



... for a brighter future

The Scientific Software Initiative at the APS

Kenneth Evans, Jr.

Presented at the EPICS Collaboration Meeting

October 13 - 14, 2007

ICALEPCS Meeting, Knoxville, TN



U.S. Department
of Energy



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

Current X-Ray Scientific Software Situation

- 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
 - Provide high-performance computing
 - Provide documentation, distribution, maintenance, and support

Why Java ?

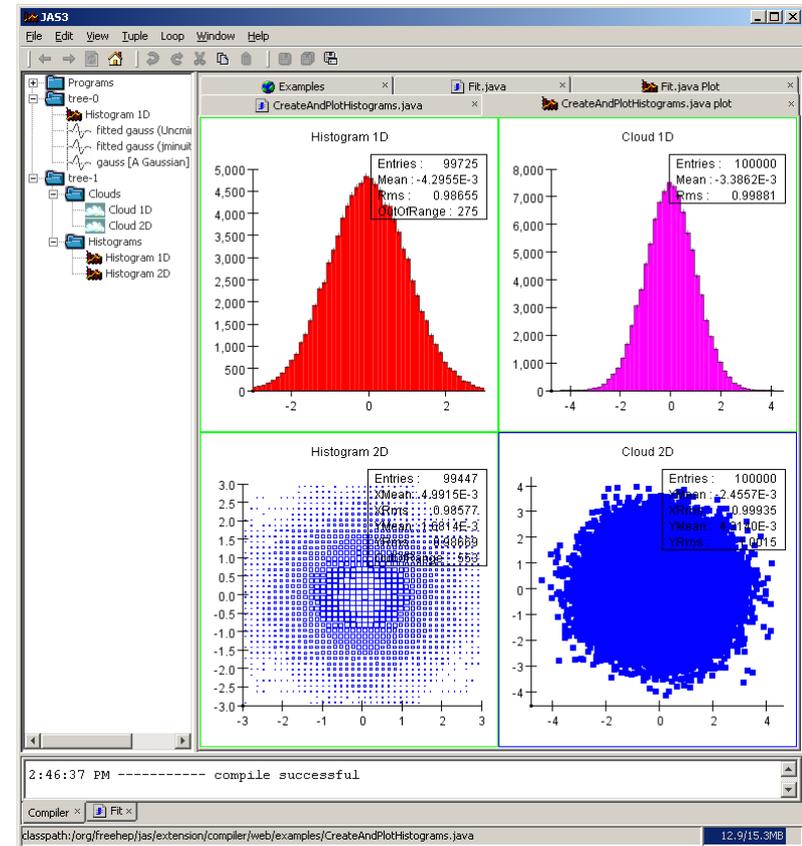
- Java has become a major and important language
- The reason is that most commercial development uses J2EE
 - It is running the financial world
 - There is money to be made improving Java and its tools
- Applications have performance approaching applications written in C
- There is already extensive scientific development in Java
- Advantages
 - 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
 - Eliminates most typos and all syntax errors
- Provides content assist
 - It tells you what to type next and provides documentation, etc.
- Compiles as you write
 - No “write – compile – load – run – figure out what happened” cycle
 - 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
- The above are just a small sample
 - Some of these are available for other languages
 - But generally not at the level they are for Java

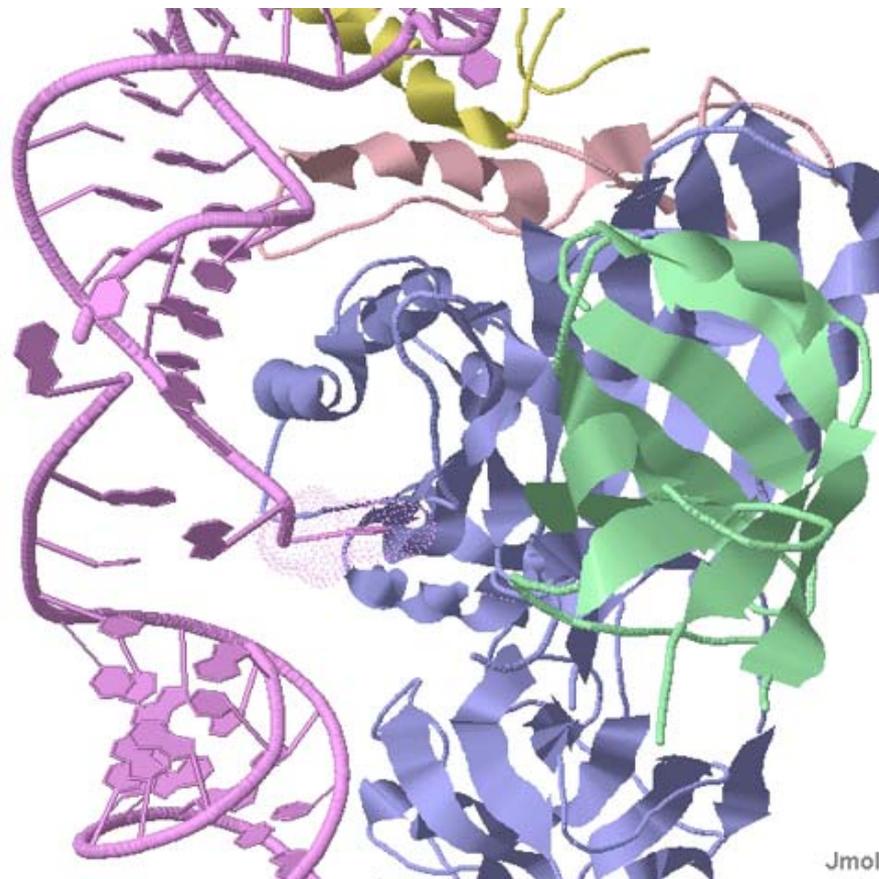
Java Analysis Studio (JAS3)

