Management of Change

Mike McKerns, Caltech

- Project Status
  - Schedule, Cost, and Earned Value
  - Issues and a Path Forward
- Project Management Infrastructure
  - Agility of the WBS
  - Software Quality and Metrics of Completion
  - Integrated Development and Management

Project Growth Profile

Please inquire to Mike McKerns
or Brent Fultz for access to
information contained on this slide.

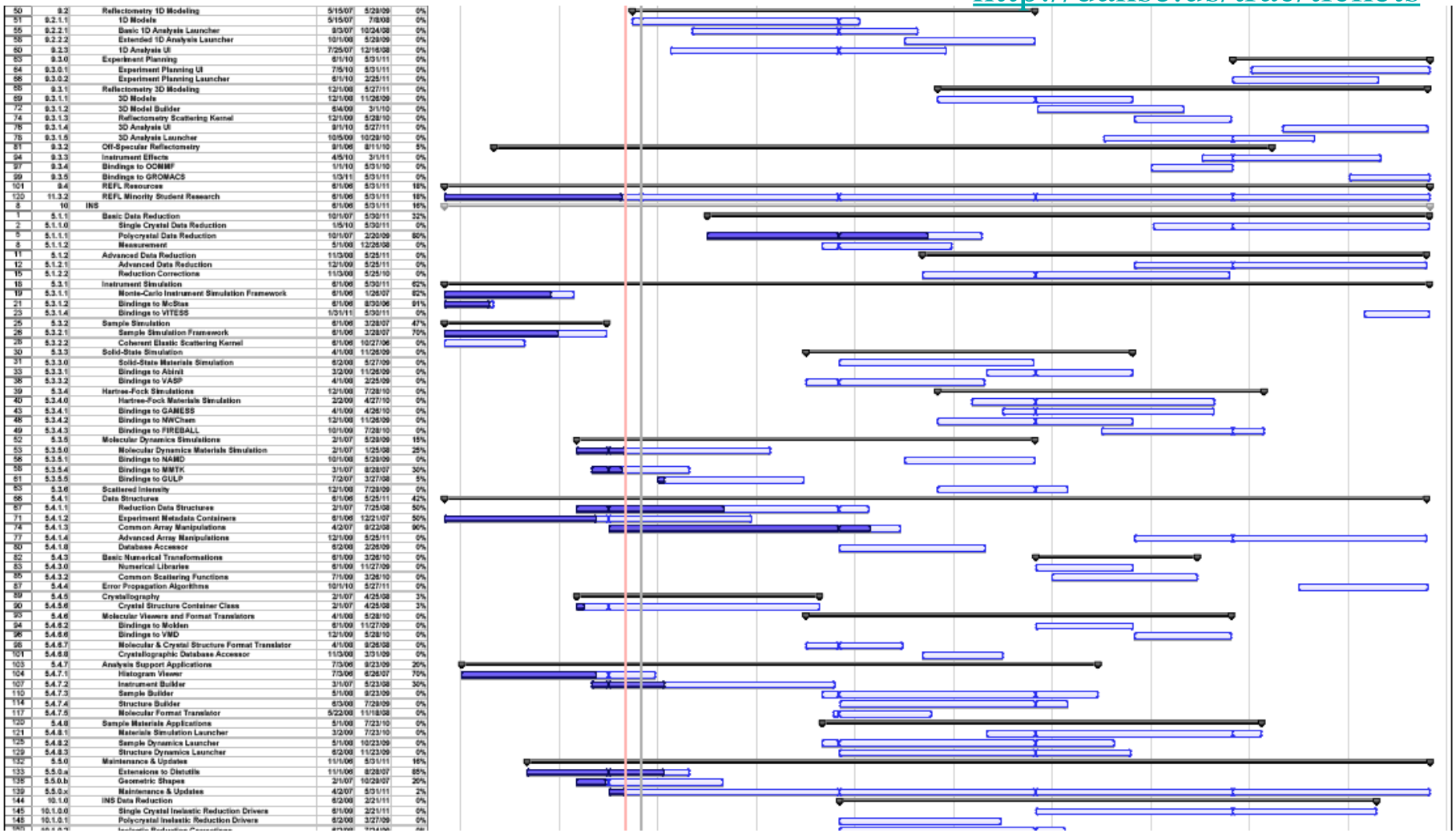
- year 1: infrastructure and training
- year 2: initial DANSE releases
- year 3: project-wide integration
- year 4-5: project robustness

Project Organization Reflected in WBS

WBS 2: Project Infrastructure	McKerns, Fultz
WBS 3: Software Quality	McKerns, Aivazis
WBS 4: Central Services	Aivazis
WBS 5: Common Algorithms	*
WBS 6: Diffraction	Billinge
WBS 7: Eng. Diffraction	Ustundag
WBS 8: SANS	Butler
WBS 9: Reflectometry	Kienzle
WBS 10: Inelastic Scattering	Fultz
WBS 11: Education & Outreach	*

Five Year Software Construction Plan

http://danse.us/trac/tickets



WBS & Earned Value Reporting

- half-year subproject funding increments tied to a SOW
- SOW built from a half-year slice of the WBS
- submit (contractual) monthly reports on scheduled WBS tasks
- earned value metrics derived from reported cost and activity

