Project Plan

Mike McKerns, Caltech

• Building the Project Plan
• The Software Production Process
• Project Management, Integration, & Change
Goals & Objectives

- The goal of DANSE is to build a software system for neutron scattering research that:
  - integrates the basic data reduction capabilities that are available today
  - enables new types of science in all major subfields of neutron scattering research
  - provides a coherent framework onto which software components can be added by scientists
  - is maintainable by the SNS software group before the end of the project
Project Milestones

Prototype Components                August, 2004 (achieved)
Baseline Design Review       December, 2005 (achieved)
Project Start                  June, 2006 (achieved)
Prototype Release                             October, 2006
Rebaseline                                   April, 2007
α-Release                                October, 2007
Critical Review                                      February, 2008
β-Release                              October, 2008
Construction Review                       April, 2009
Release 1.0                    October, 2009
Operational Readiness Review                     April, 2010
Release 1.1                    October, 2010
Project Complete                       May, 2011
Identifying the Project Scope

• Planning begins: January, 2002
  - requirements collection with neutron scientists
  - software workshops & reviews
  - email polls & surveys

• First draft for five-year project plan: November 2003
  - identified project tasks
  - problems with estimation of effort, risk, & duplication of effort

• First draft for Scope Baseline (WBS): August 2004
  - risk assessment & mitigation plan
  - resource leveling, task dependencies, & external drivers

• Project Descoped ($16.7M – $11.9M): March 2006
  - identified critical tasks & scope contingency
Building a Scope Baseline (task-by-task)

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Sr. Scientist</th>
<th>Postdoc</th>
<th>Programmer</th>
<th>Tech Writer</th>
<th>Grad Student</th>
<th>Undergrad</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modify input file writer</td>
<td>0.4</td>
<td>0.1</td>
<td>0.4</td>
<td>0.4</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Write bindings to modules</td>
<td>0.1</td>
<td>0.4</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Modify output file reader</td>
<td>0.4</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Test/use case 1: energy minimization</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Test/use case 2: constrained molecular dynamics</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Test/use case 3: normal mode analysis</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Test/use case 4: operations on dynamic trajectories</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Test/use case 5: point charge fitting</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Test/use case 6: molecular surface calculations</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Write tutorial text</td>
<td>0.60</td>
<td>0.60</td>
<td>0.60</td>
<td>0.60</td>
<td>0.60</td>
<td>0.60</td>
<td>0.60</td>
</tr>
<tr>
<td>Write tutorial code</td>
<td>0.60</td>
<td>0.60</td>
<td>0.60</td>
<td>0.60</td>
<td>0.60</td>
<td>0.60</td>
<td>0.60</td>
</tr>
<tr>
<td>Write and integrate reference documentation</td>
<td>2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Scientific review</td>
<td>0.20</td>
<td>0.20</td>
<td>0.20</td>
<td>0.20</td>
<td>0.20</td>
<td>0.20</td>
<td>0.20</td>
</tr>
<tr>
<td>User testing and subsequent maintenance</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>
The Scheduling Process

... is highly constraint-driven
# Five Year Software Construction Plan

<table>
<thead>
<tr>
<th>Software Package</th>
<th>Start Date</th>
<th>End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMTK</td>
<td>03/24/08</td>
<td>03/22/09</td>
</tr>
<tr>
<td>Scattered Intensity</td>
<td>09/06/07</td>
<td>01/24/08</td>
</tr>
<tr>
<td><strong>Basic Data Structures</strong></td>
<td>12/08/06</td>
<td>02/12/07</td>
</tr>
<tr>
<td>NeXus data containers</td>
<td>04/19/07</td>
<td>09/05/07</td>
</tr>
<tr>
<td><strong>Common Array Manipulations</strong></td>
<td>02/11/07</td>
<td>04/19/07</td>
</tr>
<tr>
<td>Advanced Array Manipulations</td>
<td>03/20/10</td>
<td>06/22/10</td>
</tr>
<tr>
<td>Data Comparator</td>
<td>11/30/09</td>
<td>02/25/10</td>
</tr>
<tr>
<td>Data Catalog</td>
<td>03/21/07</td>
<td>07/09/07</td>
</tr>
<tr>
<td>Catalog</td>
<td>02/12/10</td>
<td>07/02/10</td>
</tr>
<tr>
<td><strong>Basic Optimization Algorithms</strong></td>
<td>08/29/06</td>
<td>03/28/07</td>
</tr>
<tr>
<td>Optimization Algorithms</td>
<td>12/18/08</td>
<td>01/17/10</td>
</tr>
<tr>
<td>NumPy</td>
<td>08/14/07</td>
<td>09/17/07</td>
</tr>
<tr>
<td>FFTW</td>
<td>08/31/07</td>
<td>09/17/07</td>
</tr>
<tr>
<td>Random number generator</td>
<td>09/27/10</td>
<td>10/07/10</td>
</tr>
<tr>
<td>Symbolic Math</td>
<td>11/30/10</td>
<td>11/30/10</td>
</tr>
<tr>
<td>Error Propagation Algorithms</td>
<td>08/25/08</td>
<td>10/14/08</td>
</tr>
<tr>
<td>cctbx</td>
<td>07/22/07</td>
<td>12/13/07</td>
</tr>
<tr>
<td>FOX</td>
<td>03/20/08</td>
<td>07/09/08</td>
</tr>
<tr>
<td>CrysPML</td>
<td>03/02/10</td>
<td>07/11/10</td>
</tr>
<tr>
<td>PowderCell</td>
<td>06/02/10</td>
<td>10/24/10</td>
</tr>
<tr>
<td>ObjCry++</td>
<td>11/29/07</td>
<td>03/20/08</td>
</tr>
<tr>
<td>Crystal Structure Container Class</td>
<td>05/30/09</td>
<td>05/29/10</td>
</tr>
<tr>
<td>OpenBabel</td>
<td>05/02/07</td>
<td>07/01/07</td>
</tr>
<tr>
<td>Molden</td>
<td>04/24/08</td>
<td>06/20/09</td>
</tr>
<tr>
<td><strong>PyMol</strong></td>
<td>08/04/06</td>
<td>09/06/06</td>
</tr>
<tr>
<td>ViewMol</td>
<td>12/29/08</td>
<td>02/04/10</td>
</tr>
<tr>
<td>DRawWIt</td>
<td>08/31/10</td>
<td>10/26/10</td>
</tr>
<tr>
<td>VMD</td>
<td>06/26/07</td>
<td>08/22/07</td>
</tr>
<tr>
<td>Additional CIF reader</td>
<td>09/21/10</td>
<td>11/16/10</td>
</tr>
<tr>
<td>Structure functions from powder diffracto</td>
<td>01/07/08</td>
<td>11/11/08</td>
</tr>
<tr>
<td>Bragg-peak intensities from single-crysta</td>
<td>09/31/10</td>
<td>12/24/10</td>
</tr>
<tr>
<td>Instrument specific tasks</td>
<td>11/10/08</td>
<td>02/25/10</td>
</tr>
<tr>
<td><strong>Full profile model refinement in real and i</strong></td>
<td>09/05/06</td>
<td>12/29/08</td>
</tr>
<tr>
<td>Plug in architecture for carrying out joint</td>
<td>03/21/07</td>
<td>01/06/08</td>
</tr>
<tr>
<td>Model Independent peak fitting</td>
<td>01/28/07</td>
<td>03/25/07</td>
</tr>
<tr>
<td>Rietveld and PDF modeling extensions</td>
<td>02/03/10</td>
<td>02/12/11</td>
</tr>
</tbody>
</table>
DANSE Project Release Plan