- 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

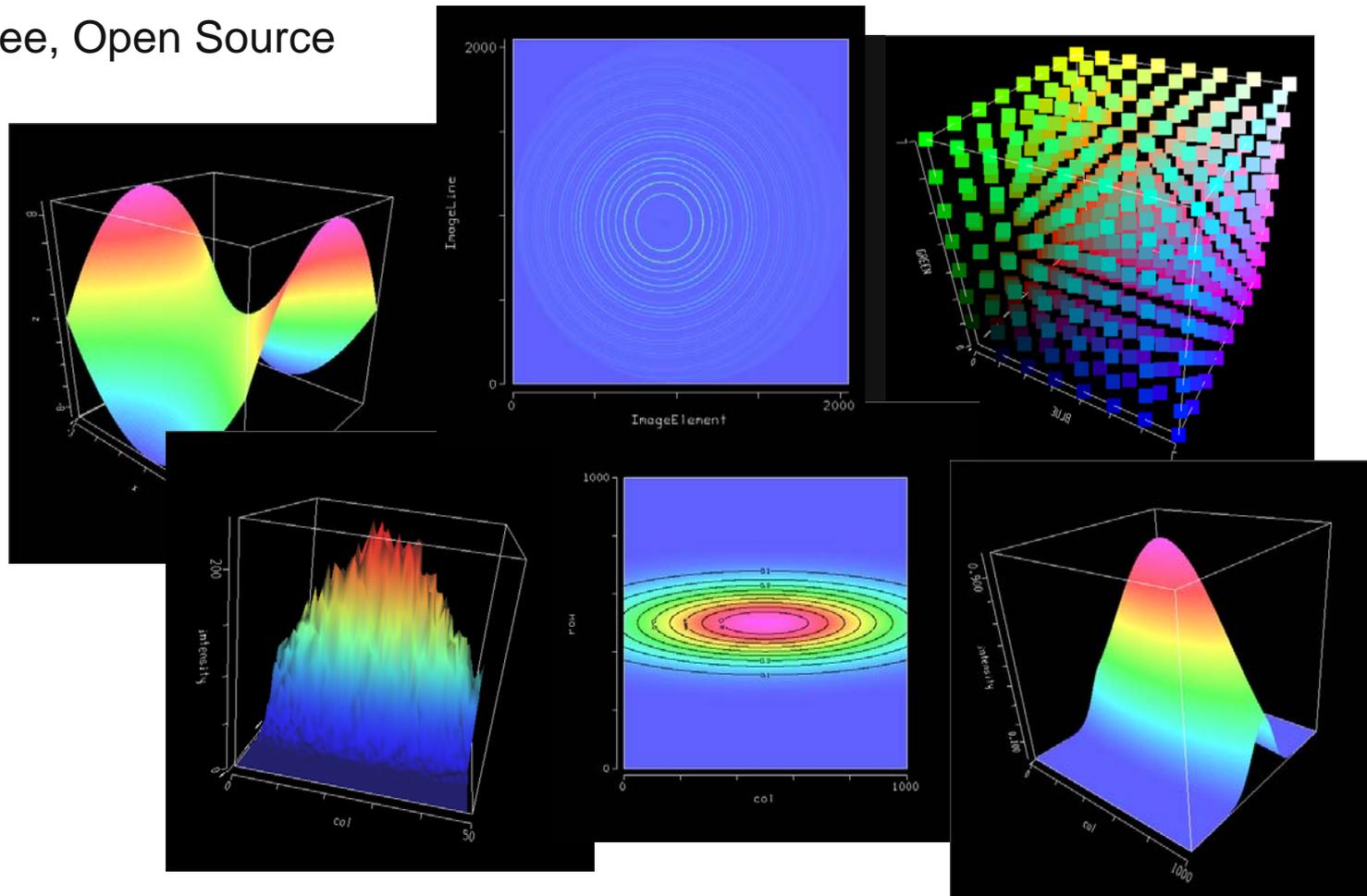
- Commonly used as an applet that can be integrated into web pages to display molecules in a variety of ways
- Also has a standalone application and a development tool kit that can be integrated into other Java applications
- Interactive, 3D
- Free, Open Source
- One of several Java Molecular Graphics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



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

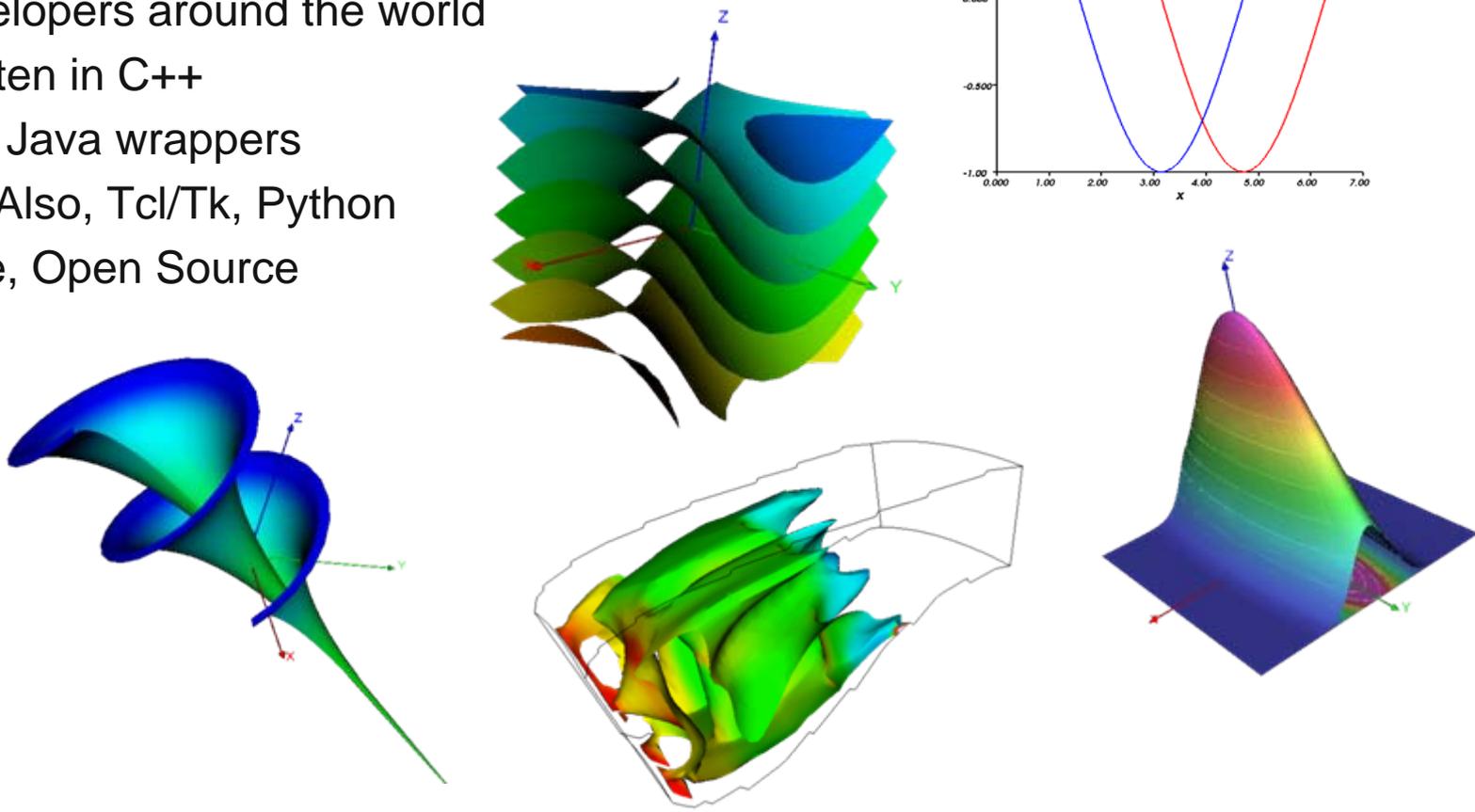
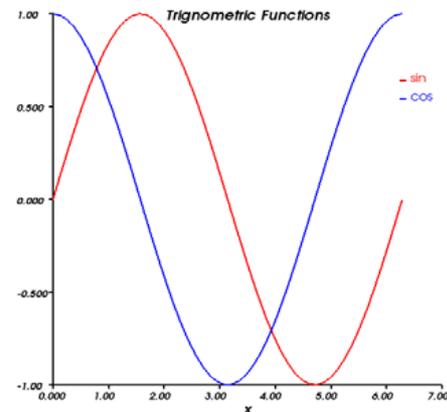
VisAD