WBS	Task	Type	Account	Year / Month	Direct Costs	Indirect Costs
7.2.1.1	Modular integration of finite element analysis (FEA) software – Al	LA	ENGDIF	2007/04	$ 1,402	$ 423
7.2.1.2	Input/output parameters for FEA	LA	ENGDIF	2007/04	$ 1,402	$ 423
7.2.2	Self-Consistent Modeling	LA	ENGDIF	2007/04	$ 1,261	$ 270
7.2.3	Inverse Problem Analysis	LA	ENGDIF	2007/04	$ 1,261	$ 270
7.3.2	Microstructure Simulation	LA	ENGDIF	2007/04	$ 2,200	$ 467
7.3.X	Maintenance and Updates	LA	ENGDIF	2007/04	$ 961	$ 254
7.4.1	Local administrative support	LA	ENGDIF	2007/04		
7.4.2	Computing hardware and software	MS	ENGDIF	2007/04	$ 147	$ 69
7.4.3	Travel	TR	ENGDIF	2007/04		
7.4.4	Communications	MS	ENGDIF	2007/04		
7.4.X	Subawards	SA	ENGDIF	2007/04		
11.3.5	Minority Student Research Position	LA	ENGDIF	2007/04		
				TOTALS	$ 8,634	$ 2,176

Budget Period: Jun 1, 2006 to May 31, 2007 (NCE)
Reporting Period: Apr 1, 2007 to Apr 30, 2007

WBS	Task	Account	Baseline Start Date	Baseline End Date	Projected % Complete at End y2p1	Prior Reported % Complete	Actual % Complete
7.2.1.1	Modular integration of finite element analysis (FEA) software – Al	ENGDIF	09/02/10	10/28/10	100%	69%	70%
7.2.1.2	Input/output parameters for FEA	ENGDIF	10/02/10	12/13/10	100%	65%	68%
7.2.2	Self-Consistent Modeling	ENGDIF	09/22/10	04/26/12	70%	29%	32%
7.2.3	Inverse Problem Analysis	ENGDIF	12/31/10	04/10/12	65%	19%	21%
7.3.2	Microstructure Simulation	ENGDIF	05/02/11	02/23/14	18%	0%	7%

Budget Period: Jun 1, 2006 to May 31, 2007 (NCE)
Reporting Period: Apr 1, 2007 to Apr 30, 2007

Project Status (year 1)

Please inquire to Mike McKerns
or Brent Fultz for access to
information contained on this slide.

Project Status (year 2)

Please inquire to Mike McKerns
or Brent Fultz for access to
information contained on this slide.

Schedule Lag: Issues

Please inquire to Mike McKerns
or Brent Fultz for access to
information contained on this slide.

Schedule Lag: Issue Resolution

- Current issues are “one-time” instead of “ongoing”
- Good hires to make progress on key software infrastructure
  - runtime component framework
  - global optimization framework
  - Monte Carlo instrument simulation framework
  - UI generator and web services
- Rebaseline the WBS
  - no reduction of scope
  - reschedule around flagship applications and release cycle
  - account for work recently done on tasks scheduled in out years

Begin with High-level Definition & Design

- Ensure identification of all tasks
  - list science objectives and workflow requirements at the subgroup-level
  - assemble science and workflow into target release products (deliverables)
  - each deliverable must be associated with a task
  - identify if task is clearly states software and science requirements
  - identify if software has the desired interfaces

- Ensure a clear definition of all deliverables & interfaces
  - identify all of the project deliverables (especially the flagship applications)
  - break deliverables into well-defined conceptual components
  - specify information to exchange between the components
  - draw UML diagram of components and data objects for each deliverable

Capture Vision in WBS Dictionary

5.3.3.0 Solid-State Materials Simulations: *Ab initio solid-state calculations of electronic structure are practical on systems with 100 atoms or so in a periodic unit cell. These quantum mechanical calculations provide electron wavefunctions within the local density approximation and its extensions. At minimum, we will support calculations of optimized atom positions, electron densities, densities of state, dynamical matrices, and total energy and forces.*

5.3.3.0.x API Implemented as Abstract Classes
5.3.3.0.x Materials Simulation Data Objects
5.3.3.0.x Result Analysis and Steering

5.3.3.1-4 Bindings to Common Solid-State Materials Simulation Packages: *We will provide bindings to Abinit, VASP, CPMD, and WIEN2K.*

5.3.3.n.x Input/Output File Parsers
5.3.3.n.x Full Simulation Driver
5.3.3.n.x Replace Parsers w/ Data Object Bindings
5.3.3.n.x Replace Full Simulation Driver w/ Drivers for each Step?

5.3.4.0 Hartree-Fock Materials Simulations: *Hartree-Fock codes with Gaussian basis sets are also practical for systems of 100 atoms or so. These codes are quite mature, and can provide both the positional information required for diffraction studies and the vibrational eigenfrequencies required for inelastic scattering. At a minimum, we will support calculations of optimized atom positions, electron densities, densities of state, dynamical matrices, and total energy and forces.*

5.3.4.0.x API Implemented as Abstract Classes
5.3.4.0.x Materials Simulation Data Objects
5.3.4.0.x Result Analysis and Steering

5.3.4.1-3 Bindings to Common Hartree-Fock Materials Simulation Packages: *We will provide bindings to GAMESS, NWChem, and FIREBALL.*

5.3.4.n.x Input/Output File Parsers
5.3.4.n.x Full Simulation Driver

Michael McKerns 1/16/07 11:25 AM
Comment: At minimum, we will deliver components that launch a full simulation through the bindings, and an API for these calls. At maximum, we will deliver bindings to individual simulation steps, with the launching and configure done in python… and includes a UI and some results analysis. Also, convenience functions.

Michael McKerns 1/18/07 2:22 PM
Comment: Group into: 1) SCF calculation (total energy, electron density, forces), 2) geometry optimization, 3) phonon properties (density of states, dynamical matrix). Should calculation of phonon properties should be in it's own separate task? …I think so. Should a calculation of G(R) or G(R,t) be available here or as a separate task?

Michael McKerns 1/16/07 4:06 PM
Comment: Correct/complete? This list depends on if we choose the max or min implementation of this task.
What specific API are needed? What about handling pseudopotentials?

