... for a brighter future

A U.S. Department of Energy laboratory
managed by The University of Chicago

Eclipse, Java, Scientific Software, Etc.

Kenneth Evans, Jr.

Presented at the European Synchrotron Radiation Facility ESRF

May 3, 2007

Grenoble, France

Outline

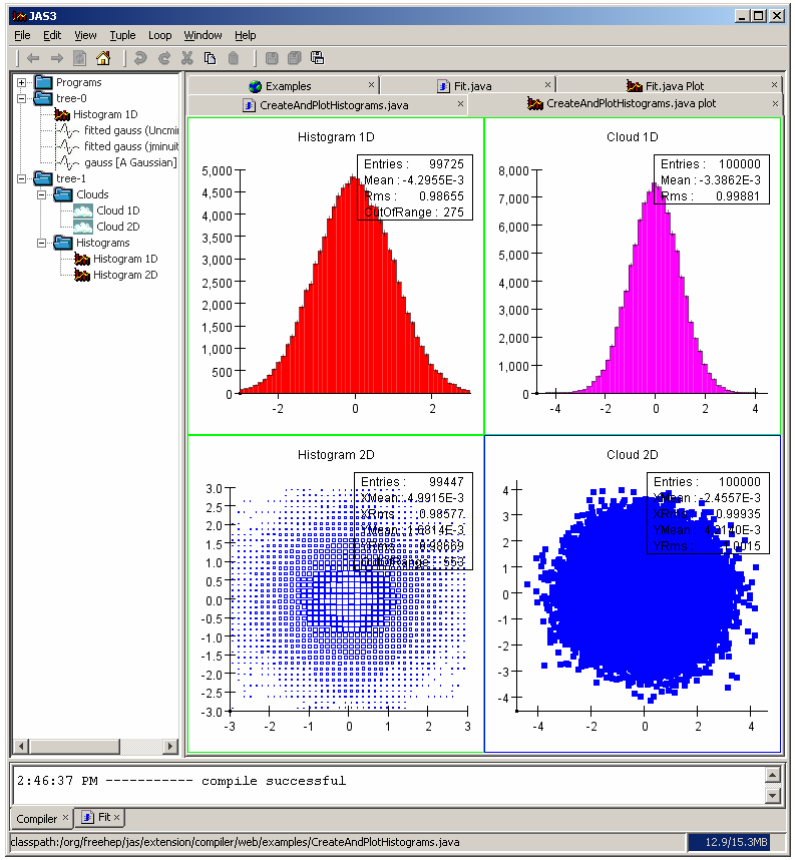
- Scientific Software and Examples
- Java
- Eclipse and Examples
- Eclipse RCP Applications
- AWT and SWT
- X-Ray Software Development at the APS

Scientific Software

- The language of choice used to be FORTRAN
  - There are still many legacy FORTRAN codes in use
- C and C++ have become popular
  - Grid computing now tends to be done in C
- Many scientists use Python
  - Reasonably powerful, yet easy to use
  - Allows them to do science rather than software
- There are now a number of significant scientific projects using Java
  - Many started out as C, but have evolved to Java
- Java is now an acceptable, if not the preferred, language for scientific software development

Java Analysis Studio (JAS3)

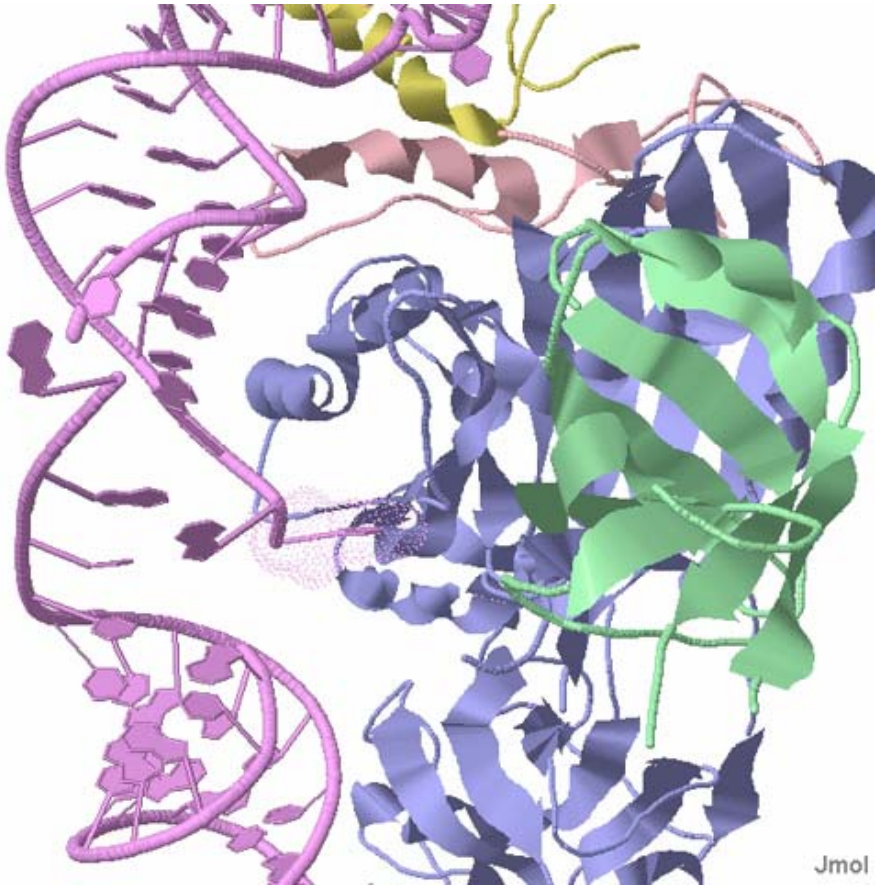
- Developed by and for the High-Energy physics community
- Plotting of 1d, 2d, 3d Histograms, XY plots, Scatter plots, *etc*.
- Open source
- Attractive plotting
- Fitting, other mathematical analysis
  - Primarily from CERN
- Highly modular structure
  - Uses plug-ins



JMol – Molecular Viewer

- Commonly used as an applet that can be integrated into web pages to display molecules in a variety of ways
- Also has a standalone application and a development tool kit that can be integrated into other Java applications
- Interactive, 3D
- Free, Open Source

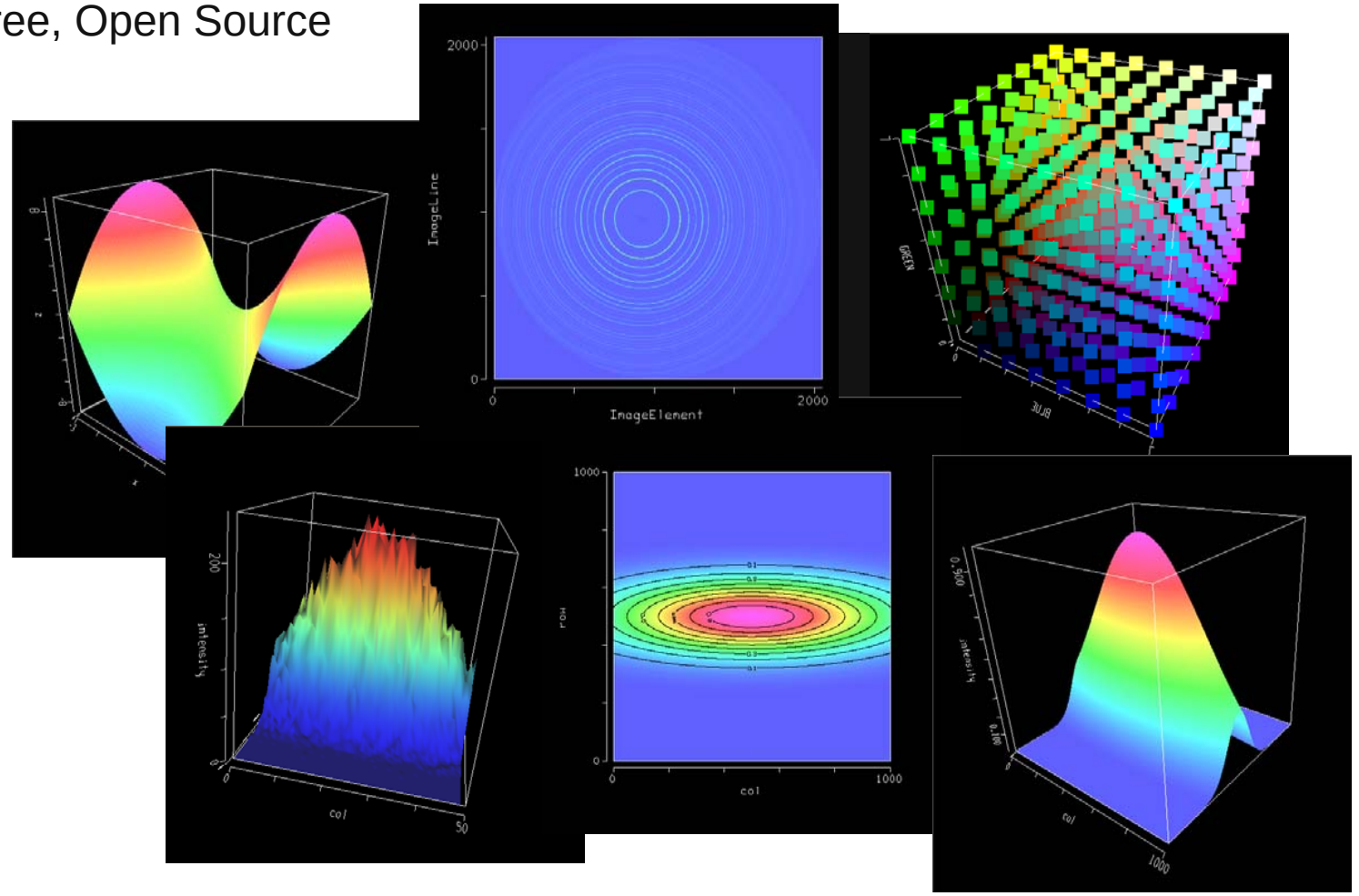
- One of several Java Molecular Graphics packages



Crystal structure of an H/ACA box RNP from Pyrococcus furiosus (PDB CODE: 2HVY)

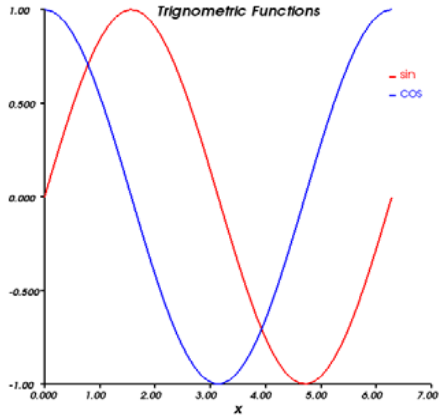
VisAD

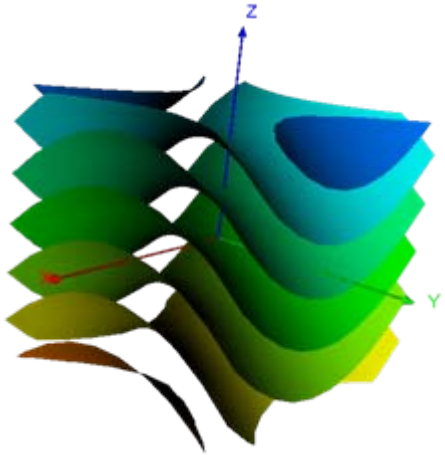
- Space Sciences and Engineering Center (SSEC) and others
- Extensive 2D and 3D visualization package
- Free, Open Source

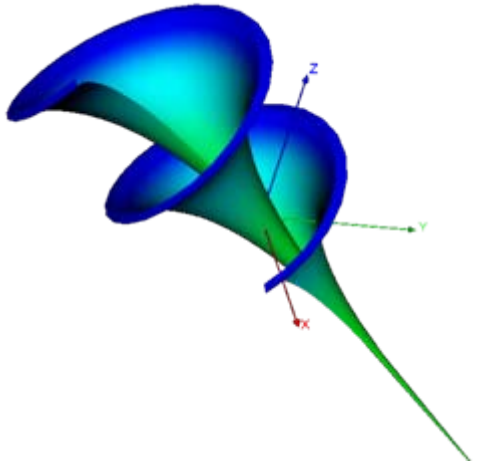


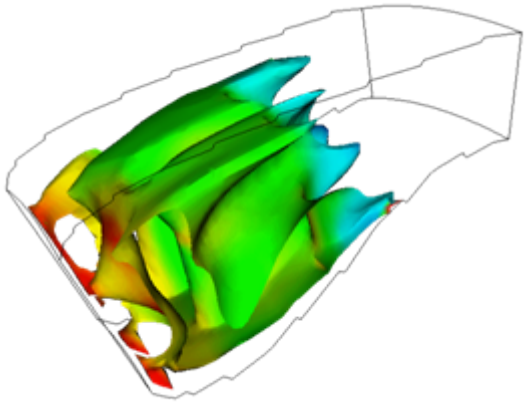
VTK

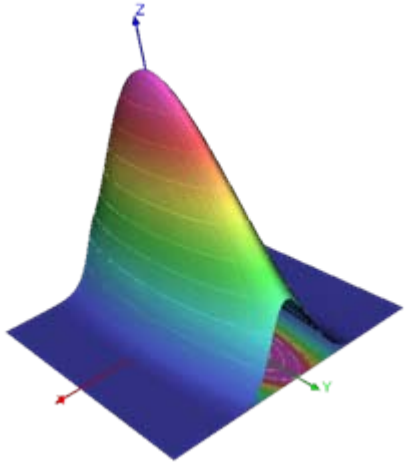
- Software system for 3D computer graphics, image processing, and visualization
- Used by thousands of researchers and developers around the world
- Written in C++
- Has Java wrappers
  - Also, Tcl/Tk, Python
- Free, Open Source