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

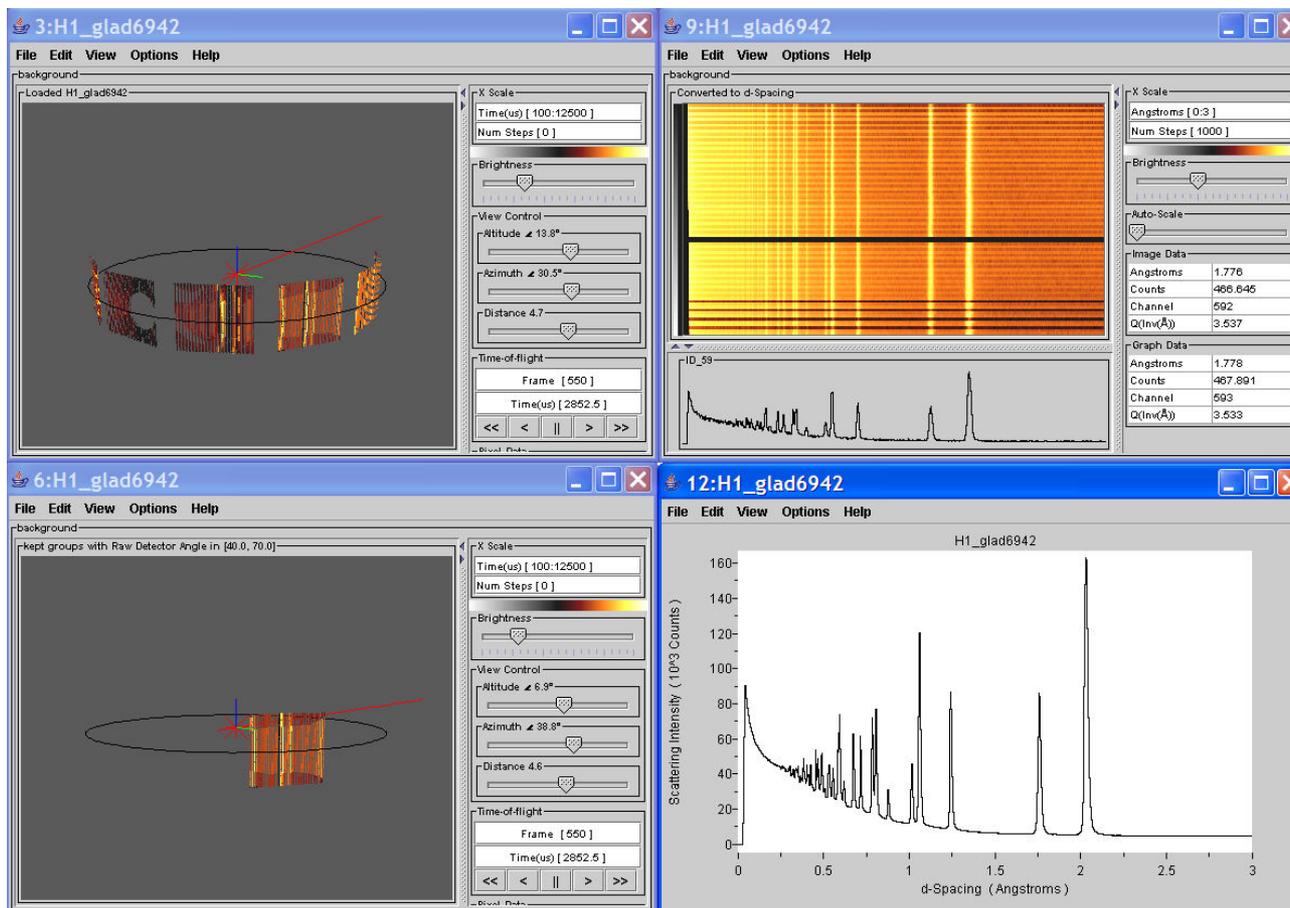


VTK

- 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



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



From: John Hammonds, IPNS

Bioclipse

The screenshot displays the Bioclipse software interface. The main window shows a 3D ball-and-stick model of a molecule. The interface is divided into several panels:

- BioResource Navigator:** A tree view on the left showing a file hierarchy with folders like 'SampleData', 'jmolScripts', 'js', 'pdb', 'sequences', 'spectra', and individual files like 'ATP.mol', 'CD0020.mol', 'C05441.mol', 'polycarpol.mol', 'reserpine.mol', 'showSecondaryProteinStructure', 'testedfeeds.opml', 'thiamin.mol', and 'viagra.xyz'.
- thiamin.mol / viagra.xyz:** A central panel displaying a table of 63 atoms with their coordinates (x, y, z) and a fourth column of values (likely charges or weights).
- Jmol View:** A large window on the right showing the 3D ball-and-stick model of the molecule.
- ChemTree / Console:** A panel at the bottom center showing a hierarchical tree structure of the molecule's components, including 'ChemSequence', 'ChemModel', 'SetOfMolecules', and 'Molecule' with a list of atom types (Atom N, Atom C, Atom S, Atom O).
- Properties:** A panel at the bottom right showing a table of properties and their values.

Atom	x	y	z	Value
63 N	-3.4932	-1.895	0.1795	0.0
C	-4.9343	-1.8728	0.3721	0.0
C	-5.6426	-2.7318	-0.6885	0.0
N	-5.3188	-2.2766	-2.0316	0.0
C	-3.8765	-2.2967	-2.2211	0.0
C	-3.1722	-1.4332	-1.1621	0.0
C	-5.9989	-3.0732	-3.0409	0.0
S	-2.6047	-1.0544	1.432	0.0
O	-3.0459	-1.5296	2.7325	0.0
O	-2.7462	0.3736	1.21	0.0
C	-0.8416	-1.5089	1.2833	0.0
C	-0.4767	-2.819	0.9782	0.0
C	0.8715	-3.1614	0.8754	0.0
C	1.9131	-2.2421	1.0805	0.0

Property	Value
CDK	
Atom count	63
Bond count	0
File format	XYZ
Formula	C22H30N6O4S
Mass	474.20456
Natural mass	474.58374
SMILES	N.C.C.N.C.C.C.S.O.O.C.C.C.C.C.O.C.C...
Strand count	
General	
Format	XYZ
Name	viagra.xyz
Parent	N/A
Path	C:\Documents and Settings\evans\My Docu...
Registered type	not bioclipse model resource FileResource

Wolfram Workbench (Mathematica)