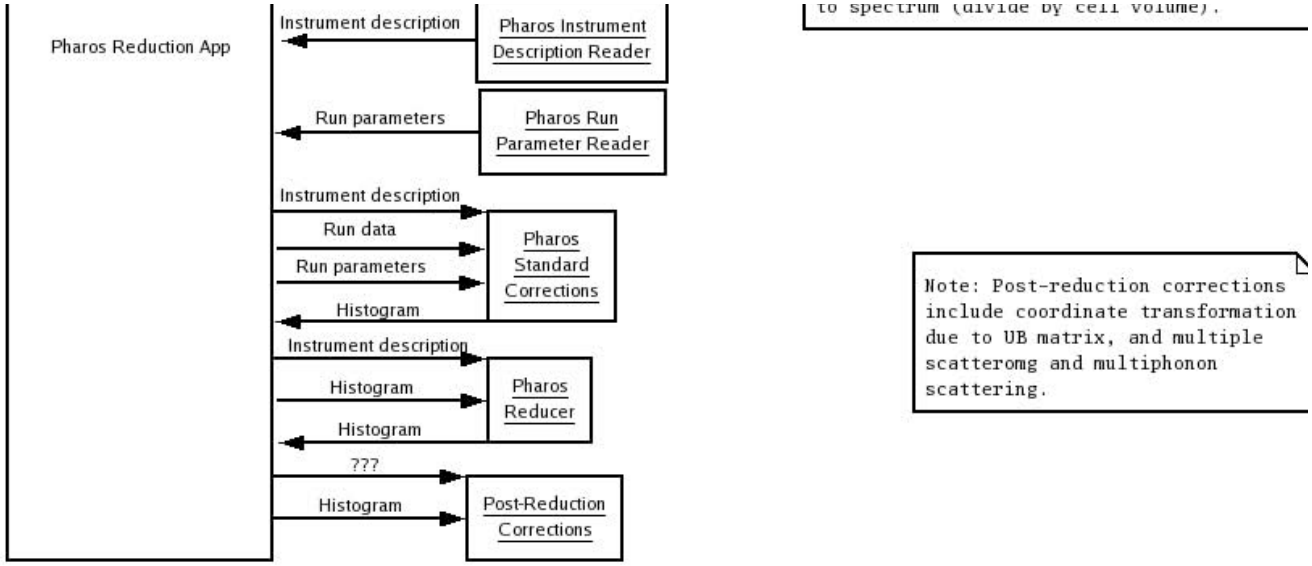
Michael McKerns 1/16/07 11:20 AM
Comment: Executable configure and launch.

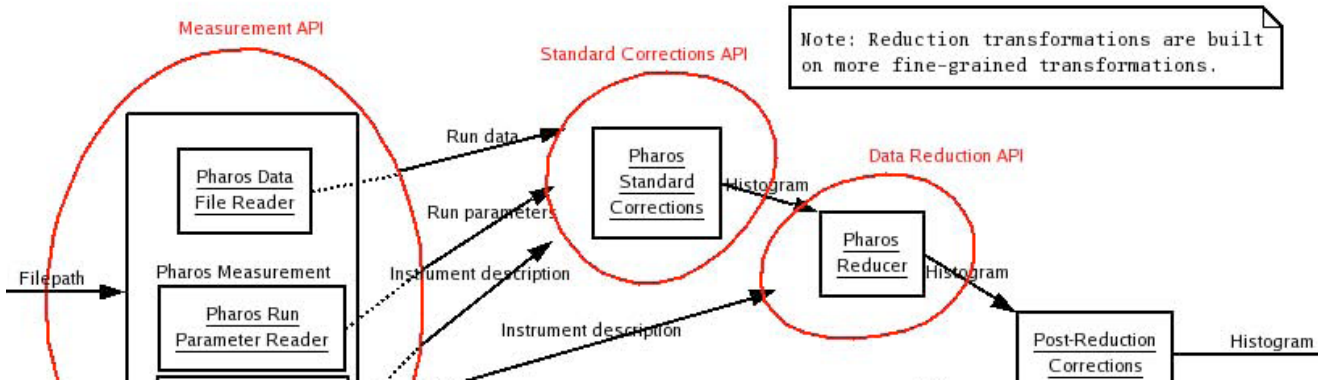
Michael McKerns 1/16/07 4:02 PM
Comment: Depends on min or max implementation, as in 5.3.3.0. Max implementation (drivers at each step) is not likely; it's be most useful for geometry optimization, but probably not worth the effort.

Michael McKerns 1/16/07 11:25 AM
Comment: Same as 5.3.3.0.

