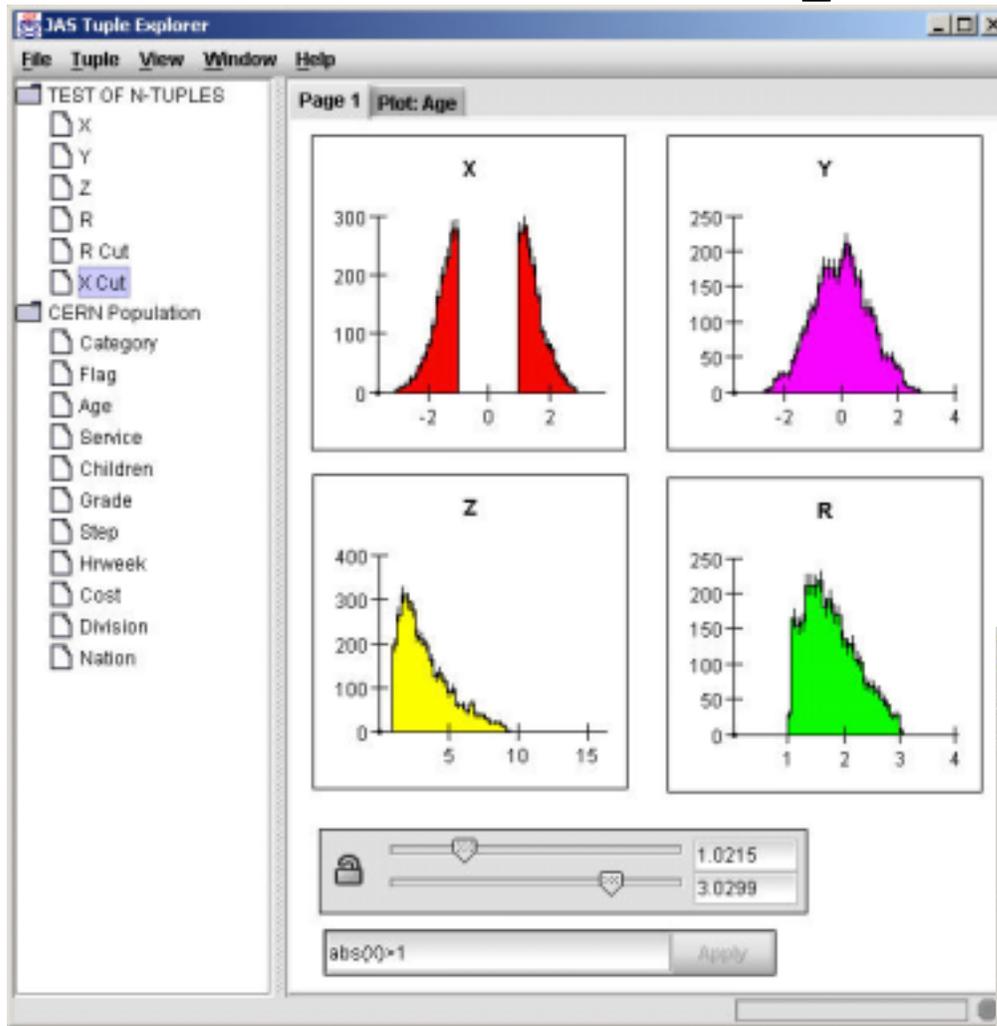
JAS Tuple Explorer

- AIDA compliant
 - Based on AIDA N-tuple
 - Will be able to exchange data/plots with other AIDA applications
- Simpler tool for first time users
 - No remote data access
 - Only support N-Tuple access
- Runs now as standalone application
 - Will also run as plugin in full JAS 3.0
 - Based on JAS 3.0/FreeHEP application framework