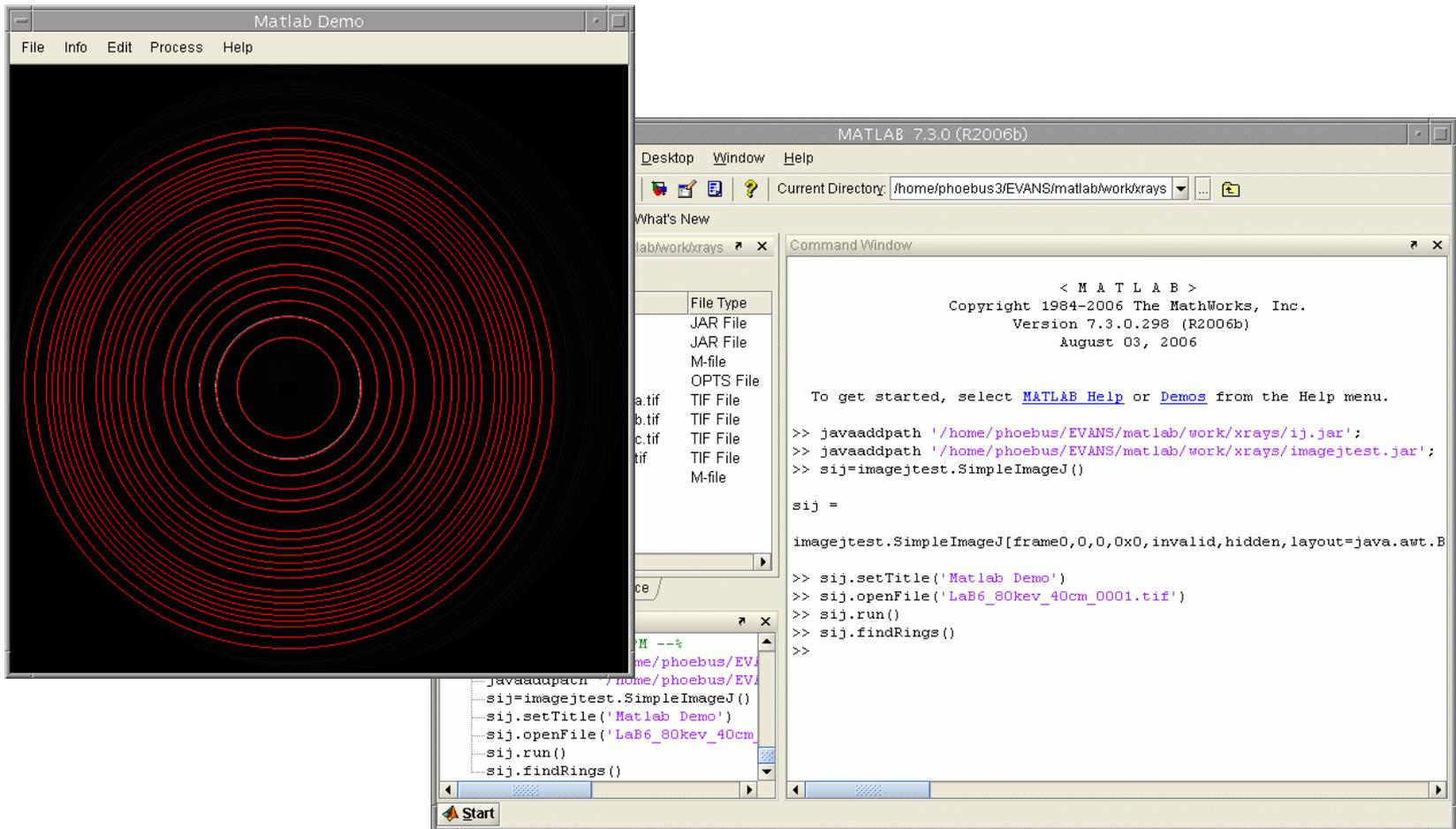
The screenshot displays the Wolfram Workbench (Mathematica) interface. The main window is titled "Mathematica Development - XraysTest.m - Wolfram Workbench". The interface is divided into several panes:

- Package Explorer:** Shows a hierarchical tree of the project structure. The "XraysTest" package is selected, showing its sub-packages: "javaSource", "JRE System Library [1.1]", "JLINK_JAR - C:\Program Files\Wolfram R...", "XraysTest", "Java", "Kernel", "SimpleImageJ.m", "XraysTest.m", "FileCheck.nb", "javaTest.nb", "javaTestModified.nb", "JavaWindows.nb", "SimpleImageJTest.nb", and "XraysTest.nb".
- Code Editor:** Displays the source code for "XraysTest.m". The code is as follows:

```
1 (* Mathematica Package *)
2
3 (* Created by the Wolfram Workbench Sep 7, 2007 *)
4
5 BeginPackage["XraysTest`", {"JLink`"}]
6
7 RunGC::usage = "RunGC[] runs the Java garbage collector.";
8
9 SysInfo::usage = "SysInfo[] returns system information.";
10
11 CheckFile::usage = "CheckFile[] checks various information about fileName.";
12
13 Begin["`Private`"]
14
15 RunGC[] :=
16   Module[ {},
17     InstallJava[];
18     LoadJavaClass["java.lang.Runtime"];
19     java`lang`Runtime`gc[];
20   ]
21
22 SysInfo[] :=
```
- Tasks/Problems/Console/Outline/Search/Properties:** Shows the "Outline" pane, which lists the public and private symbols defined in the package. Under "Public expressions", it lists "RunGC::usage", "SysInfo::usage", and "CheckFile::usage". Under "`Private`", it lists "RunGC[]", "SysInfo[]", and "CheckFile[fileName_String]".

Java in Matlab (or IDL or Mathematica or...)

- Matlab has extensive support for Java
 - A scientific software framework can also be used in these apps



Why Eclipse?

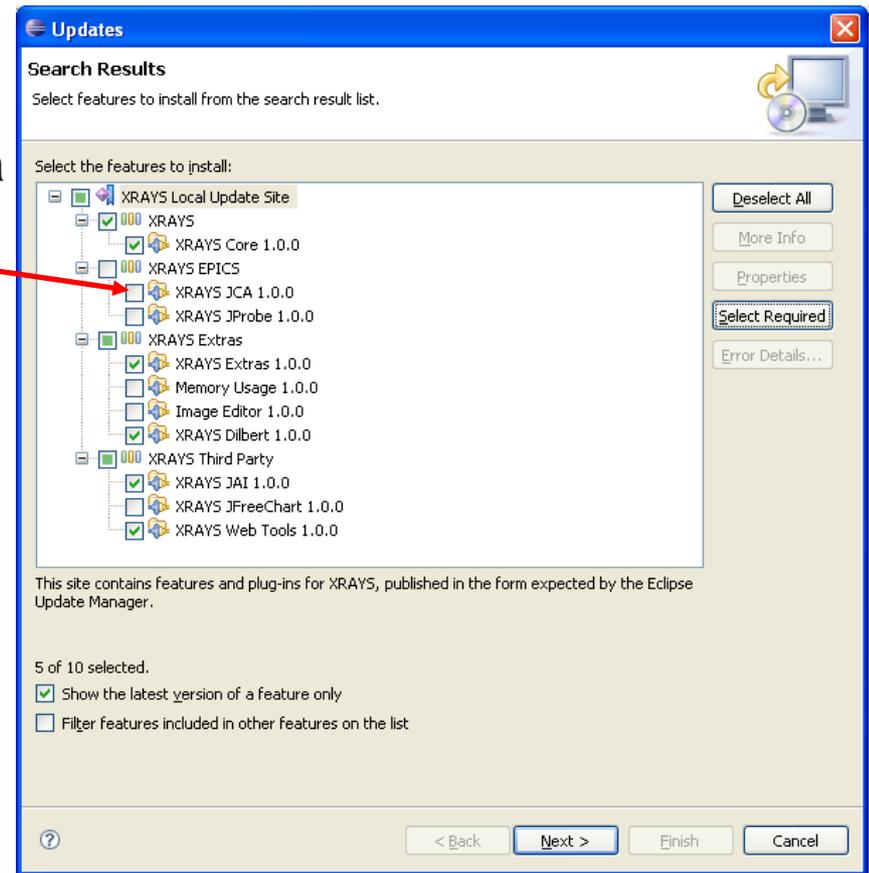
- Powerful, flexible, extensible
- Open-source
- Has a huge community with many projects
- Java development environment leads to high productivity
- Deployment via plug-ins appears to solve many problems

- We intend to use Eclipse, not as an IDE, but as a workbench (IE)
 - Something users will use
 - A way to organize their work

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

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 JCA” contains org.scatteringsw.xrays.jca which wraps JCA
 - Including DLLs and Shared Objects
- Guarantees they are versions that work with your applications on all platforms you support
- Makes it easy for the user to install and update both your stuff and the 3rd party stuff



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
- 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 (such as CSS)
 - Especially for our less-controlled environment
 - You provide the plug-ins
 - The user manages his Workbench as he or she pleases

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
 - Currently more than 3000 Java source files in 120 projects
 - 38 Java projects intended for distribution (org.scatteringsw.xrays.xxx)
 - 10 ready-to-deploy features (collections of plug-ins) in 4 categories