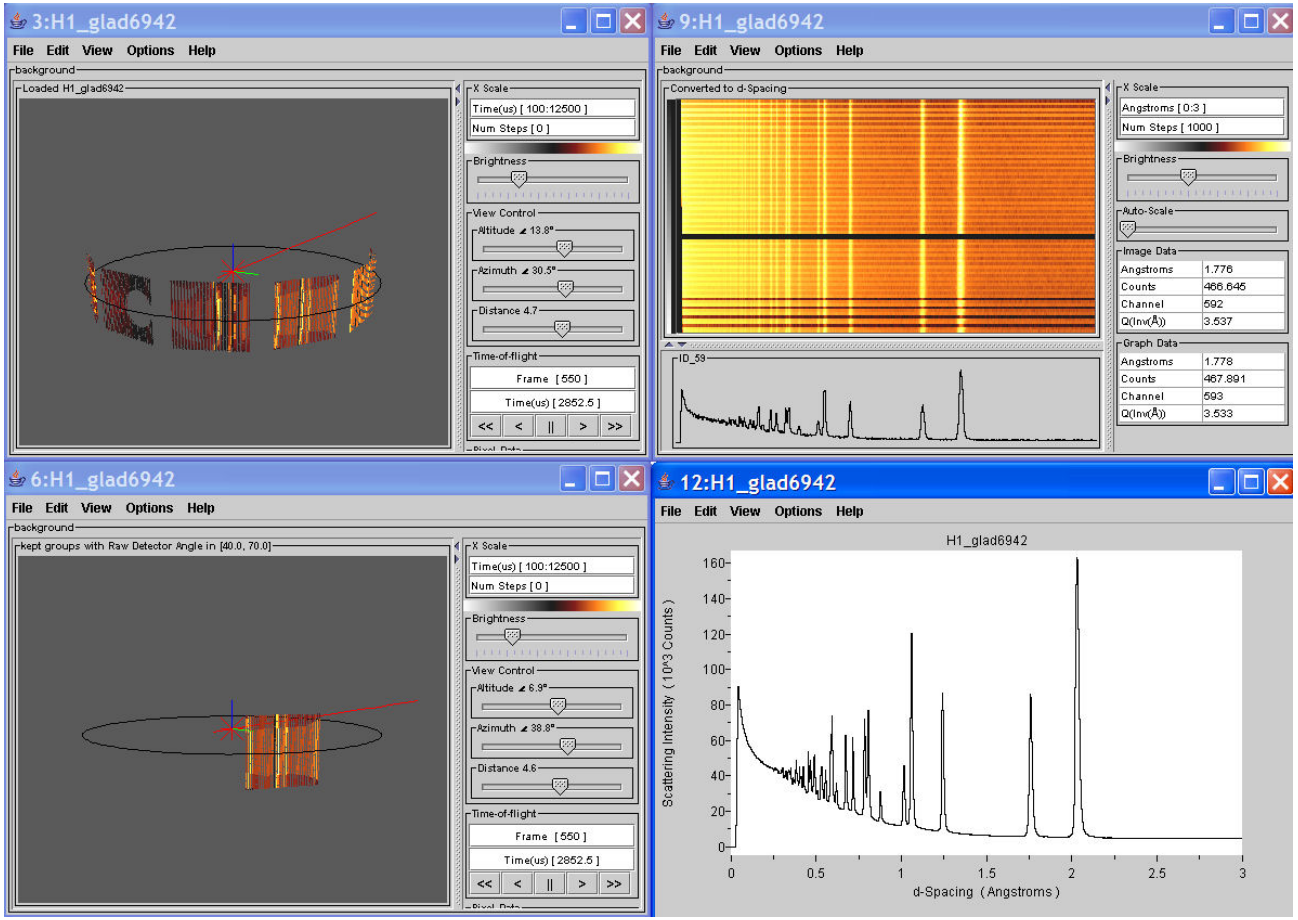






ISAW

- The primary tool for analyzing neutron scattering data at the IPNS
- Has an extensive and sophisticated interface



From: John Hammonds, IPNS

Java ?

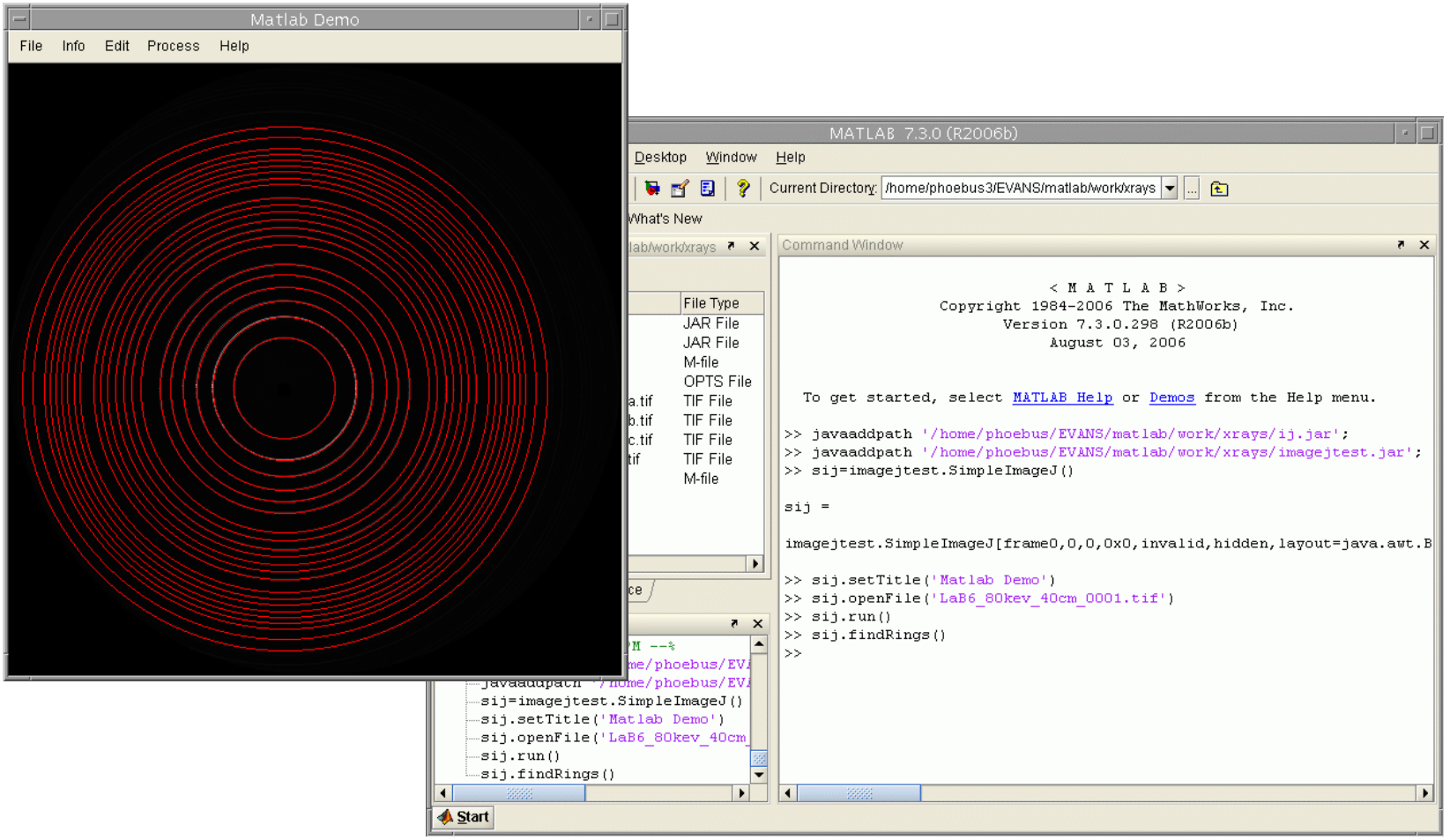
- Java has become a major language
- The reason is that most commercial development uses J2EE
  - There is money to be made improving Java and its tools
- Applications have performance approaching applications written in C
- There is already extensive scientific development in Java
- In my opinion, there is no other viable choice for high-quality, cross-platform, GUI development
  - Huge API
  - Write once, run anywhere
  - Easy to code (compared to C or C++, anyway)
  - Good performance
  - Excellent development tools

Java Development Tools

- Spell checks as you go
  - No “write – compile – load – run – figure out what happened” cycle
  - Probably the one most significant productivity enhancement
- Provides content assist
  - Probably the next most significant productivity enhancement
- Compiles as you write
  - Cycle is now “write – run”
- Massive refactoring
  - e.g. Change a variable name in all your files in all your projects
- Wizards and Tools to help at every stage
  - e.g. Generate getters and setters for all your properties
  - e.g. Add and/or clean up imports
- The above are just a small sample
  - Some of these are available for other languages
  - But usually not at the level they are for Java

Java in Matlab

- Matlab has extensive support for Java
  - Your favorite software framework can also be used in Matlab



Eclipse

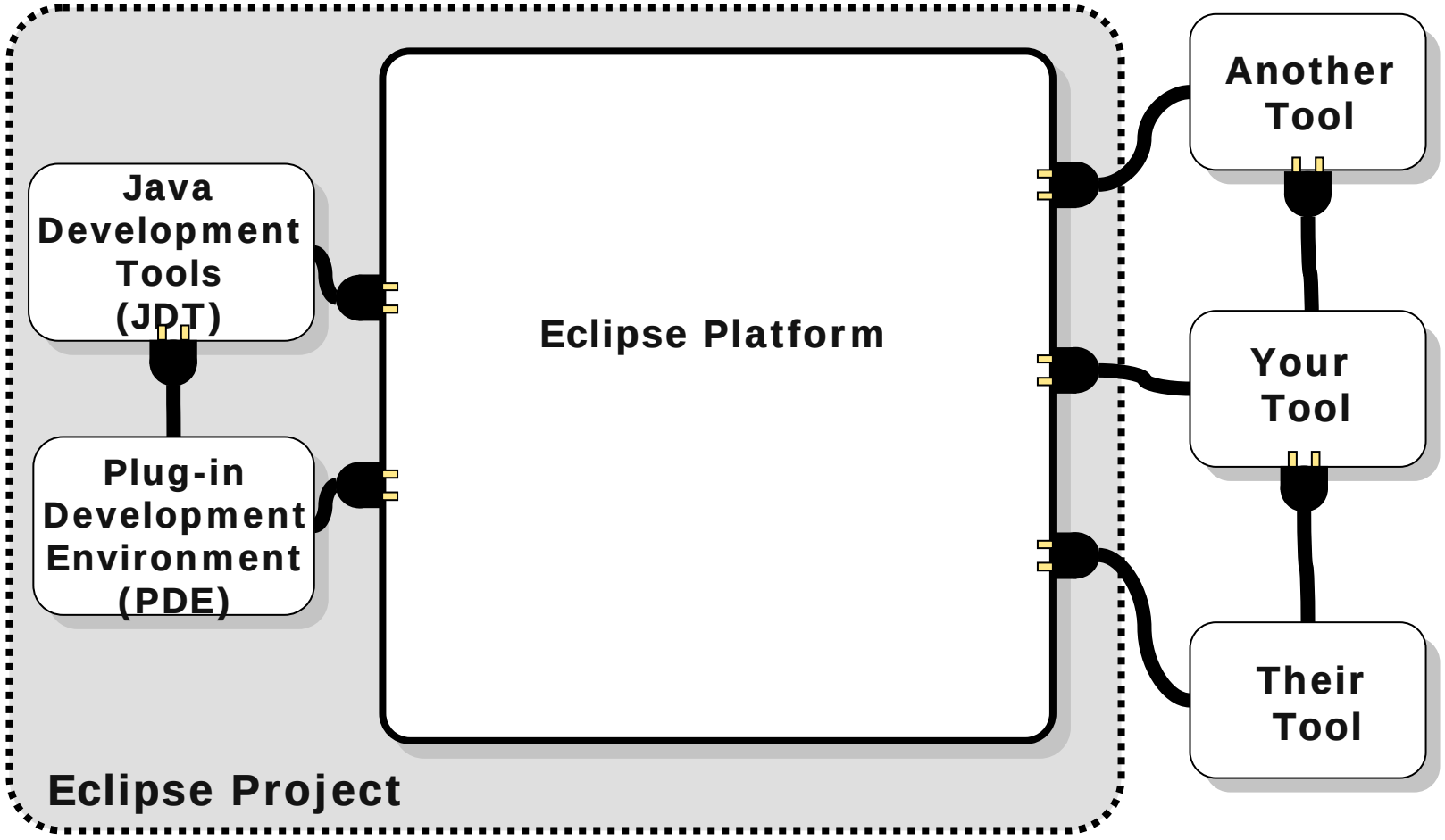
- Eclipse is an Open Source community
- It was started in 2001 by IBM
  - IBM donated a lot of research
  - Controlled the early development, but later relinquished control
- It is now controlled by the Eclipse Foundation
  - Strategic members contribute up to $500K and 8 developers
  - Currently 17 strategic members
  - Currently more than 150 developers