JAS Tuple Explorer - Features

- Many data formats supported
 - PAW files, Root files, SQL Database, ASCII text files
 - Uses JAS DIM interface – so easy to add new formats
 - Supports very large (too big for memory) Ntuples
 - Columns may be int, float, double, boolean, String, Date, Object...
- Built-in expression compiler/evaluator
 - Allows new columns to be defined on the fly
- 1D, 2D histograms, profile plots, scatter plots
 - Multiple overlays on same plot
- Dynamic Cuts – based on expression or sliders
- N-tuple Tabulation – supports large N-tuples
- Plots and Cuts can be arranged on page
 - Drag and drop for easy page building
- Highly extensible – easy to add new cuts/plot types/etc

JAS Tuple Explorer



Define New Column dialog box. Column Name: R. Expression: $\sqrt{X^2+Y^2}$. Buttons: OK, Cancel, Help. A tooltip indicates 'Expression to be evaluated'.

Add Cut... dialog box. Numeric 1D Cut | General Cut | Cut Set. Cut Name: Rich Swiss. Expression: `Nation.equalsIgnoreCase("CH") && Cost>20000`.

Cut: defaultCuts dialog box. Show Cut Details. Add... Remove New... R Cut: Disable Invert. X Cut: Disable Invert. Expression: `abs(X)>1` Apply.

More Info:

• <http://jas.freehep.org/documentation/TupleExplorer/>

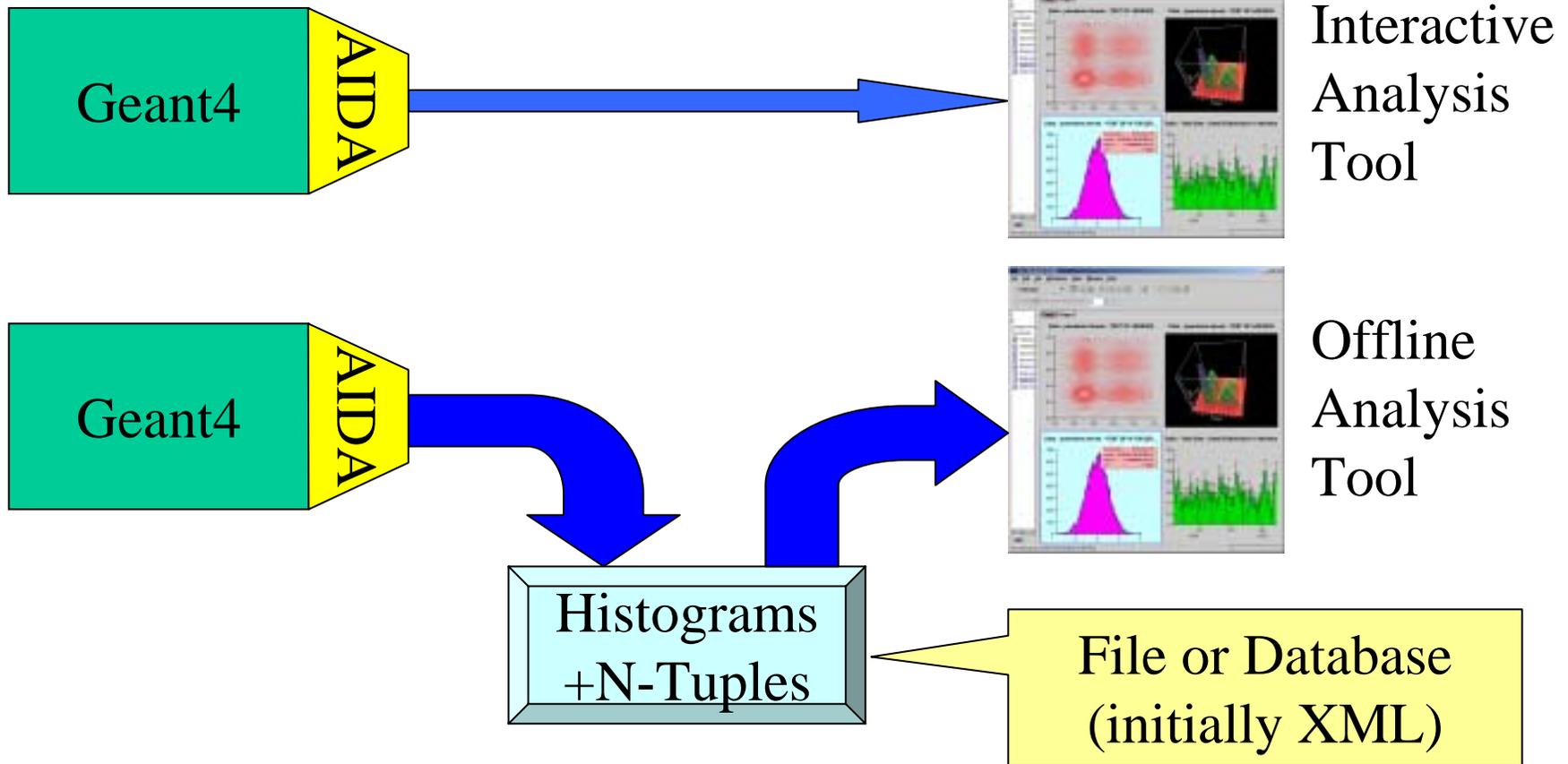
JAS Tuple Explorer

The screenshot shows the JAS Tuple Explorer application window. The title bar reads 'JAS Tuple Explorer'. The menu bar includes 'File', 'Tuple', 'View', 'Window', and 'Help'. On the left, a tree view shows a folder 'TEST OF N-TUPLES' containing sub-items 'X', 'Y', 'Z', 'R', 'R Cut', and 'CERN Population'. Below 'CERN Population' are individual attribute folders: 'Category', 'Flag', 'Age', 'Service', 'Children', 'Grade', 'Step', 'Hrweek', 'Cost', 'Division', 'Nation', and 'Age Cut'. Below these is 'SQL Dataset 1' with attributes 'ID1', 'id', 'age', and 'height'. The main area displays 'Page 3' of the 'CERN Population' table. The table has 13 columns: Category, Flag, Age, Service, Children, Grade, Step, Hrweek, Cost, Division, Nation, and Age Cut. The data rows are as follows:

Category	Flag	Age	Service	Children	Grade	Step	Hrweek	Cost	Division	Nation	Age Cut
341	15	56	21	1	10	5	40	10757	EF	FR	false
412	11	60	20	0	7	9	40	6773	LEP	FR	false
202	13	47	20	2	9	5	40	8729	LEP	FR	false
424	13	51	20	2	6	13	40	6655	SPS	FR	true
340	15	52	20	1	8	13	40	9064	ST	FR	true
511	15	48	20	1	10	5	40	10757	DD	CH	false
566	15	55	20	0	5	13	40	6293	FI	FR	false
532	12	41	20	2	6	11	40	6474	FI	FR	false
497	13	58	21	0	5	13	40	5720	ST	FR	false
530	15	51	20	2	8	12	40	9183	PE	BE	true
341	15	45	20	1	9	4	40	9103	SPS	FR	false
402	15	62	21	0	7	12	40	7487	PS	IT	false
304	13	44	20	3	7	11	40	7497	LEP	FR	false
402	13	50	21	1	7	12	40	7096	LEP	FR	true
208	15	44	20	1	9	6	40	9409	PS	FR	false
300	9	50	20	0	8	9	40	7345	ST	CH	true
411	15	52	21	0	7	6	40	6820	ST	FR	true
530	10	48	20	0	7	11	40	6990	LEP	FR	false
500	15	51	20	0	12	10	40	14847	LEP	FR	true
417	15	58	21	0	6	11	40	6544	ST	FR	false
204	15	46	20	0	10	1	40	9771	EP	SE	false
201	15	46	20	3	9	6	40	9911	SPS	AT	false
204	15	46	20	2	12	3	40	13643	EF	FR	false
543	12	42	20	1	7	3	40	6185	LEP	FR	false
414	15	48	21	1	7	9	40	7405	LEP	FR	false
202	15	45	20	3	10	9	40	11992	SPS	BE	false
300	11	46	20	0	8	5	40	7366	EF	CH	false

JAS+AIDA+Geant 4

- Geant 4 will use AIDA to allow it to interoperate with any AIDA compliant analysis tool.