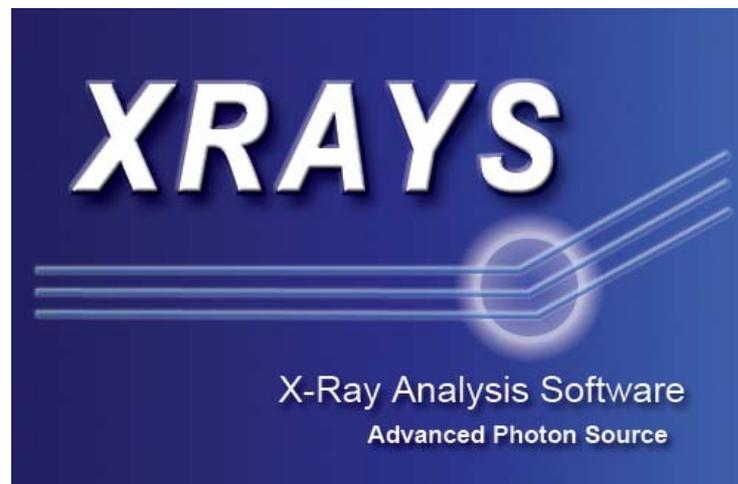
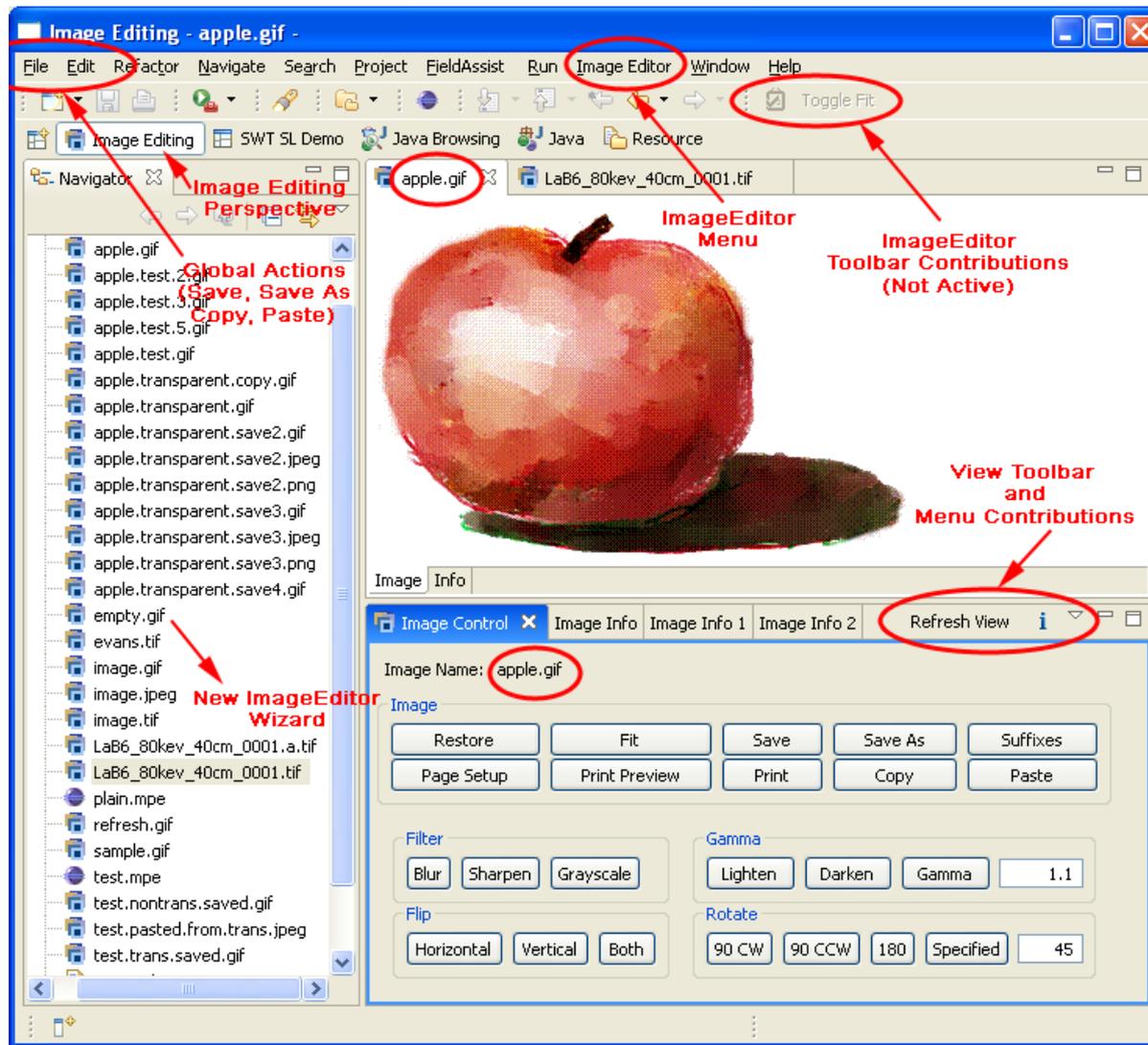
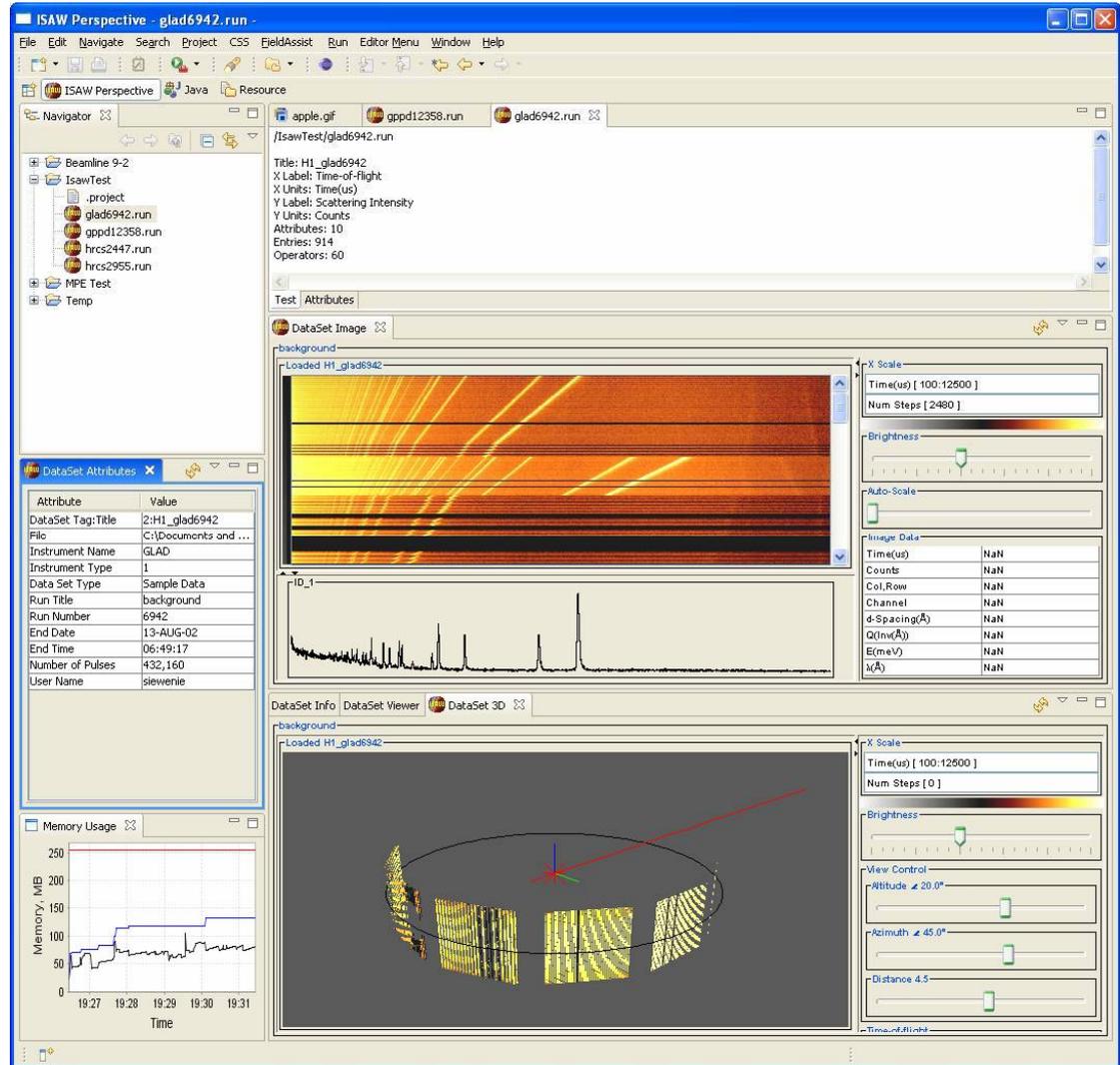