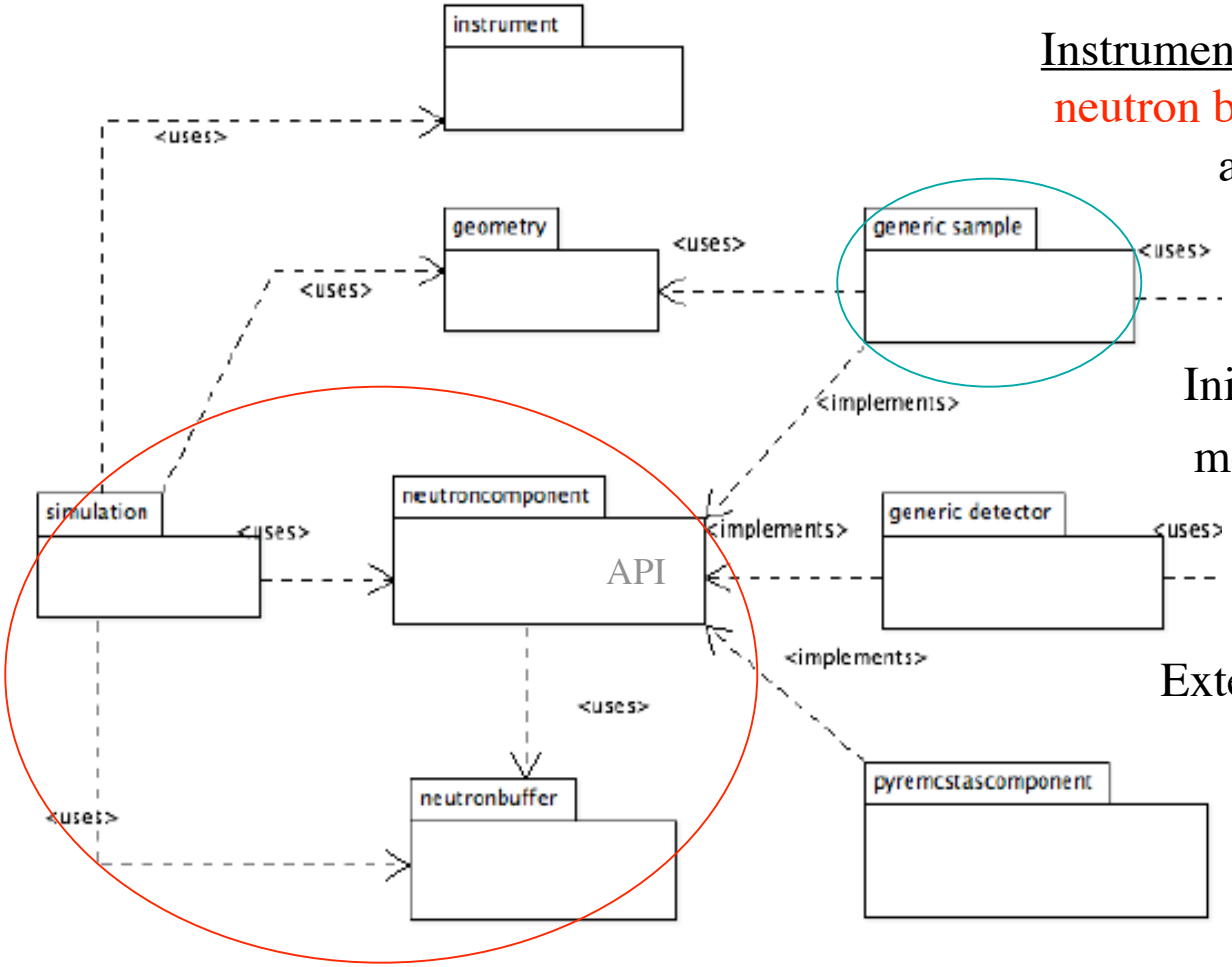
Michael McKerns 1/16/07 11:28 AM
Comment: Same as 5.3.3.0.

Capture High-level Specifications





Use Good Software Engineering Design



Instrument simulation is simulation engine, neutron buffer, and neutron components… and uses instrument and geometry

Initially, McStas components provide monitors, guides, samples, detectors

Extend by building neutron component with generic sample construction

Capture Software Engineering in the WBS

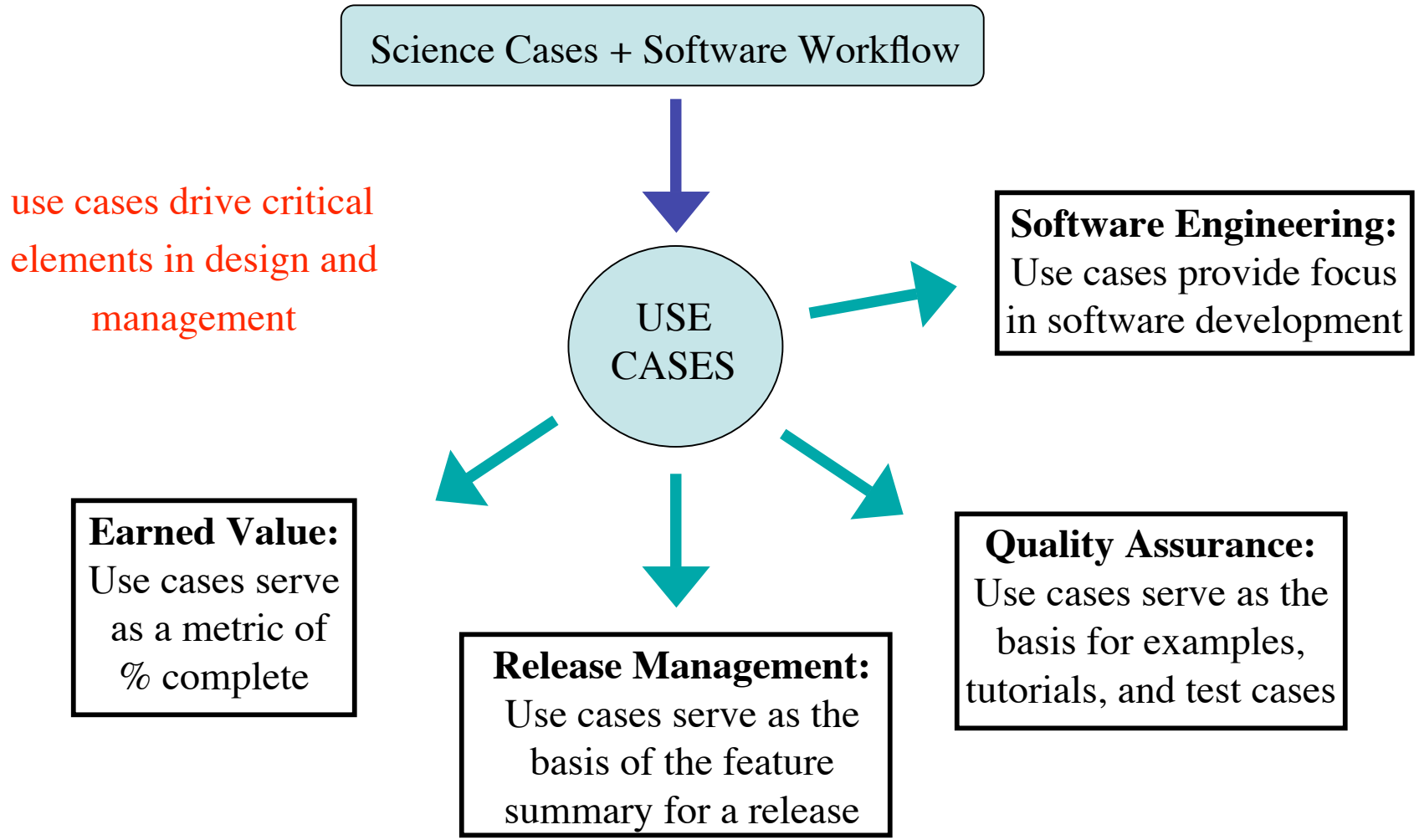
- Building tasks with narrower scope
  - creates additional tasks (but not scope)
  - tasks tend to be shorter duration with better-defined scope
  - code tends to be more reusable
  - development is more efficient and there is less rework
  - creates less surprises