- Out of the box it looks like a Java IDE (Integrated Development Environment)
- It is really a Plug-in manager
  - That happens to come with Java Development plug-ins
  - You can make it be most anything you want

Eclipse Consortium Strategic Members

* Strategic Consumer

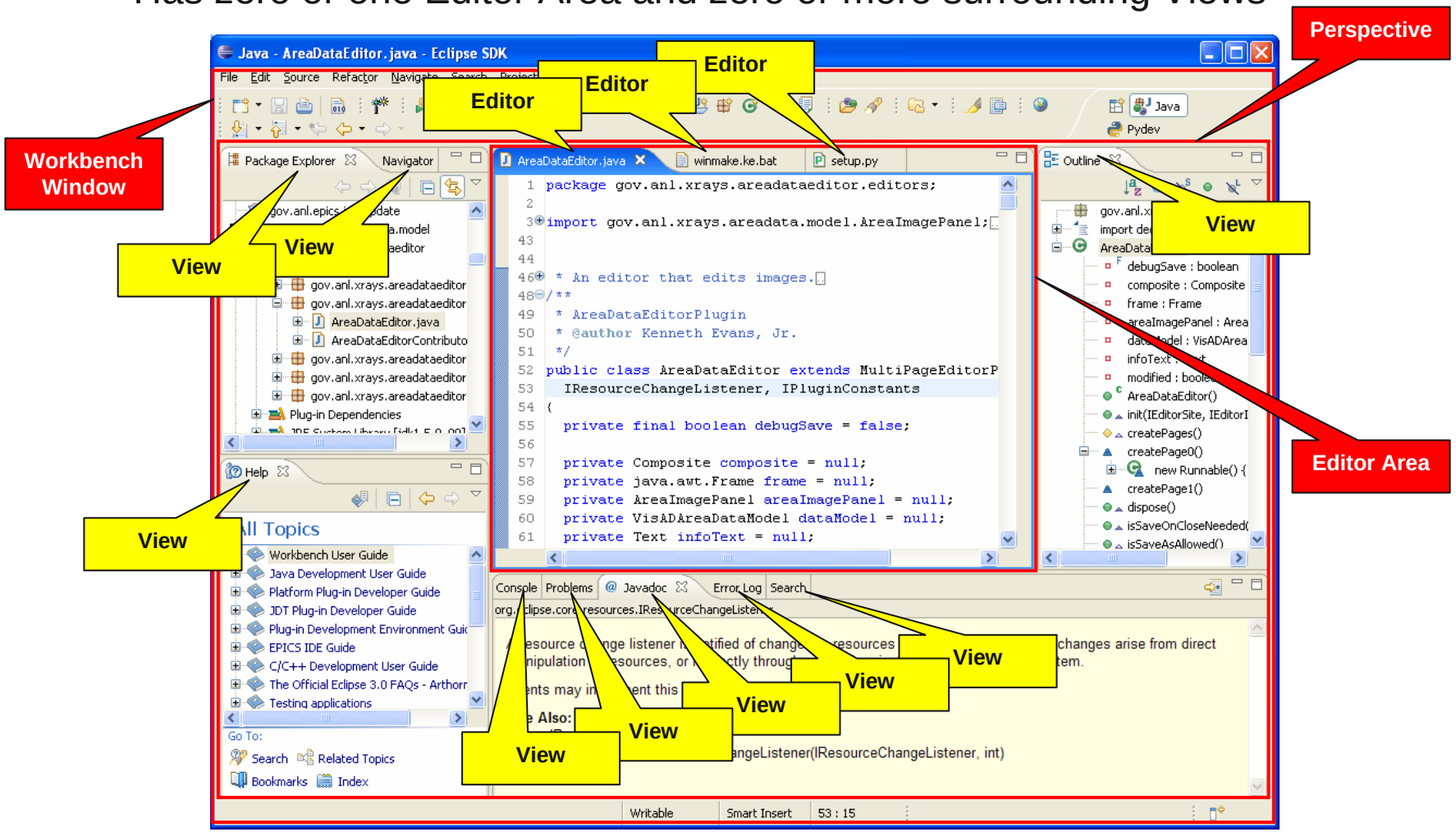
Eclipse is Very Extensible and Very Flexible



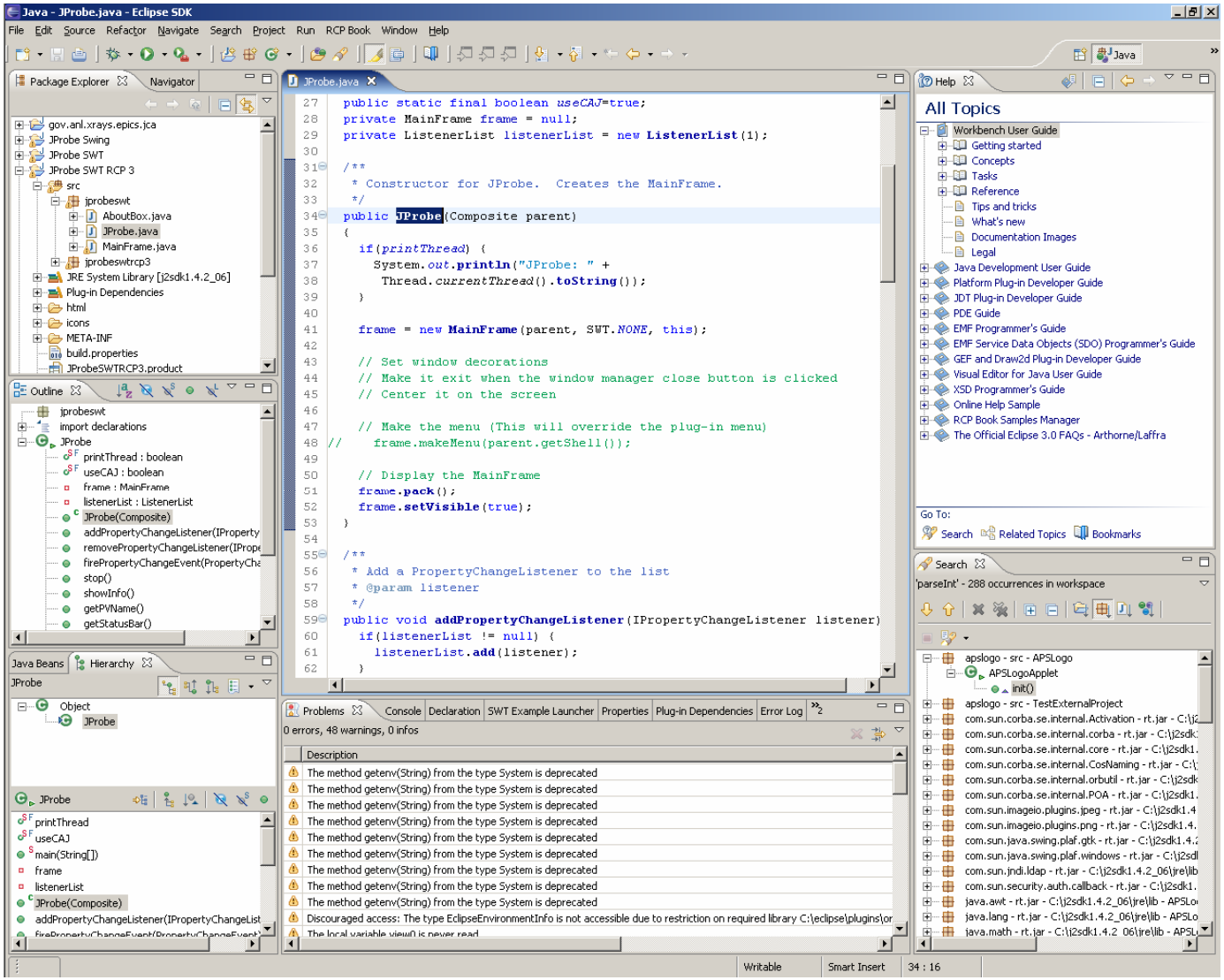
Modified From: Tony Lam, ICALEPCS Presentation, October 2005

Eclipse Layout Fundamentals

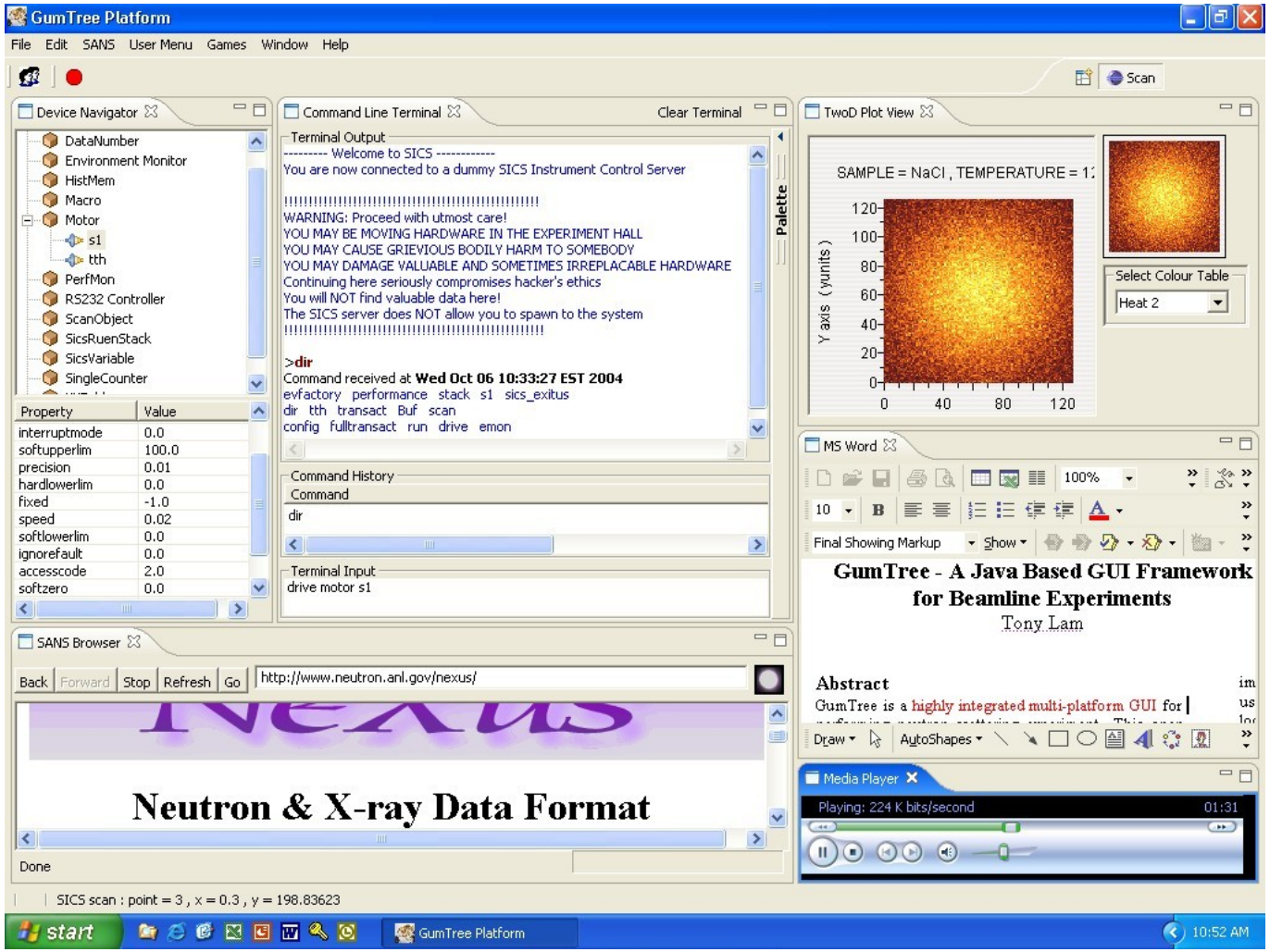
- Perspective: A particular layout of a Workbench window
  - Has zero or one Editor Area and zero or more surrounding Views