Image Editor as an Experiment Prototype

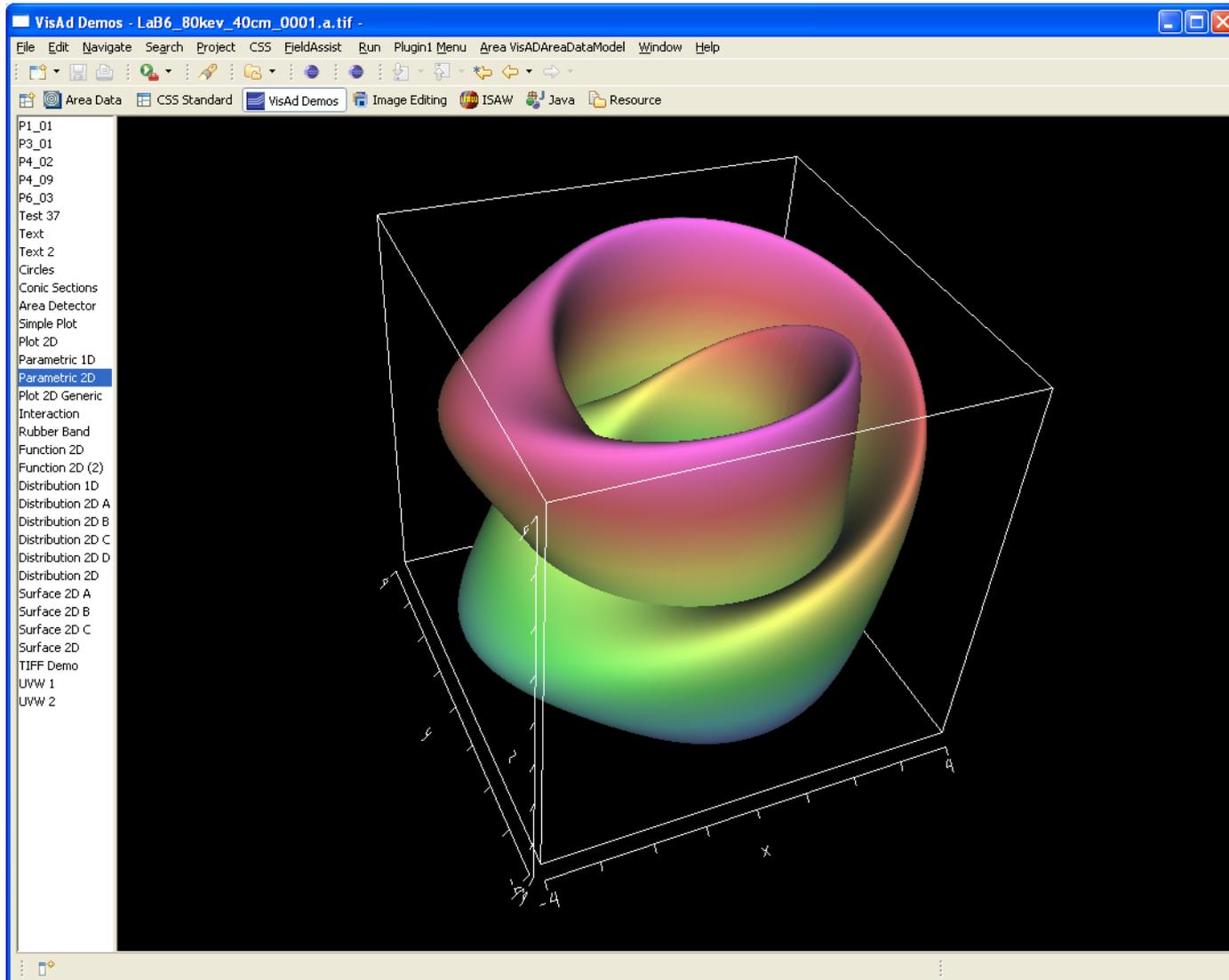


Prototype Implementation of ISAW

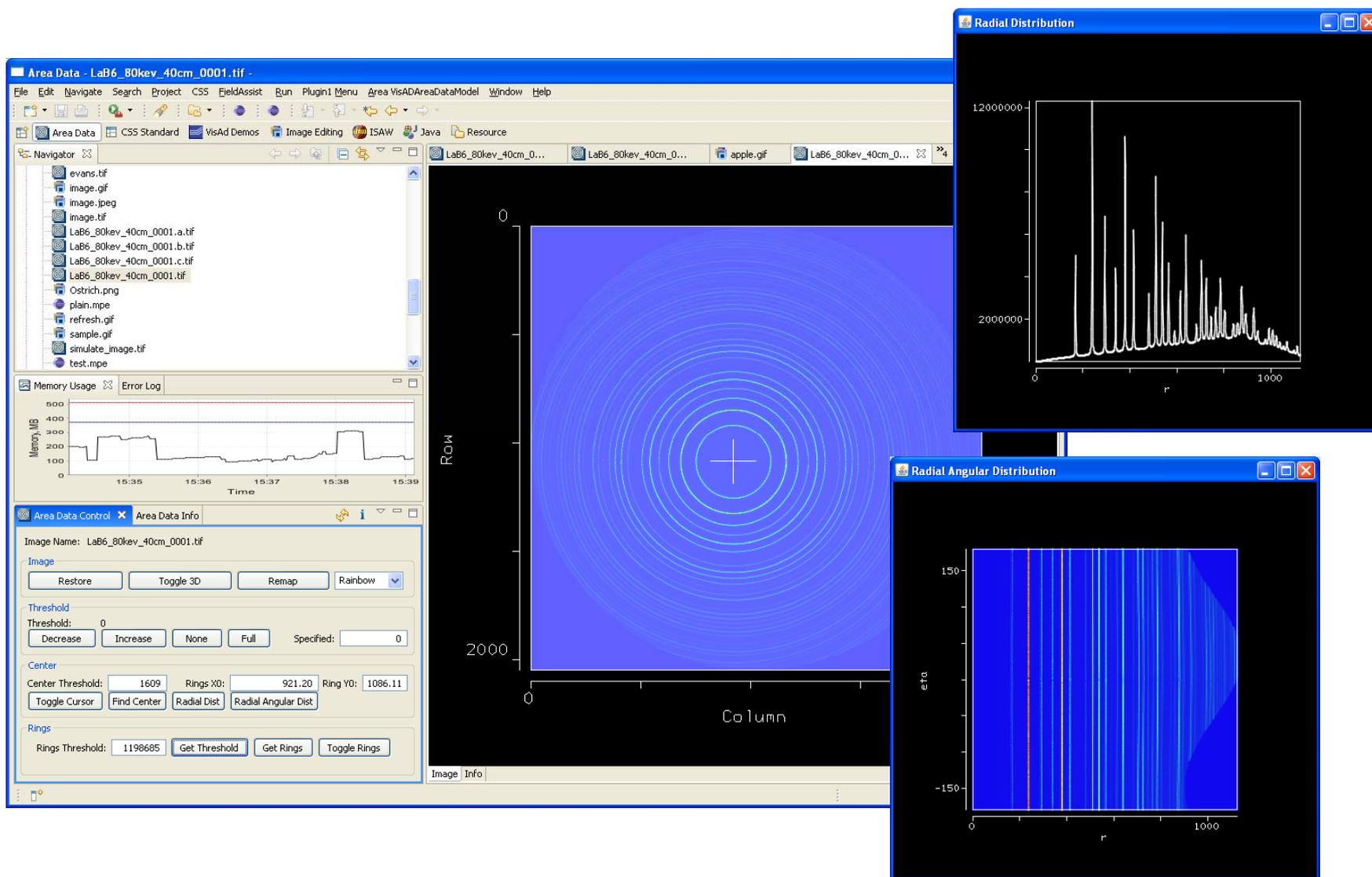
- Includes:
 - A Perspective