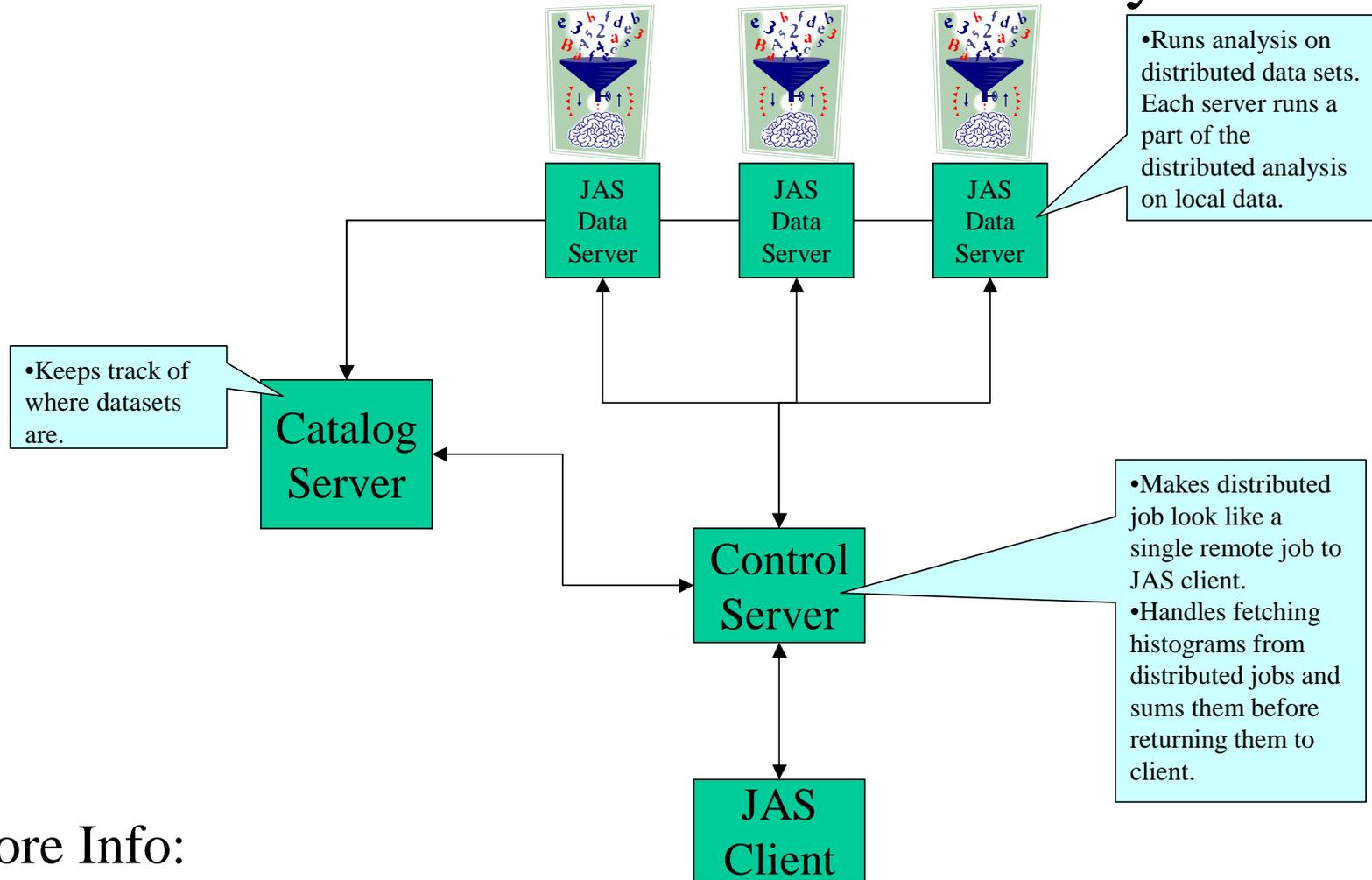
- Similar model could work for any simulation or reconstruction program