Eclipse as a Java IDE

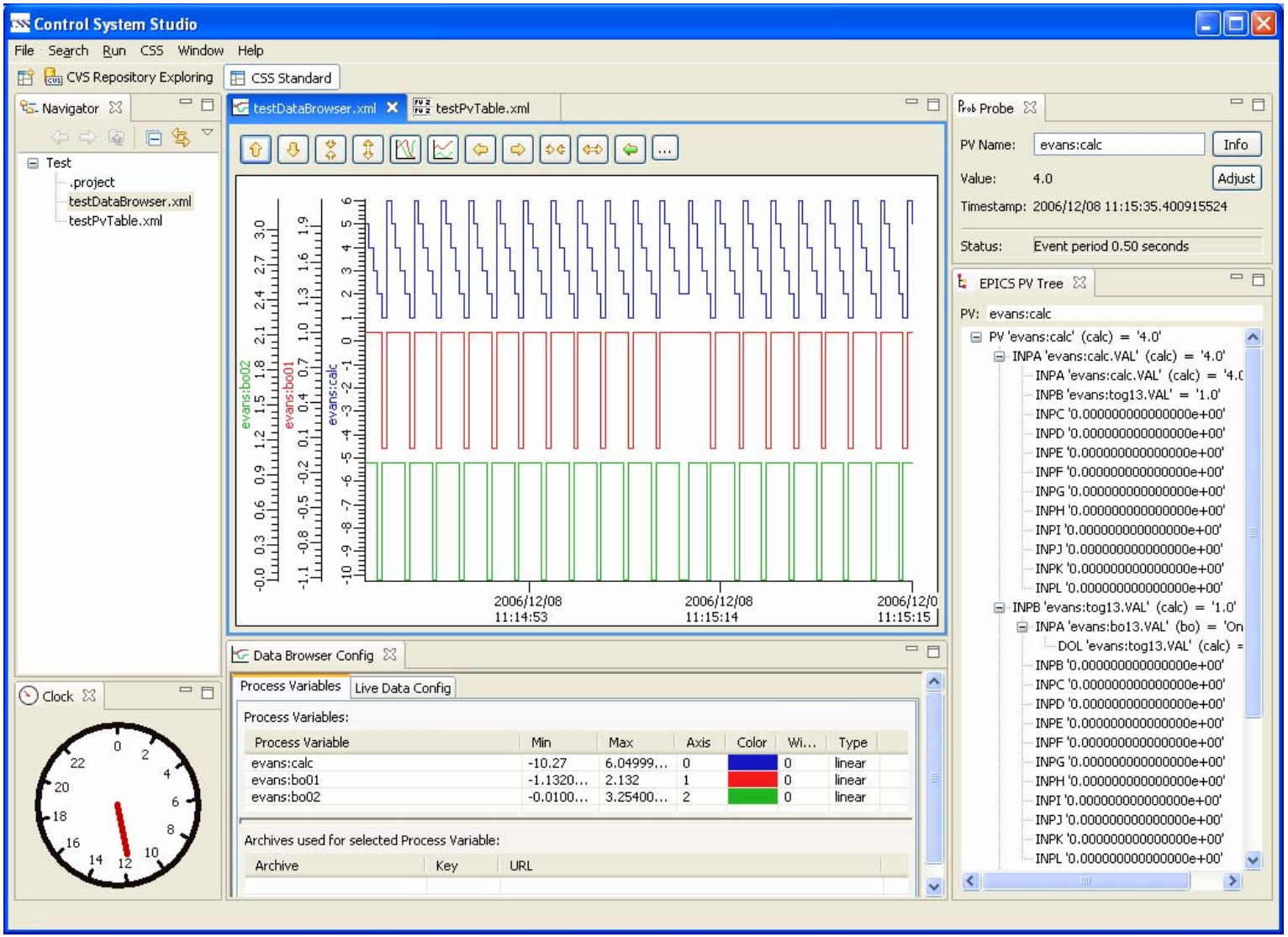


GumTree (ANSTO)