- Schedule tasks once WBS dictionary is updated
  - developers focus on more than one task
  - more tasks run simultaneously, thus durations can be lengthened
  - more akin to development in a small dynamic group
  - less well-defined tasks must be scheduled later in the project

Outlook for Updating the WBS

- Current WBS is good, but not complete enough…
  - Needs to capture flagship applications and higher-level tasks
- **"Functionality" is stable, but requirements are dynamic**
  - Science requirements are well understood
  - Software requirements are still being defined
- Path Forward:
  - We have science objectives well in hand (i.e. science scope is stable)
  - We need to schedule the WBS to facilitate planned project-wide integration
  - We need to better organize the WBS around release products
  - We need to add more emphasis on software workflow to the WBS

Centralized Role of Use Cases



Software Quality & Metrics of Completion

- Estimates of complete based on:
  - completion of science use cases
  - completion of software use cases
  - verification by a collection of testable data objects
  - status of accompanying documentation

- Completion based on review of release products:
  - released code (i.e. applications, libraries, modules)
  - user's guide and developer's guide
  - release website

Metrics and the Review Process

Top three risks:	See Requirement document: - Speed - Parameter access - Use of models from other modules
Top three open issues:	Deployment strategy (to be discussed with central services)

General Observations

Overall, there is a lot of good content. Task status is clear and on schedule. The high-level architecture diagrams look solid. A more detailed high-level behavior diagram for the in-scope application would help clarify the system boundary and may also better define the API. However, there are a few critical elaboration details missing; thus, several action items have been created, and must be closed before the elaboration phase is declared complete. Data structures have been identified as "model objects"; more details will be needed in the construction phase.