 - An Editor for ISAW DataSets
 - *.run*, *.isd*
 - Some Views
- All work together
 - Views change when the edited file changes



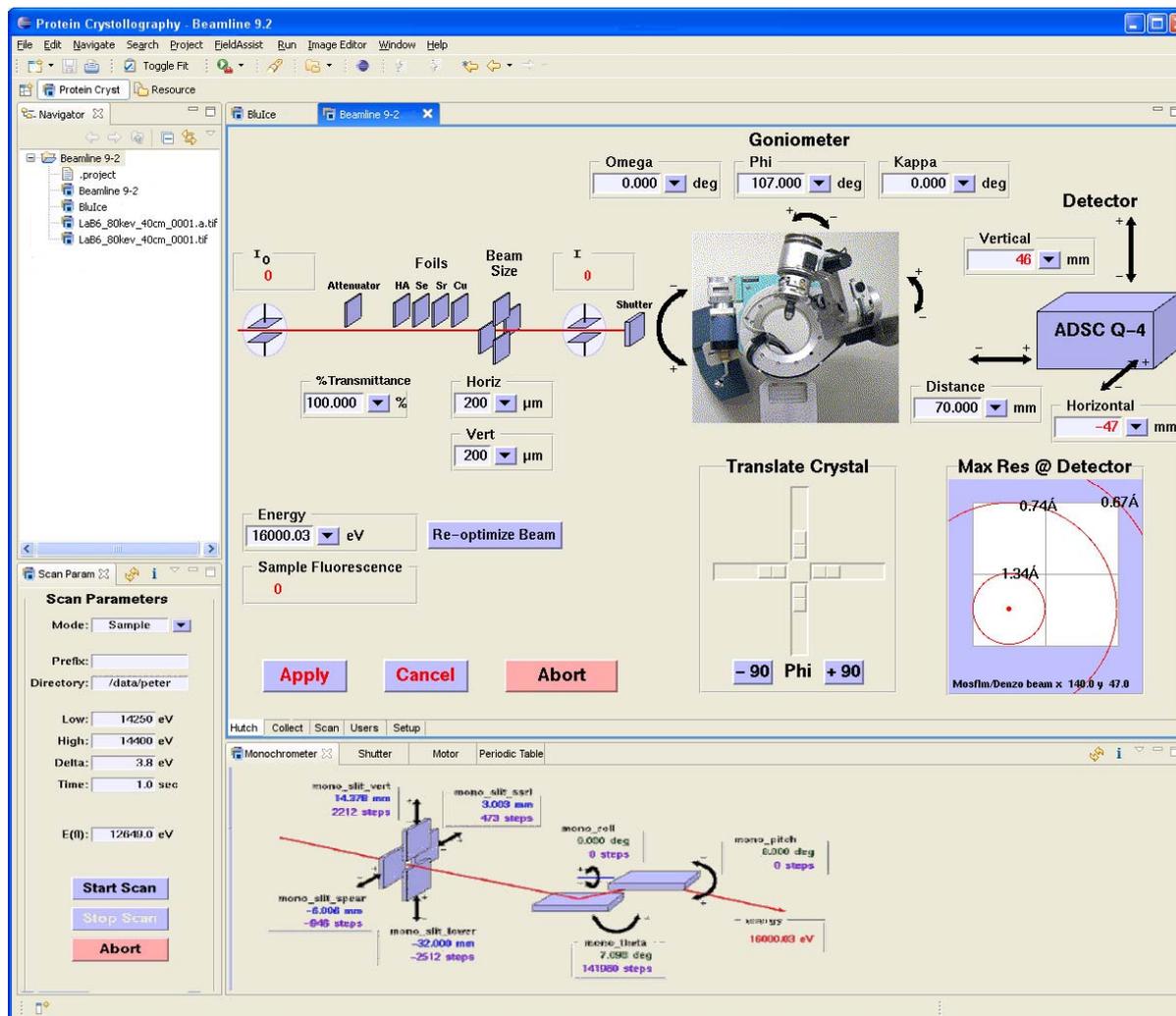
A Perspective Can be a Single Application