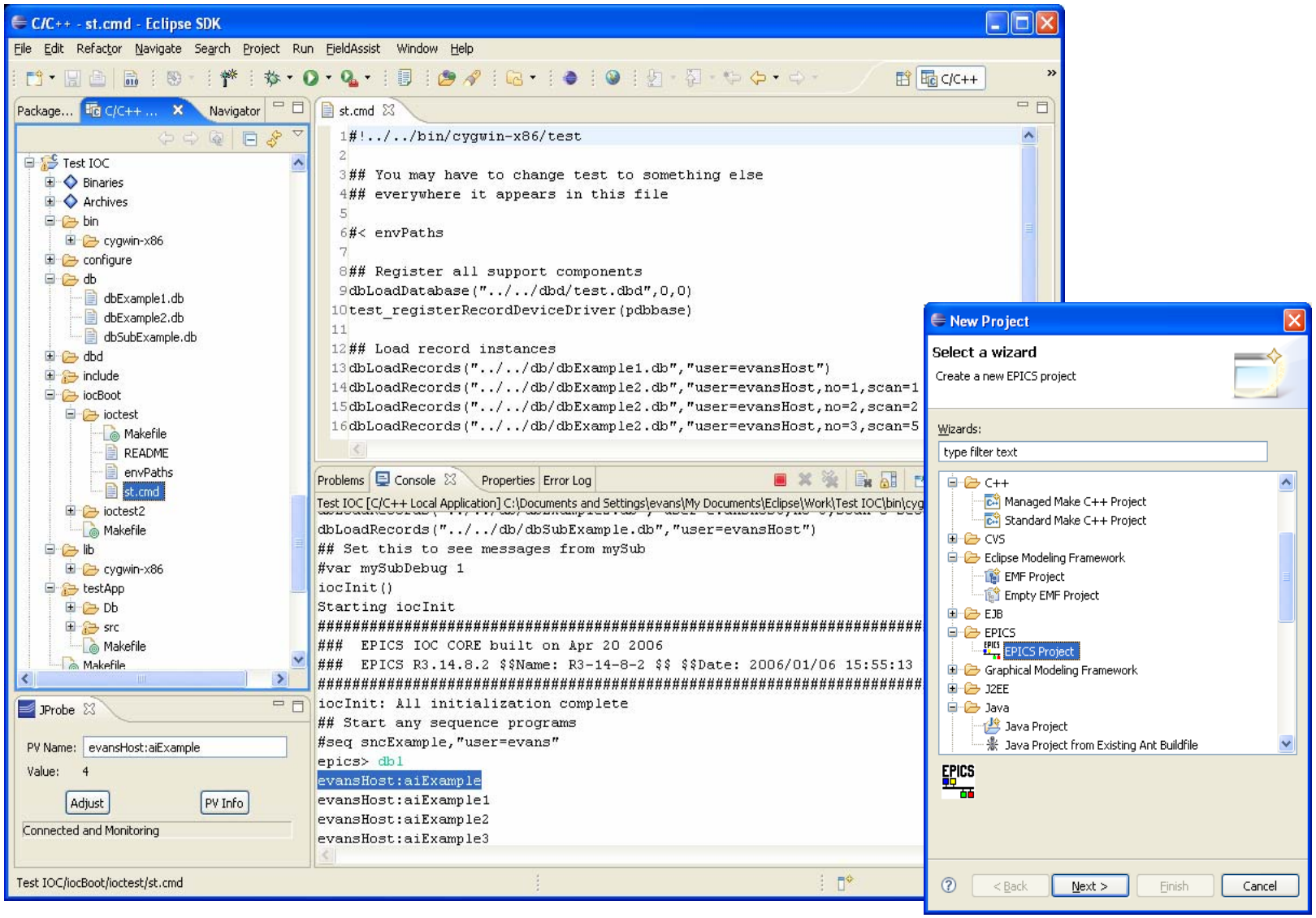


From: Tony Lam, ICALEPCS Presentation, October 2005

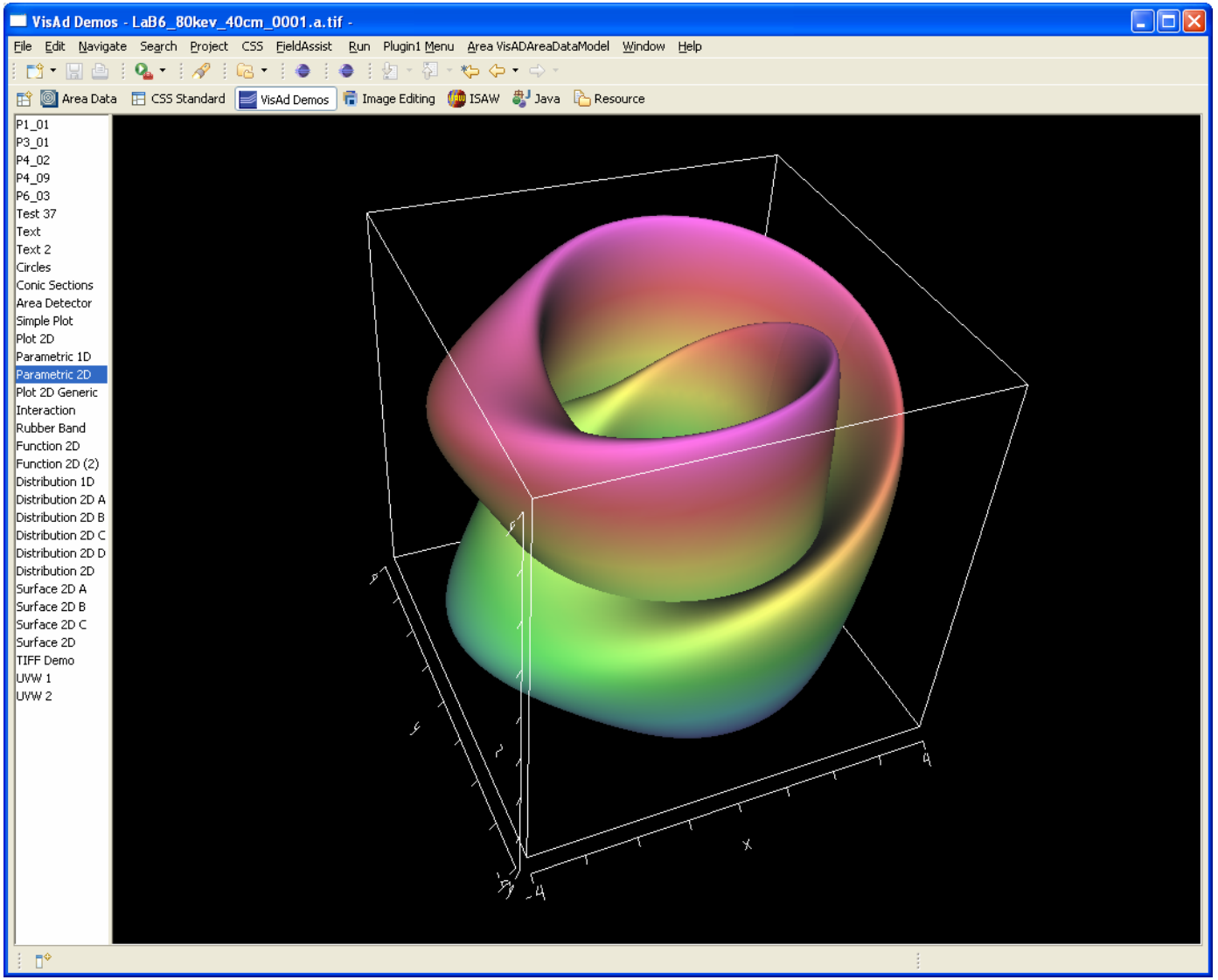
EPICS Control System Studio



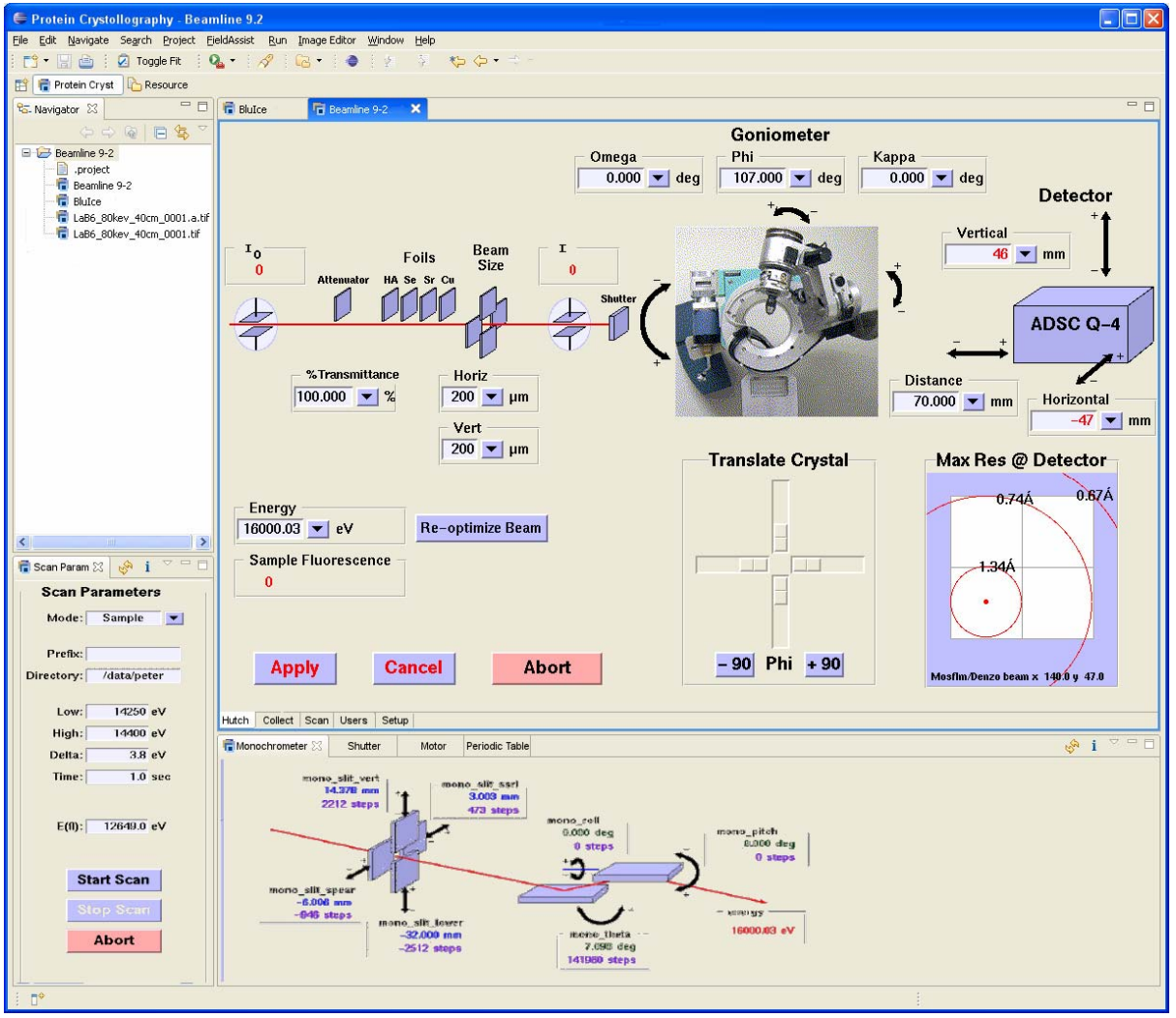
EPICS IDE : IOC Development



A Perspective Can be a Single Application



X-Ray Experiment



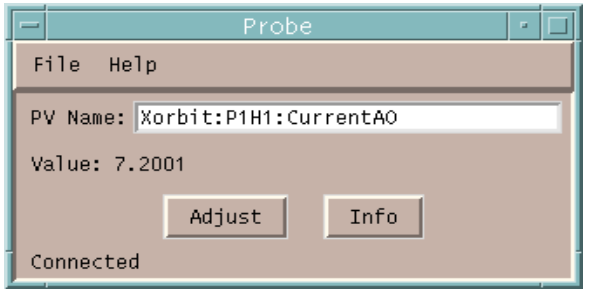
Images from: BLU-ICE and the Distributed Control System, NOBUGS III, January 2000

Rich Client Platform (RCP)

- "Rich Client" is a term from the early 1990s that distinguished applications built with Visual Basic and the like from "Console" or "Simple" applications
- Eclipse is particularly suited to Rich Client applications
- The possibility of using the Eclipse platform for applications was there from the beginning, but foreshadowed by its use as an IDE
  - In the early days it required hacking to make Rich Clients
- RCP is now (as of Eclipse 3.1) supported by the interface and encouraged
- You essentially use Eclipse as a framework for your application
  - You inherit all of its built-in features
  - As well as those from other community plug-ins
- You include only the plug-ins you need
- Is a very extensible development platform
  - You can use plug-ins developed by others as needed
  - Others can use yours and extend them

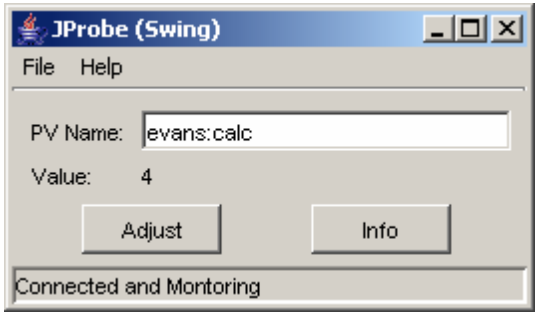
Eclipse as a Rich Client Platform

- Looks like an application, not an IDE
- Inherits a lot of functionality
  - Persistence (Properties and Preferences)
  - Help
  - Featured About dialog (like Eclipse's)
  - Splash screen
  - Dockable windows, and much more …

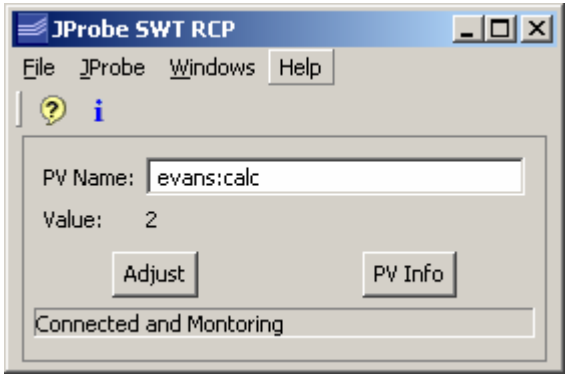


Pyre Application

probe.py --frame.pvName=Xorbit:P1H1:CurrentAO

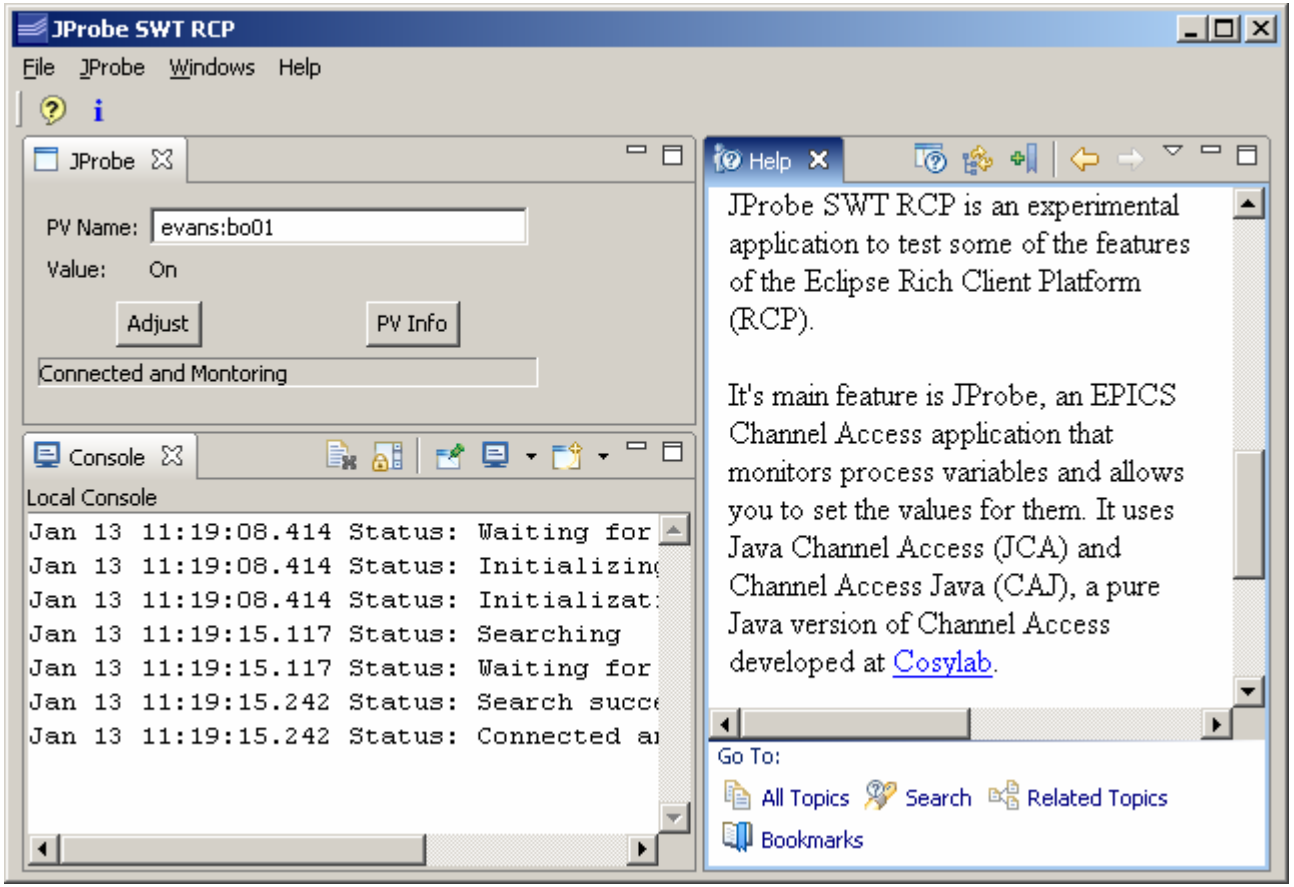


Java Application

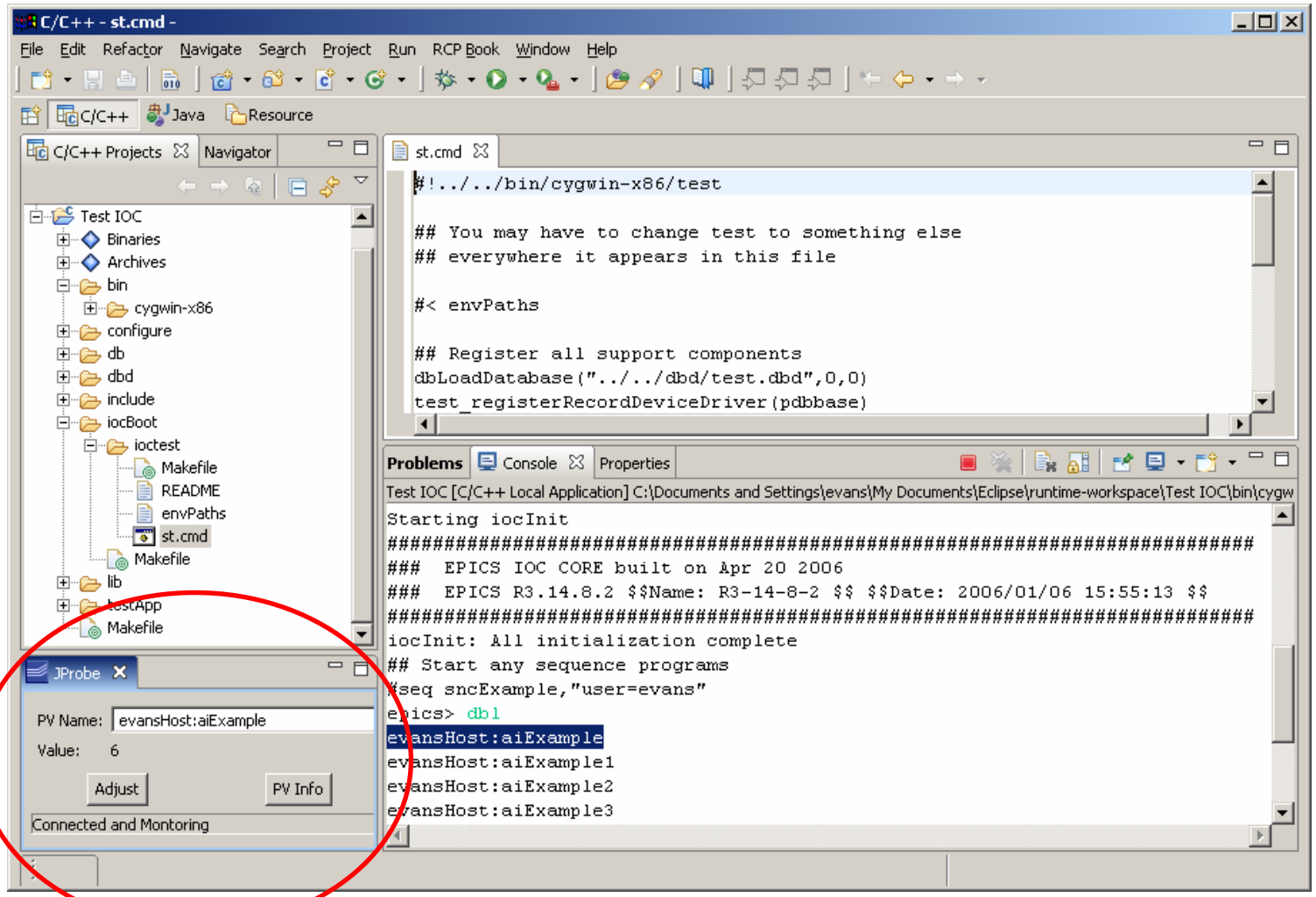


RCP Application

Probe on Steroids
Leveraging the Eclipse Framework



An RCP Application is also a Plug-In

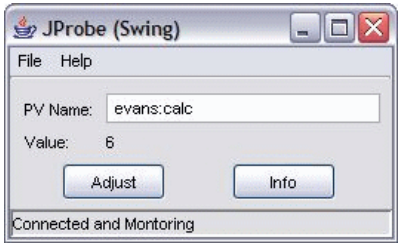


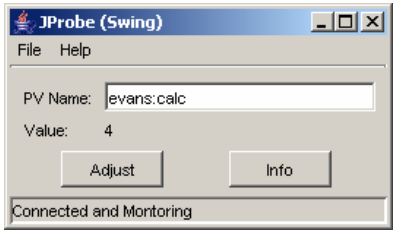
AWT vs. SWT - You Have to Decide

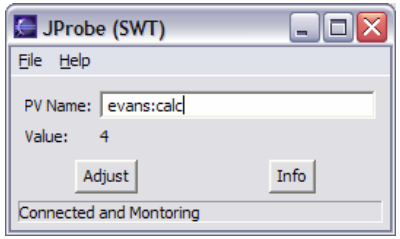
- AWT / Swing (Abstract Windowing Toolkit)
  - Write once, run anywhere
  - Formerly ugly, with bad performance
  - Now look and work well
  - Use garbage collection
  - Come with the JDK and JRE
- SWT / JFace (Standard Window Toolkit)
  - The important fact is that Eclipse uses SWT, not AWT
  - Supposed to look better, run faster
  - A thin wrapper around native widgets
  - SWT components must be disposed (vs. garbage collected)
    - *Owing to need to free native resources*
  - Need JNI libraries for each platform
  - Distribution is through the Eclipse Foundation, not Sun