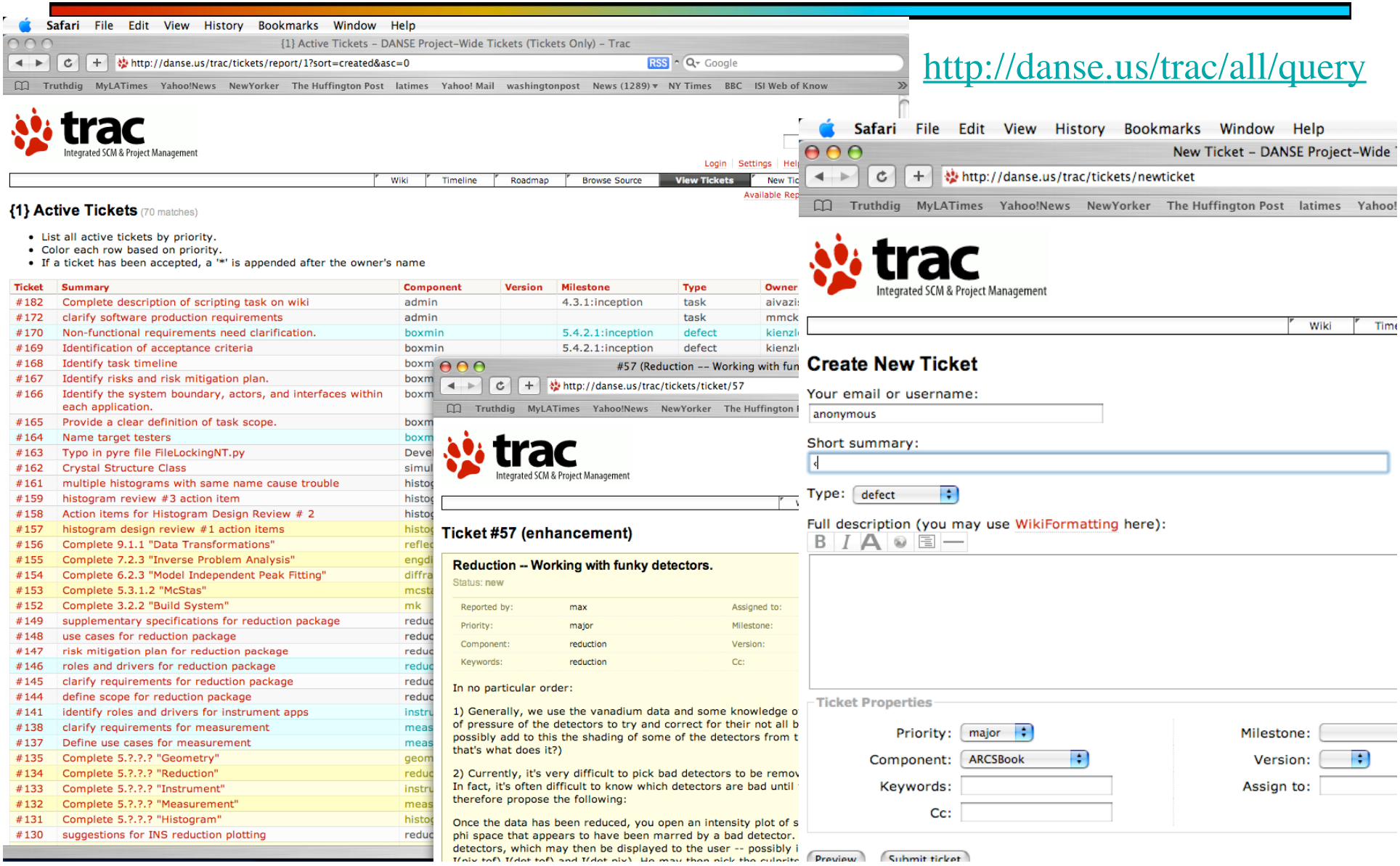
Actions

ID	Action Item	Assigned To	Due By
1	Inform reviewers/testers of their roles (http://danse.us/trac/tickets/ticket/175)	Butler	05/07/2007
2	List in-scope 1D models (http://danse.us/trac/tickets/ticket/176)	Butler	05/07/2007
3	Provide evidence of prototype application (http://danse.us/trac/tickets/ticket/177)	Butler	05/07/2007
4	Generate inheritance diagram for each in-scope component (http://danse.us/trac/tickets/ticket/178)	Butler	05/07/2007
5	Capture behavior specifications for each in-scope	Butler	05/07/2007

Action items are a result of a review or audit. Reviews are closed when all action items are closed.

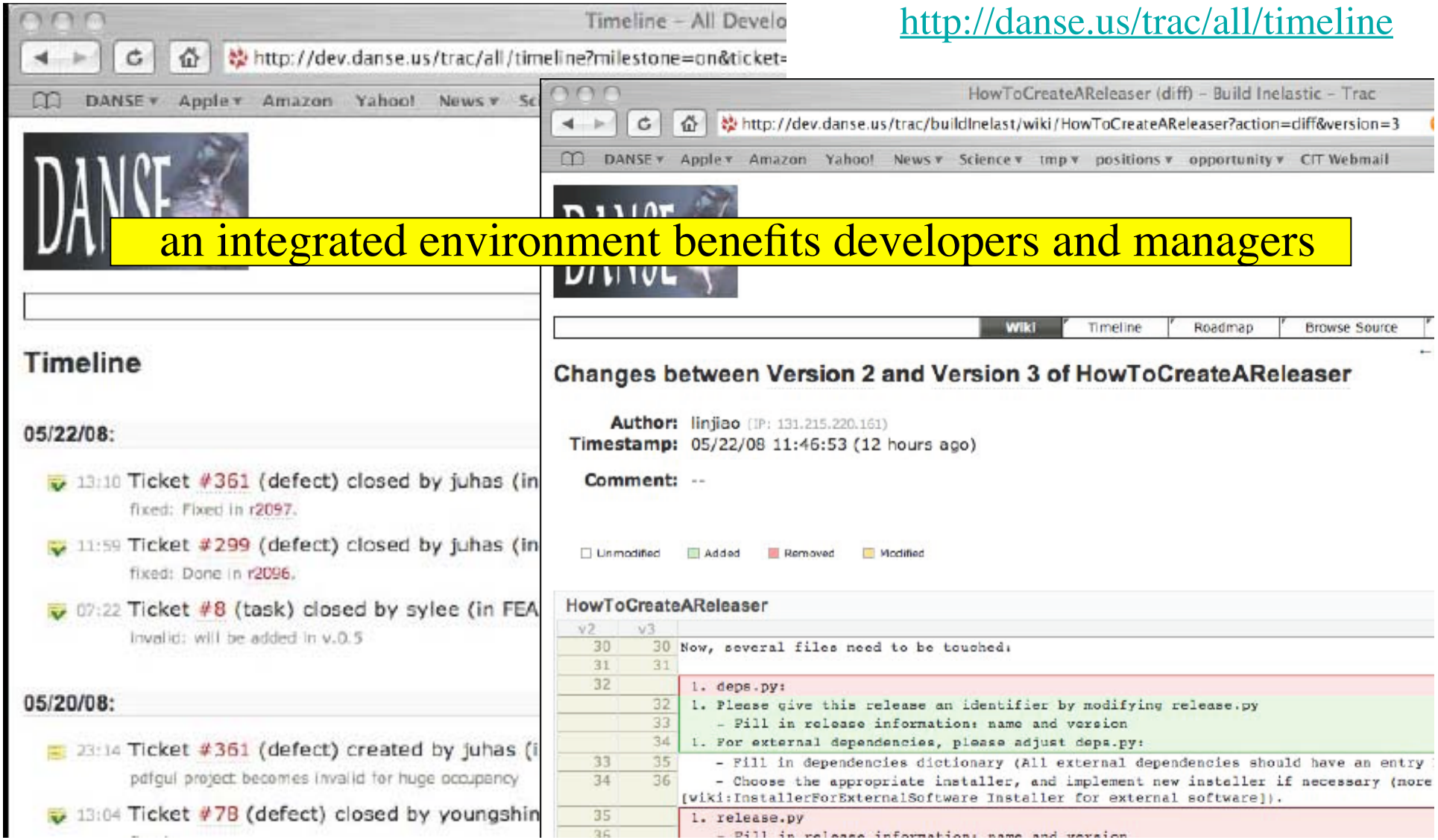
Action items can be treated as tickets in the online bug and issue tracking system.