Area Data Editor



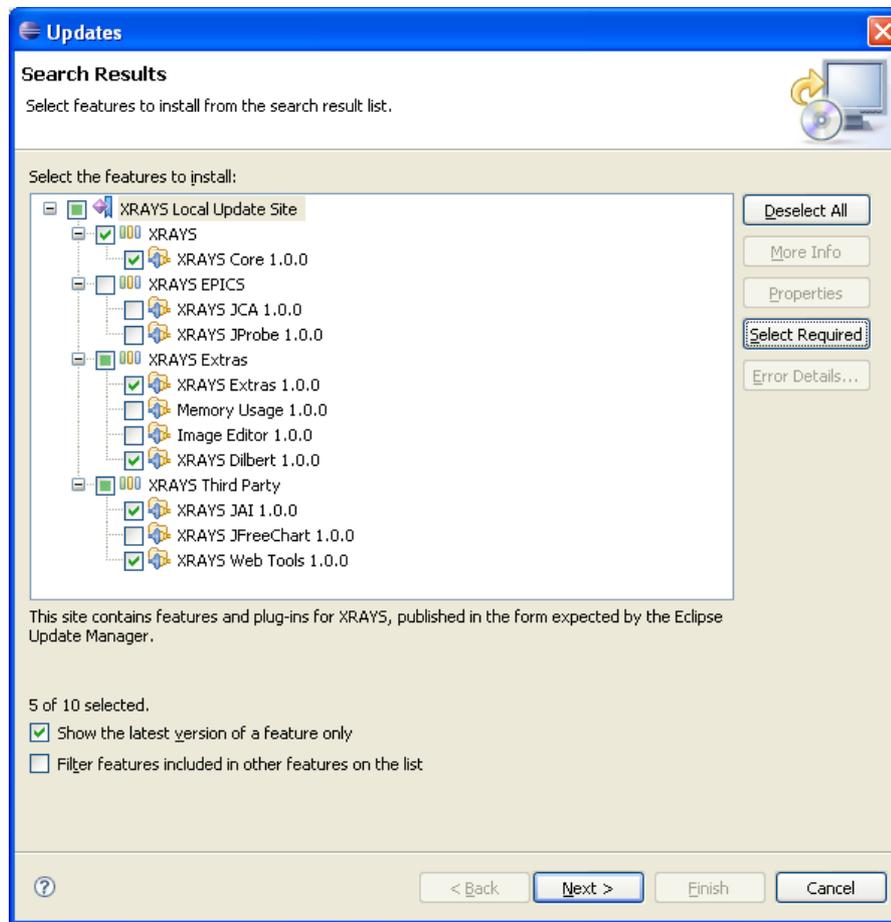
X-Ray Experiment

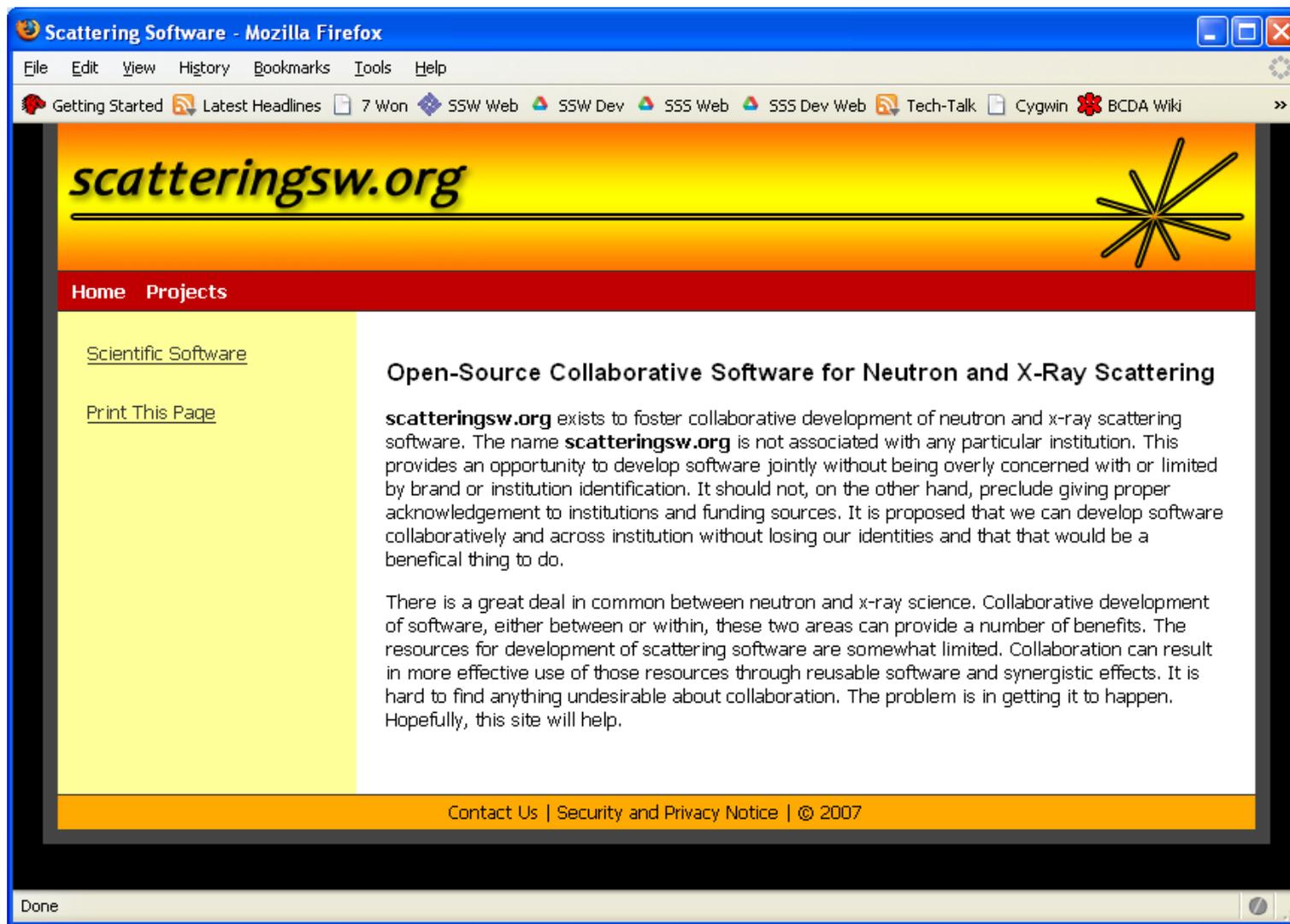


Images from: BLU-ICE and the Distributed Control System, NOBUGS III, January 2000

Deployable Plug-ins

- These Eclipse projects are available through the Eclipse update mechanism at <http://www.scatteringsw.org/XRAYS/updates>
- 10 features in 4 categories consisting of ~25 plug-ins
- Work in Progress
 - Not thoroughly tested
 - Not fully documented
- Not really scientific software
 - But potentially useful





The screenshot shows a Mozilla Firefox browser window with the title "Scattering Software - Mozilla Firefox". The address bar contains the URL "scatteringsw.org". The browser's menu bar includes "File", "Edit", "View", "History", "Bookmarks", "Tools", and "Help". The toolbar shows several bookmarks: "Getting Started", "Latest Headlines", "7 Won", "SSW Web", "SSW Dev", "SSS Web", "SSS Dev Web", "Tech-Talk", "Cygwin", and "BCDA Wiki".

The website header features the text "scatteringsw.org" in a large, stylized font on a yellow-to-orange gradient background. To the right of the text is a logo consisting of several black lines radiating from a central point, resembling a star or a scattering pattern.

Below the header is a red navigation bar with the text "Home" and "Projects".

The main content area is divided into two columns. The left column has a yellow background and contains two links: "Scientific Software" and "Print This Page". The right column has a white background and contains the following text:

Open-Source Collaborative Software for Neutron and X-Ray Scattering

scatteringsw.org exists to foster collaborative development of neutron and x-ray scattering software. The name **scatteringsw.org** is not associated with any particular institution. This provides an opportunity to develop software jointly without being overly concerned with or limited by brand or institution identification. It should not, on the other hand, preclude giving proper acknowledgement to institutions and funding sources. It is proposed that we can develop software collaboratively and across institution without losing our identities and that that would be a beneficial thing to do.

There is a great deal in common between neutron and x-ray science. Collaborative development of software, either between or within, these two areas can provide a number of benefits. The resources for development of scattering software are somewhat limited. Collaboration can result in more effective use of those resources through reusable software and synergistic effects. It is hard to find anything undesirable about collaboration. The problem is in getting it to happen. Hopefully, this site will help.

At the bottom of the page, there is a footer with the text "Contact Us | Security and Privacy Notice | © 2007".

Acknowledgements

- Pete Jemian – Group Leader of the APS BCDA group
 - Is invaluable in helping provide the direction for this effort.

Thank You

*This has been an
APS Scientific Software Section
Presentation*

Thank You

*This has been an
APS Scientific Software Section
Presentation*