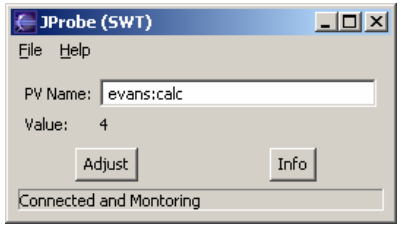
AWT vs. SWT - More Considerations

- It is not easy to convert between them
- The SWT look is not obviously better
- The performance difference may not be there either, today
- Eclipse uses SWT
  - They are supposed to mix and match, but ???
- Sun is unlikely to include SWT support in the JDK and JRE soon



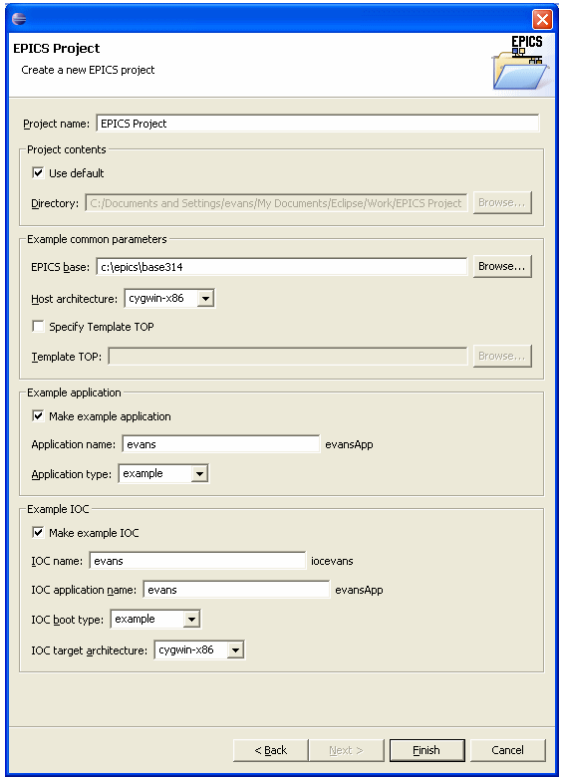


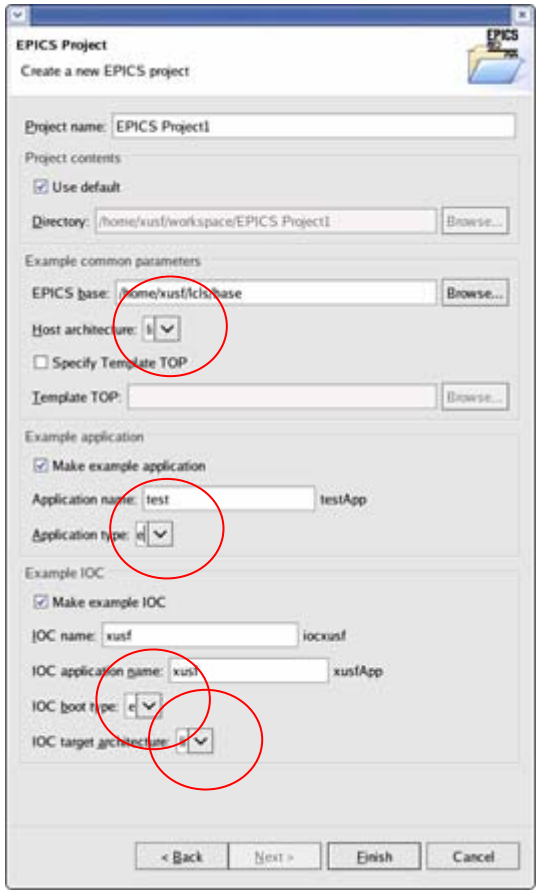




SWT Platform Dependence

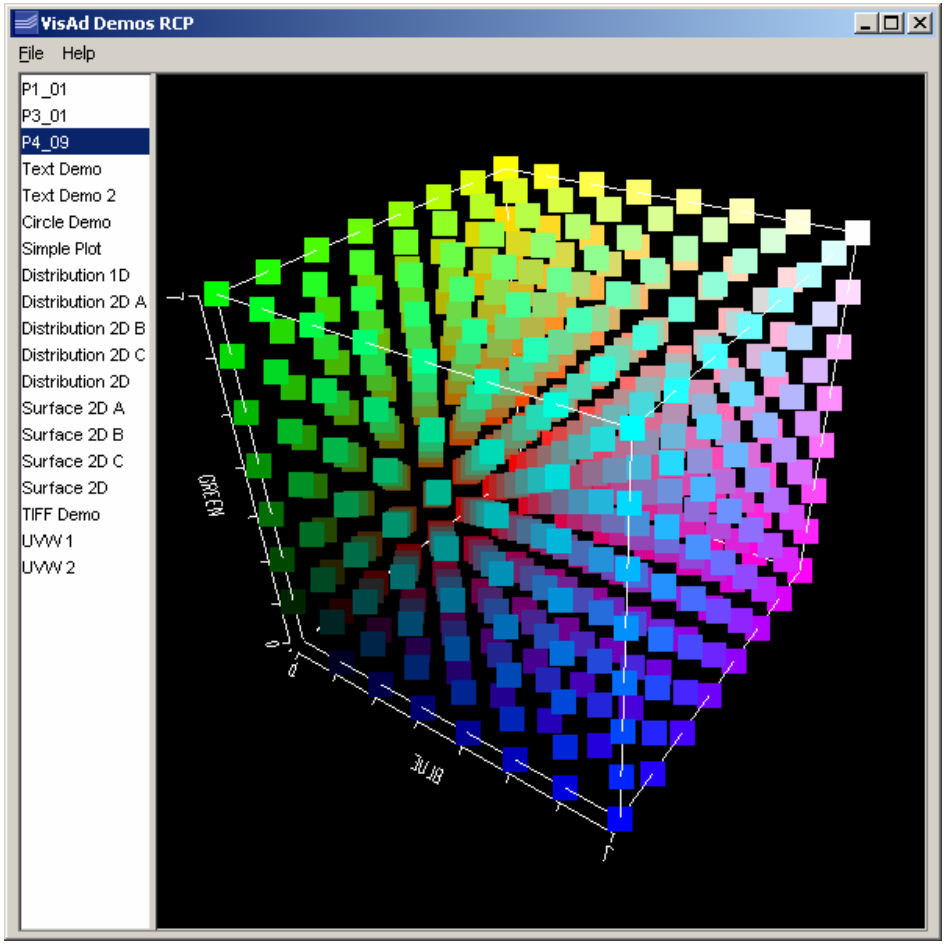
- Example: Working Windows dialog doesn't work right on Linux





Combining Swing and SWT - SWT_AWT Bridge

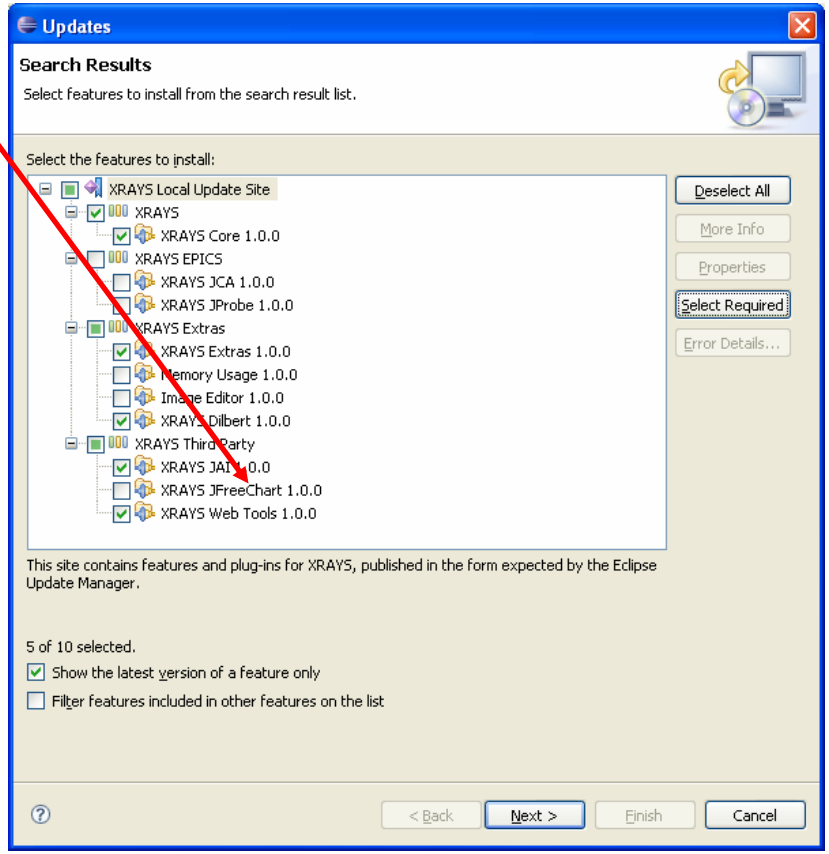
- ContentPane of JFrame is embedded in an SWT Composite
- Menu Initialization is separate from other UI initialization
  - Standalone Swing version uses Swing menus
  - RCP versions uses RCP workbench menus
  - Both can call same instance methods (or not)
- This application also uses JAI and J3D
  - Both are Java extensions
  - Don't play well with Eclipse



Deployment is a Major Reason for Using Eclipse

- Both Java and Eclipse are multi-platform
- Updates are easily made through the Eclipse update mechanism
- You can wrap 3rd party applications in your own plug-ins
  - For example:

    The Feature “XRAYS JFreeChart” contains gov.anl.xrays.jfreechart which wraps JFreeChart
  - Including DLLs and Shared Objects
- Guarantees they are versions that work with your applications on all supported platforms
- Makes it easy for the user to install and update both your stuff and the 3rd party stuff



Eclipse Bottom Line

- Is a very powerful and extensible IDE and Framework
- Is also an IE - A way to organize your work
- Is Open Source
- Has a community
- Is supported by most of the industry
- Has a large number of developers (>150)
- Has significant financial backing
- Are many 3rd-party Plug-ins, both free and commercial
- Are more than 60 open-source projects
  - From Web Tools to Code Profilers
- Is continuing to expand and improve rapidly
- Is free
- Downsides
  - Is a continually changing, moving target

Lessons Learned

- Eclipse as a workbench seems more attractive than Eclipse RCP applications
- Writing full-fledged Eclipse plug-ins entails a fairly steep learning curve
- The SWT / AWT dichotomy is a nuisance but not a real impediment
- The best (perhaps only feasible) way to handle 3rd party libraries is to wrap them in a plug-in
- Eclipse solves a number of thorny problems
  - Especially deployment and commonality
- We don’t see a better alternative

X-Ray Software Development at the APS

- Best described as “Uncoordinated”
- Wide variety of languages
  - FORTRAN, C, C++, Perl, Tcl/Tk, Python, Java, …
- Visualization relies on (different) commercial products
  - IDL, IGOR, Matlab, …
- Each beamline tends to do its own thing
- Modeling and Analysis is not well integrated with Data Acquisition
- Lack of real-time data reduction
- Little high-performance computing
- Little remote access
- No common data format

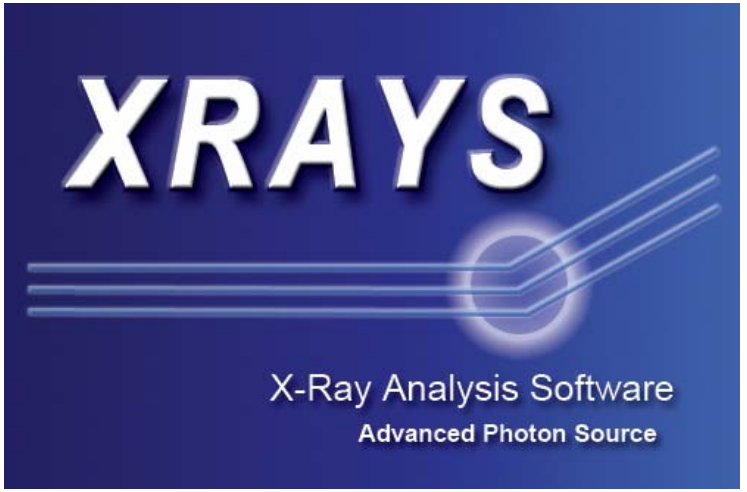
- A Scientific Software Section was formed to help remedy this situation