Submitting a Ticket



Integrated Change Tracking System

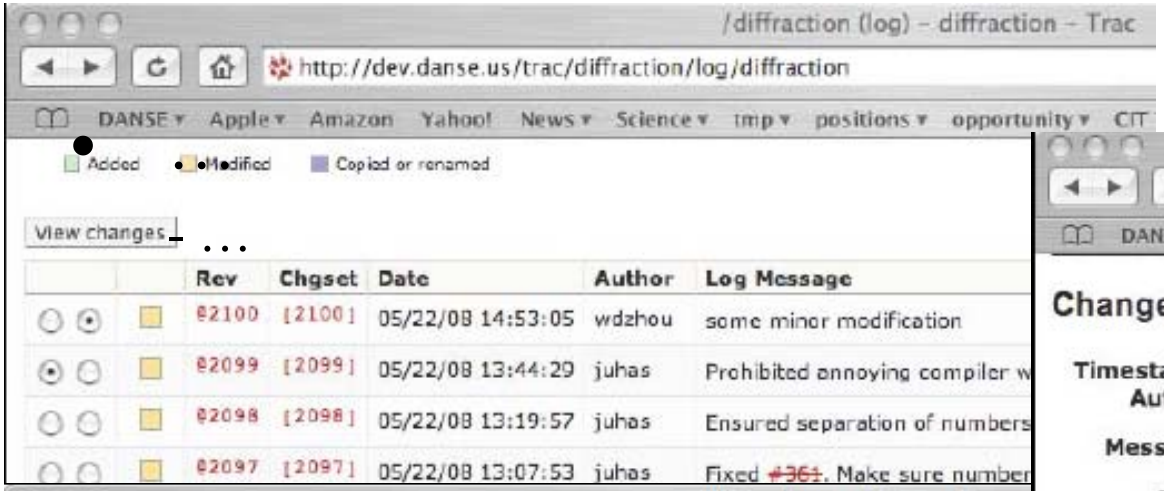
http://danse.us/trac/all/timeline

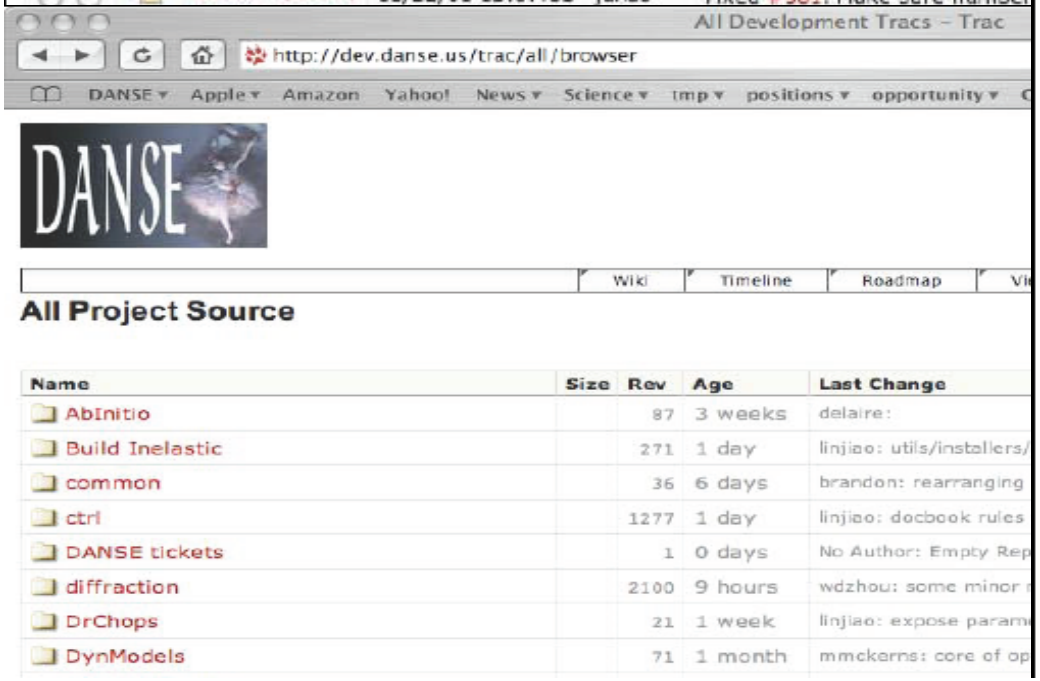


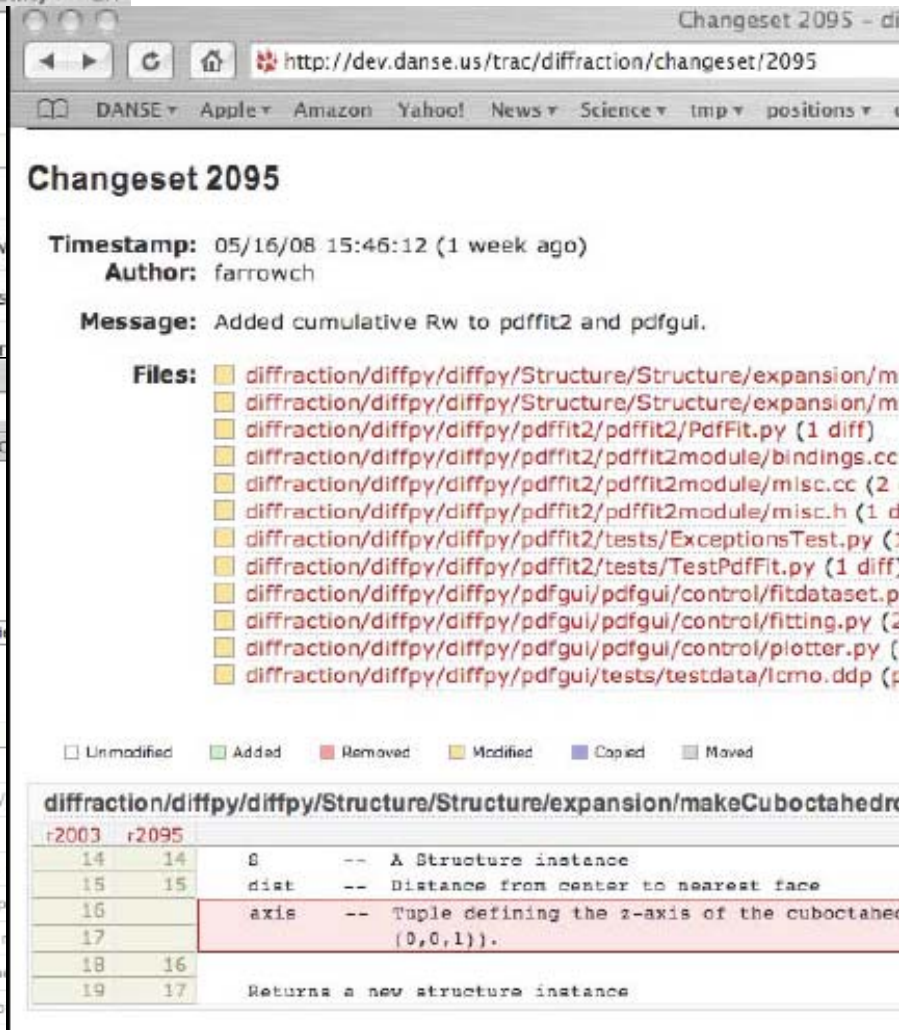