JAS + GRID

- Prototype system for distributing analysis on a farm of machines is now available
 - Increase throughput by distributing data across many machines
 - Increase processing power by exploiting many CPU's
- Future Plans
 - Integrate with GRID services including
 - Distributed authentication services
 - Data catalog services
 - Resource locator services
 - Working with others working on similar projects
 - For example “BlueOx” from Jeremiah Mans

Distributed Analysis

- Current prototype allows interactive analysis to be performed on ~50 nodes simultaneously.



More Info:

- <http://jas.freehep.org/documentation/DistributedComputing/>

JAS+Root

- Root Data Interface Module (DIM) exists
 - Allows Root Data to be read and analyzed in JAS
 - Files must be written with Root 3.0 or later
 - Full support for user-defined objects
 - Based on Java implementation of Root IO from FreeHEP Java library
 - Current implementation is slow, *much* faster version will be available by the end of 2001

Root Object Browser (JAS Plugin)

The screenshot displays the Java Analysis Studio (JAS) interface. The title bar reads "Java Analysis Studio". The menu bar includes "File", "Edit", "Job", "Histogram", "View", "Window", "Root", and "Help". The toolbar contains various icons for file operations and navigation. Below the toolbar is a search bar with the text "An example of a ROOT tree" and a search icon. The main workspace is split into two panes:

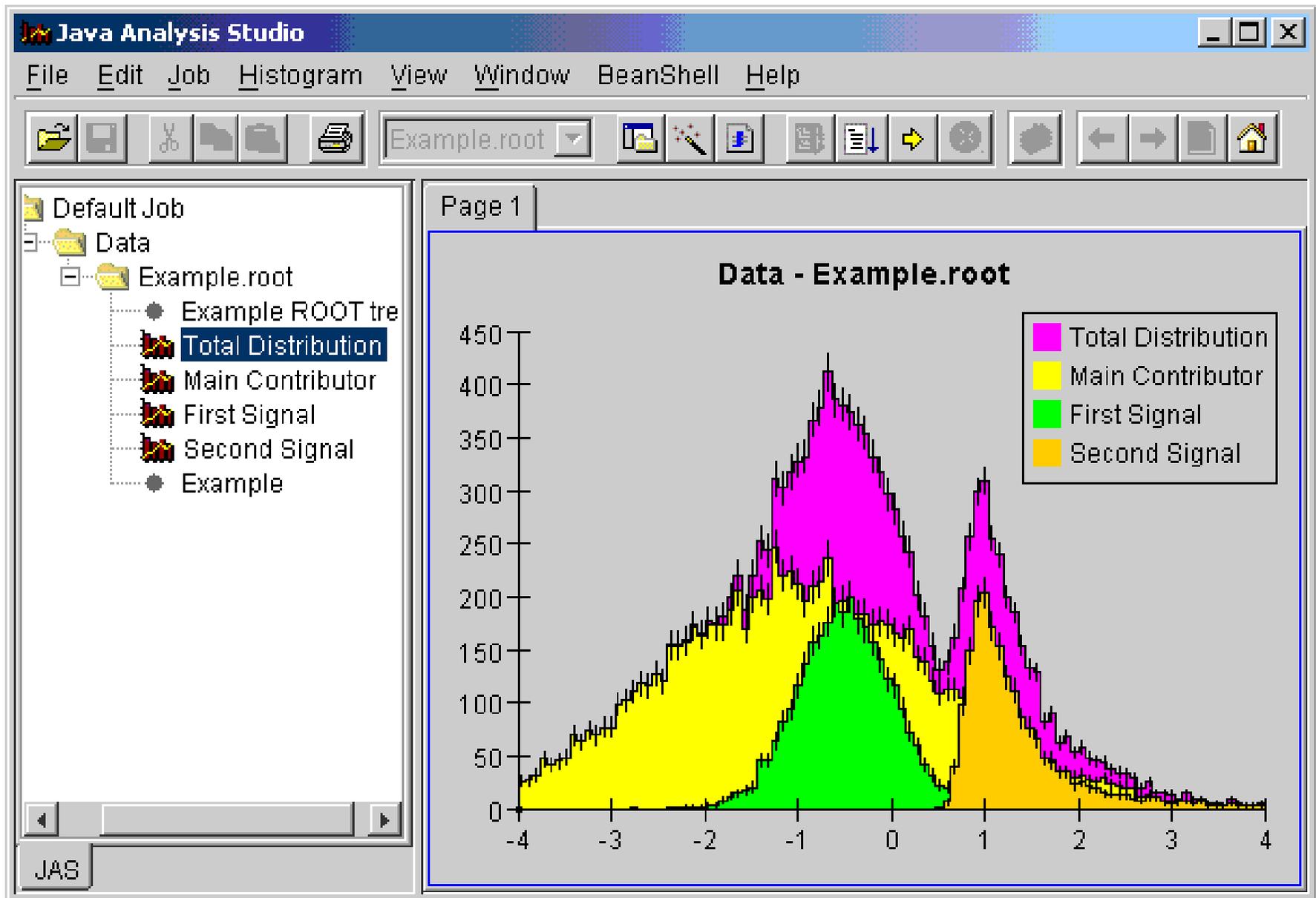
- Left Pane:** A project tree showing the following structure:
 - Default Job
 - Data
 - Event.root
 - Real-Time to write versus time
 - An example of a ROOT tree
 - event
 - Event Histogram

- Right Pane:** A detailed view of an object's fields, titled "Page 1 Welcome Event Browser". The object is "An example of a ROOT tree (entry # 0)". The fields are:
- event (Class Event)
 - Class TObject
 - fType (Array)
 - fNtrack = 594
 - fNseg = 5904
 - fNvertex = 3
 - fFlag = 0
 - fTemperature = 20.318974
 - fEvtHdr (Class EventHeader)
 - fTracks (List)
 - [0] (Class Track)
 - Class TObject
 - fPx = 1.9158666
 - fPy = 1.3706384
 - fPz = 2.3556728
 - fRandom = 318.97293
 - fMass2 = 4.5
 - fBx = 0.051961213
 - fBy = -0.02772567

JAS

Analyzed 1 events in 0 milliseconds

Root Histogram Viewer



JAS Future

- JAS 3.0
 - Will use FreeHEP/plugin architecture
 - Will introduce:
 - Fully AIDA compliant analysis environment
 - Scripting – probably using Jython
 - Tuple Explorer integrated into JAS
 - Improvements to JAS 2.2.x will continue in parallel to JAS 3 development
- Manpower
 - Max Turri (SLAC computing services) has now joined JAS team (approximately full-time)
 - Anticipate further increases in manpower in near future
 - Modular design enables us to take advantage of contributions from FreeHEP project, Babar, LCD and others.

Conclusions

- JAS 2.2.x is a stable release
 - Used extensively for US linear collider physics studies
 - JAS + Java makes excellent rapid design tool for studies of analysis and reconstruction strategies
- Many new features coming in JAS 3.0
 - Will combine FreeHEP + JAS + WIRED
 - Continue to emphasize inter-operability with other programs
- More Info:
 - <http://jas.freehep.org>