Scientific Software Section

- Specific goals:
  - Combine existing analysis and visualization codes with beamline data acquisition software and transform these codes into easy-to-use software
  - Provide a scientific workbench program that is easy to use and learn and from which users can access all the software that is necessary to manage the entire scientific work flow
  - Create new analysis and visualization applications that can be used on all beamlines and that are easily integrated into the standard workbench
  - Develop a software framework, perhaps more than one, that provides tested and debugged scientific routines, such as fitting and visualization, which can be used by developers to create applications
  - Create an interface to the facilities necessary to provide high-performance computing
  - Provide documentation, distribution, maintenance, and support

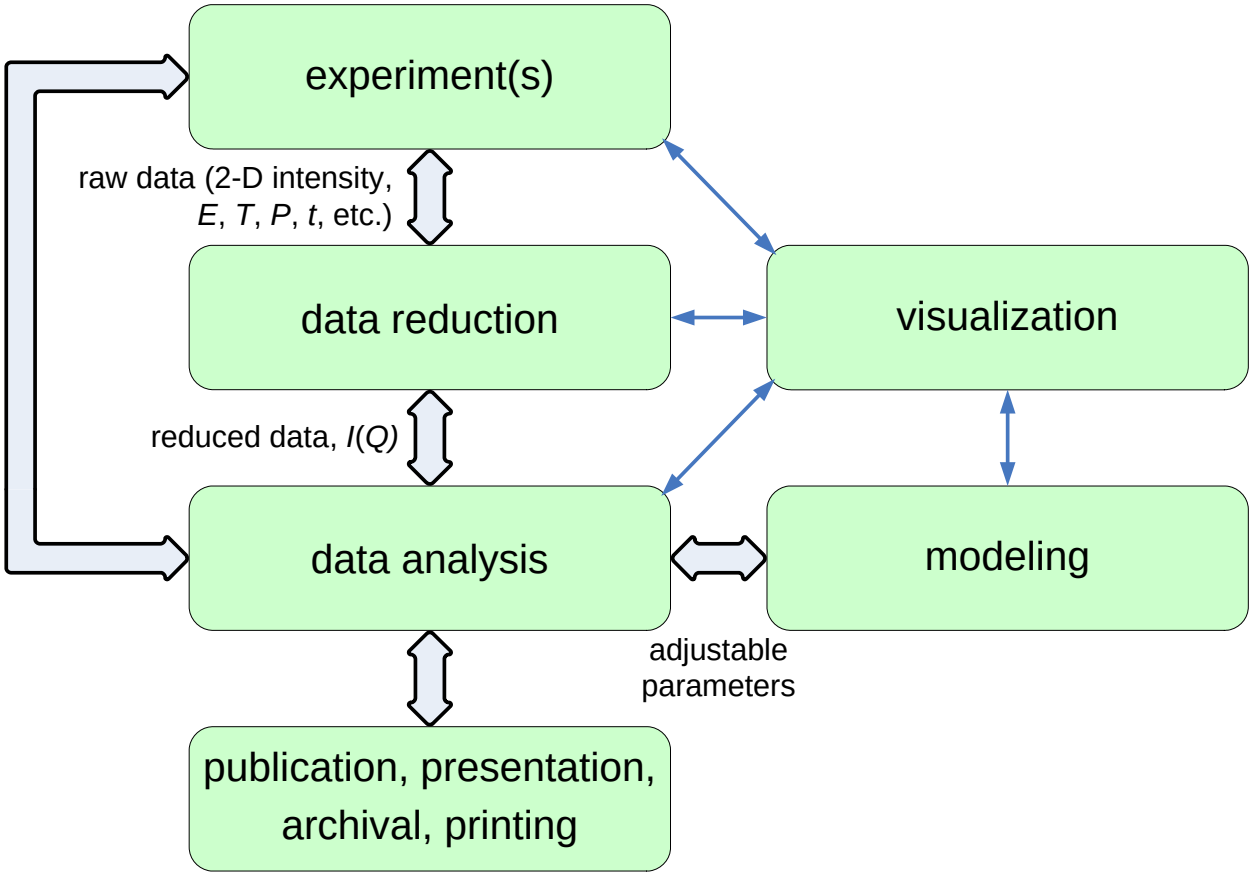
Scientific Software Section Web Page

XRAYS



- Stands for X-Ray Analysis Software
  - (or X-Ray Software)
- It is expected to grow into a large suite of analysis and visualization applications
- These will include:
  - Scientific workbench program
  - New analysis and visualization applications
  - Updating and coordination of existing analysis and visualization applications
  - A framework of software routines that developers can use to write applications
- It currently consists mostly of exploration and prototype applications
  - This is the groundwork for what we really want to do
  - More than 1200 Java source files in 60 projects
  - 38 Java projects intended for distribution (gov.anl.xrays.xxx)
  - 10 ready-to-deploy features (collections of projects) in 4 categories

We Want to Manage the Entire Experimental Data Flow



XRAYS Rationalization for Eclipse

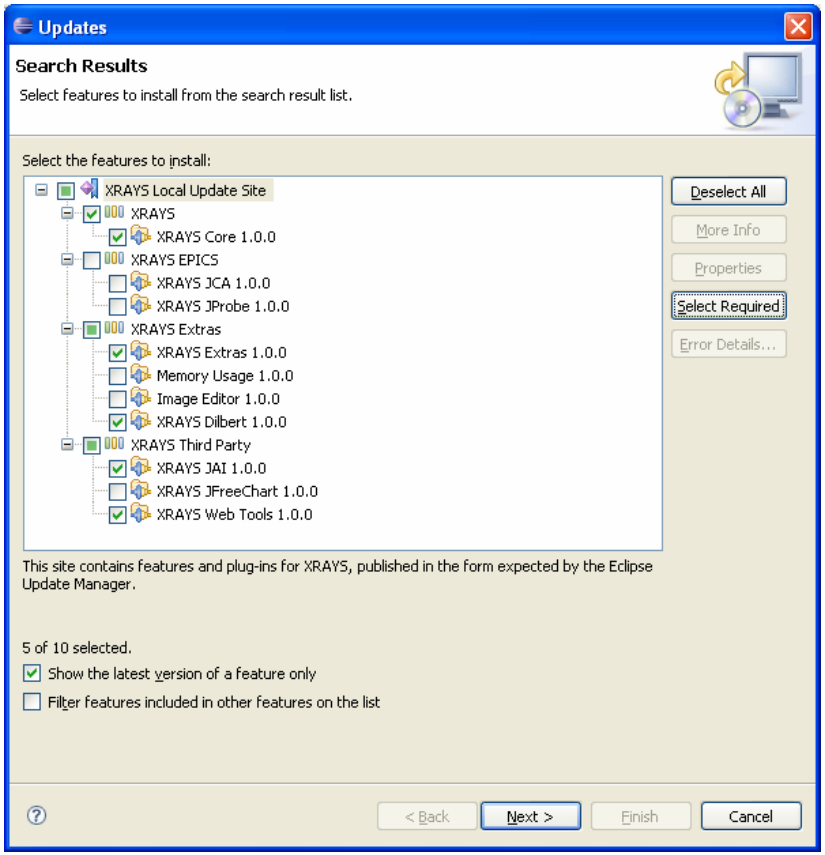
- Providing coordination is a primary goal
- Resources are limited
- Have to choose something
  - Eclipse seems like the best choice
  - Powerful, flexible, extensible
  - Open-source
  - Huge community with many projects
- Java development environment leads to high productivity
- Deployment via plug-ins appears to solve many problems

- We intend to use Eclipse, not as an IDE, but as a workbench
  - Something users will use

- Downsides
  - Most x-ray beamline staff and users are not using Eclipse now
  - 95% will be unhappy [with anything we do]

Ready to Deploy Now

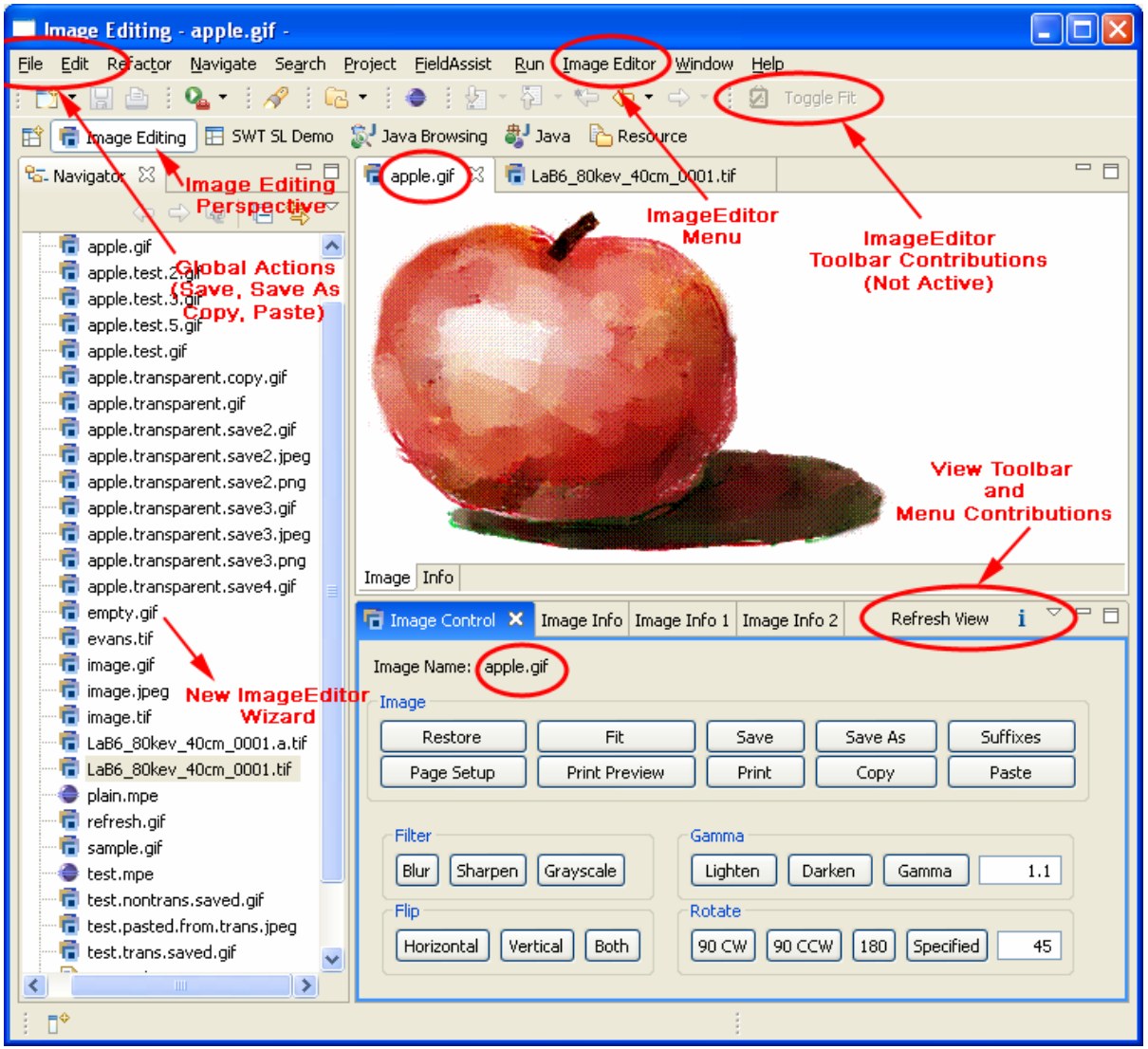
- These Eclipse projects are ready now to be made available through the Eclipse update mechanism
- Waiting for license and license-related issues