- Project milestones include yearly DANSE project releases.

- DANSE is an application-driven component framework, hence applications and components are the deliverables.

- Project releases are a collection of applications, components, and any pyre framework development.

- Snapshot of completed packages.
Management at the Package Level

• Identify & sequence work products
  - Vision Statement
  - Architecture Specification & Behavior Specification
  - Code Implementation & Test Cases
  - ...

• **Software Production Plan** is incremental, with phases of development
  - Inception [< 5% complete]: mostly planning and assessment
  - Elaboration [5-20% complete]: mostly requirements capture
  - Construction [20-90% complete]: mostly code and test implementation
  - Transition [> 90% complete]: documentation, deployment, and certification
- review and quality control process integrated with development

- review and testing process enable efficient software engineering

- producing documentation and specifications early in the process minimizes rework and eases user manual generation

- full code construction and core documentation by beta release
Software Review Process

• **Design Review**
  - vision statement summarizing purpose, functionality, and acceptance criteria
  - UML & auto-generated interface documentation for architecture specification
  - refinement of requirements and use cases into behavior specification

• **Code Review**
  - both component/application and test code inspected
  - automated test for quality standards, manual inspection for code functionality

• **Reviews conducted by Architect, PM, SubPI, developers**
  - review performed during breakout session
  - review action items entered to project tracking system
  - approval archived to versioning repository
  - followed by reassessment of task cost and schedule
Software Testing Process

- automated builds tied to code repository & reporting system
- automated tests of process and standards adherence
- manual inspection of functional behavior

- tests developed concurrent to code
- rapid bug identification, notification
- overall product quality and developer efficiency increases
Earned Value Management

- Earned Value Management provides estimate of project progress and adherence to project plan
  - Sequenced tasks in WBS provide coupling of expected (scheduled) work completion to expected cost
  - Software Production Plan identifies basis for estimation of % complete, and is captured in the Statement of Work
  - Project Manager and Subproject Leader use project tracking tools to map work done to % complete
  - Monthly reports from subproject business systems provide actual costs

- EVM serves as a guide for project change decisions
• Successful development with distributed resources requires centralized management.
Communication Processes

• Coordination and Planning Mechanisms
  - mailing lists and ticketing system
  - weekly technical and management meetings; breakout sessions
  - control boards

• Production Mechanisms
  - versioning repositories
  - standardized development and release process
  - 1-on-1 VNC sessions

• Configuration Control and Reporting Mechanisms
  - repository tracking
  - feature and bug tracking
  - automated builds and testing
Tracking the Development Process

- Integrated tracking system monitors development & bugs
  - based on standard bug tracking system
  - bug reports from users and developers
  - WBS tasks added as feature requests
  - review action items entered as feature requests
  - error reports from automated builds and test harness

- Repository check-ins provide additional information
  - check-in messages reference to archived design documentation
  - finer-grain detail provided by repository 'diff' command

- EVM metrics will be tested for weighting task completion
  - tracking based on production process (design, review, test, release)
  - tracking based on functional requirements and specifications
Project Plan must be Agile

- Estimation of effort used to build the WBS should be continually estimated based on the % complete and EVM
- Software development is an agile environment, and python is particularly dynamic
- By developing in an open source community, the packages that we utilize will also change dynamically (hopefully, for the better!)
Control Boards

• Membership:
  Aivazis, Anderson, Fultz, McKerns

• Change Control: McKerns, Chm.

• Configuration Control: Aivazis, Chm.

• Risk Control: Fultz, Chm.

• Project Change Requests submitted to alter the project plan
Collaborating with DANSE

• Initiate contact
• Present software demonstration, use cases, and architecture specifications for your software
• Exchange requirements with us
• Decide on common ground
• (Project Change Request Approved)
• Exchange repository and project access
• Collaboration is now mutually beneficial and no duplication of effort
End Presentation