Eclipse for Users, not Developers

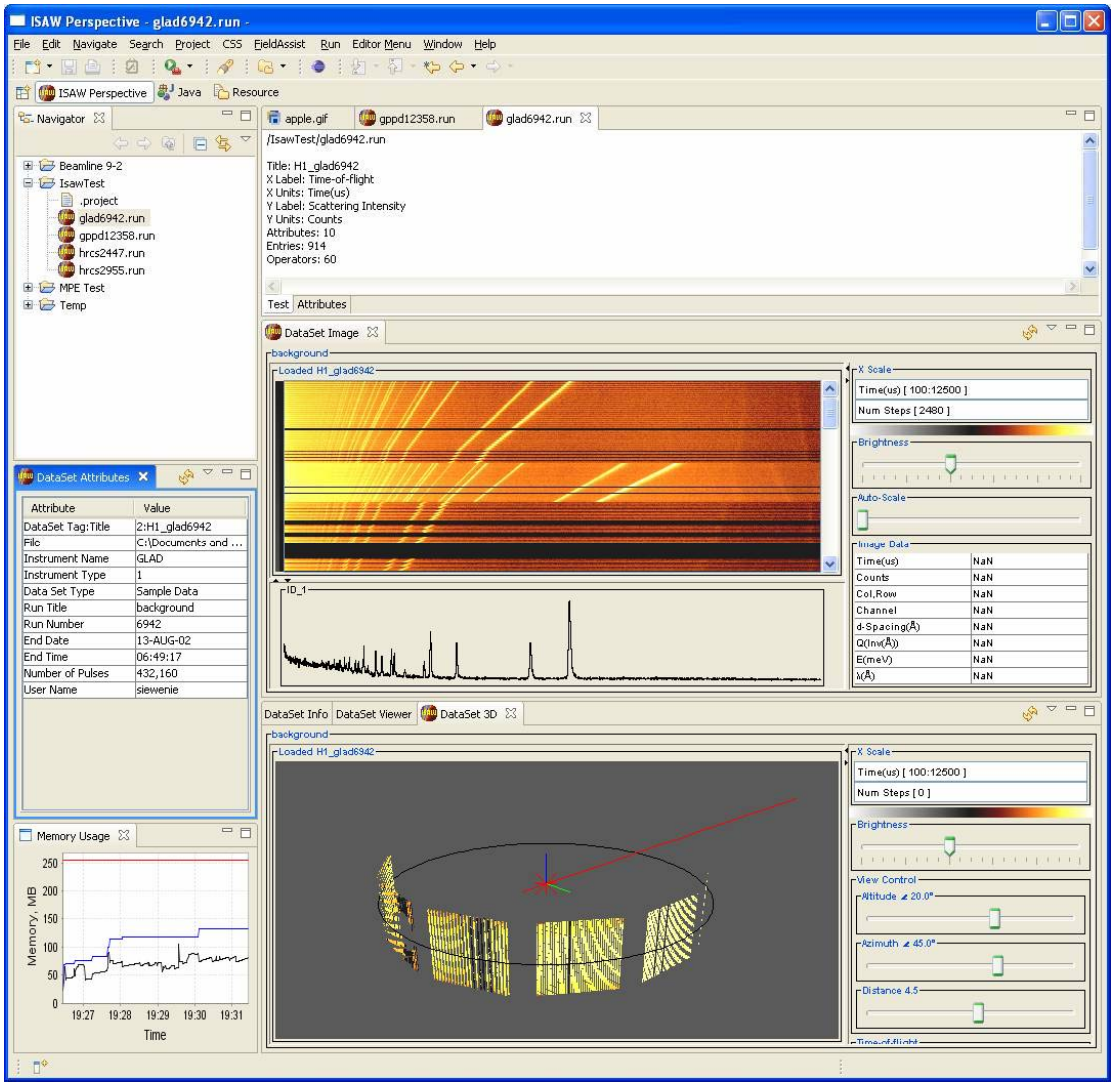
- We intend to use Eclipse as a workbench
- Something a user can come in and be up and running with in a short time
  - Probably with community help
- Each user can use and customize it in his or her own way
  - (That is what Eclipse provides)
- They will probably use it for more than one thing
  - That is why the layout by Perspective is important
  - You just switch perspectives to change tasks
- I think this paradigm is better than using RCP applications
  - You provide the plug-ins
  - The user manages the Workbench as he or she pleases

Image Editor as an Experiment Prototype

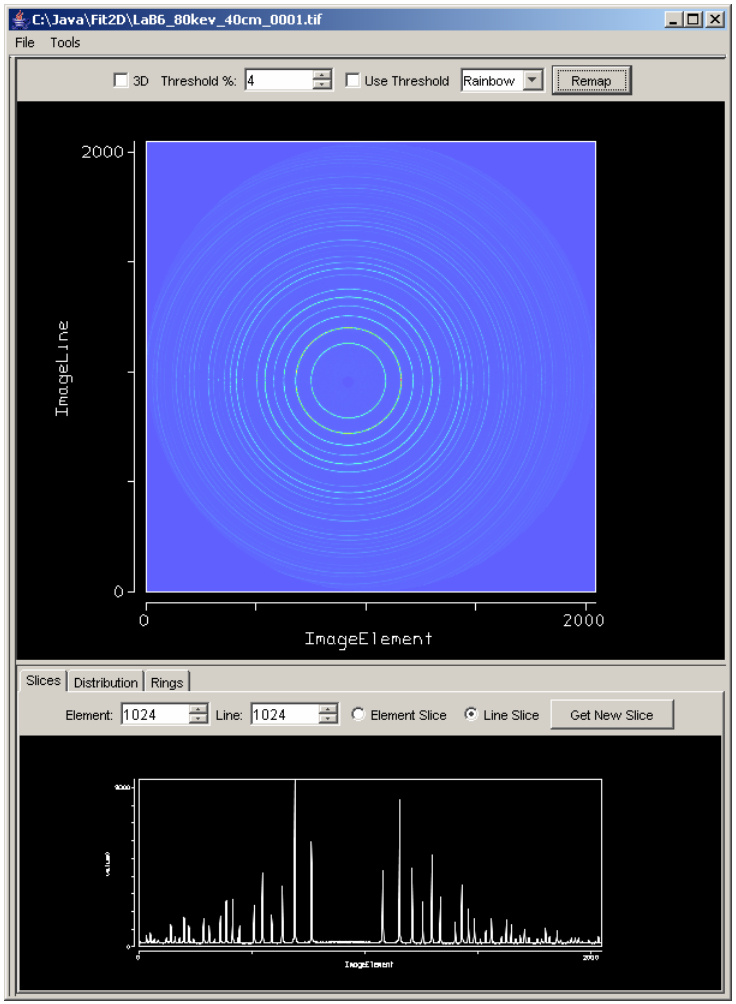


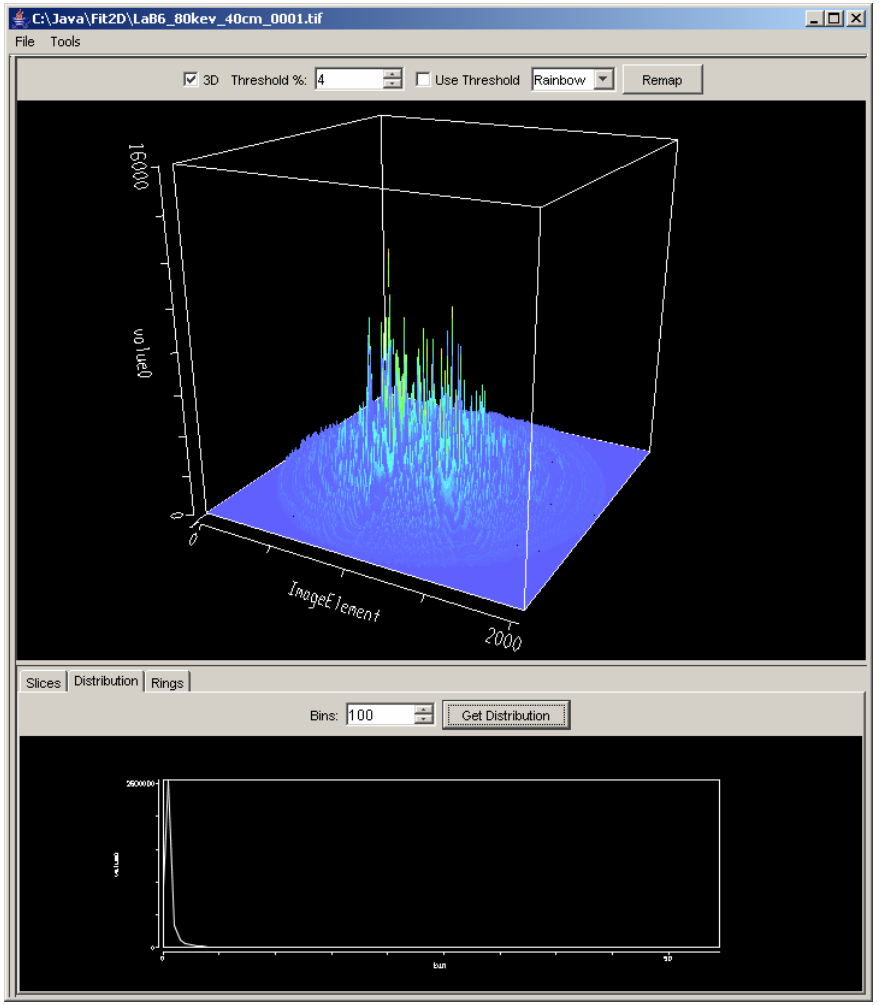
Prototype Implementation of ISAW

- Includes:
  - A Perspective
  - An Editor for ISAW DataSets
    - *.run, .isd*
  - Some Views
- All work together
  - Views change when the edited file changes

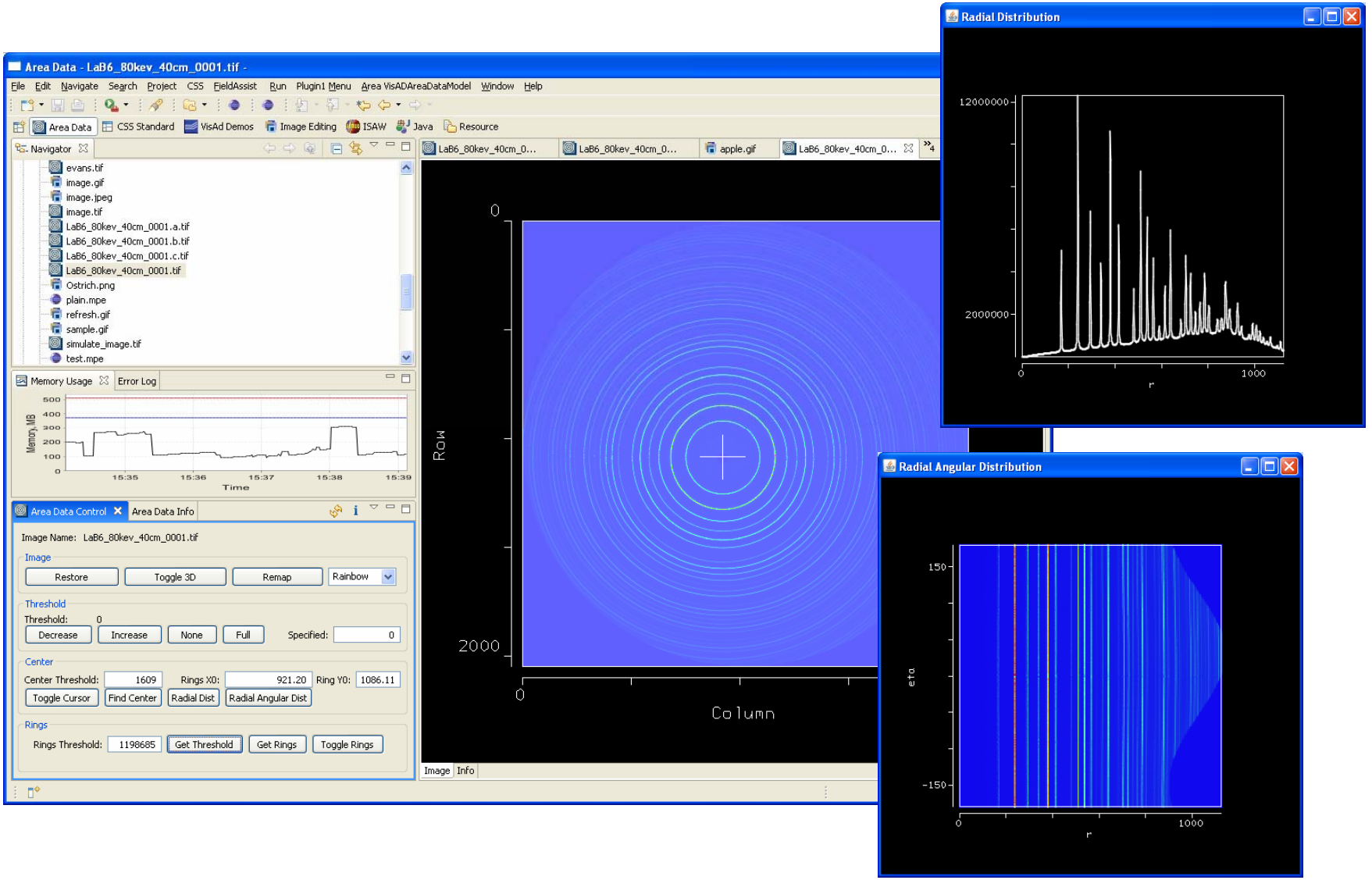


Prototype Image Analysis Tool using VisAD Graphics





Now Incorporated Into Eclipse



Longer Term

- Resources are currently limited
- Eclipse and Java applications are where we are starting
  - These are client-based applications
  - Partly driven by the fact that x-ray data is localized
- We expect to incorporate high-performance computing
  - Typically means clusters and grid computing
  - These are server based
  - Data on centralized servers is more typical of other communities
  - Eclipse is not the obvious tool
- We cannot limit ourselves to Java
  - There are legacy FORTRAN codes that need to be incorporated
  - There are many other languages
    - *In particular, Python is heavily used in scientific communities*
    - *C and C++ will continue to be important*
- The licensing and other legal structure need to support all of these

Thank You

This has been a

Scientific Software Presentation

Thank You

This has been a

Scientific Software Presentation