an integrated environment benefits developers and managers

Browseable Code & Document Source

http://danse.us/trac/all/browser

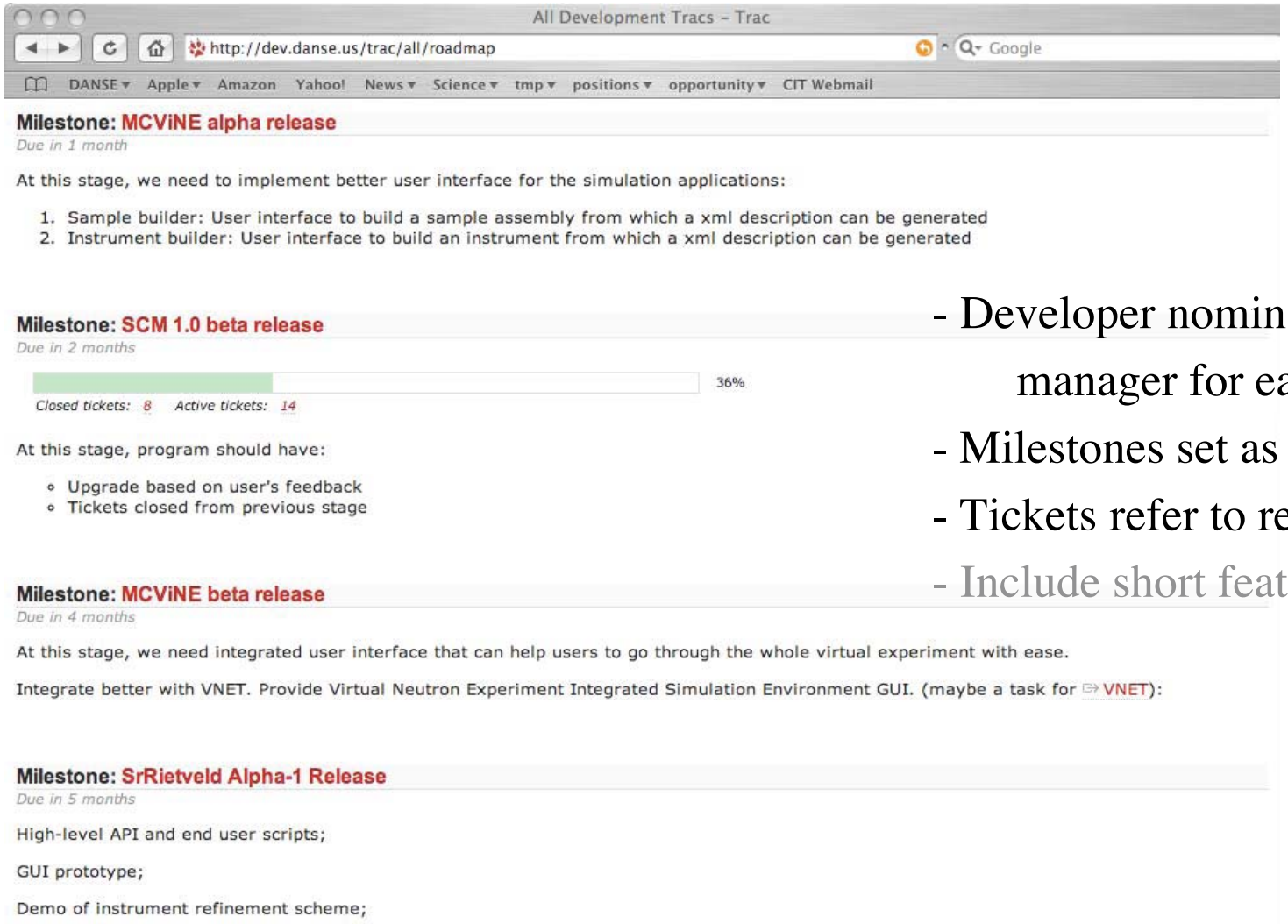






Roadmap for Release Management

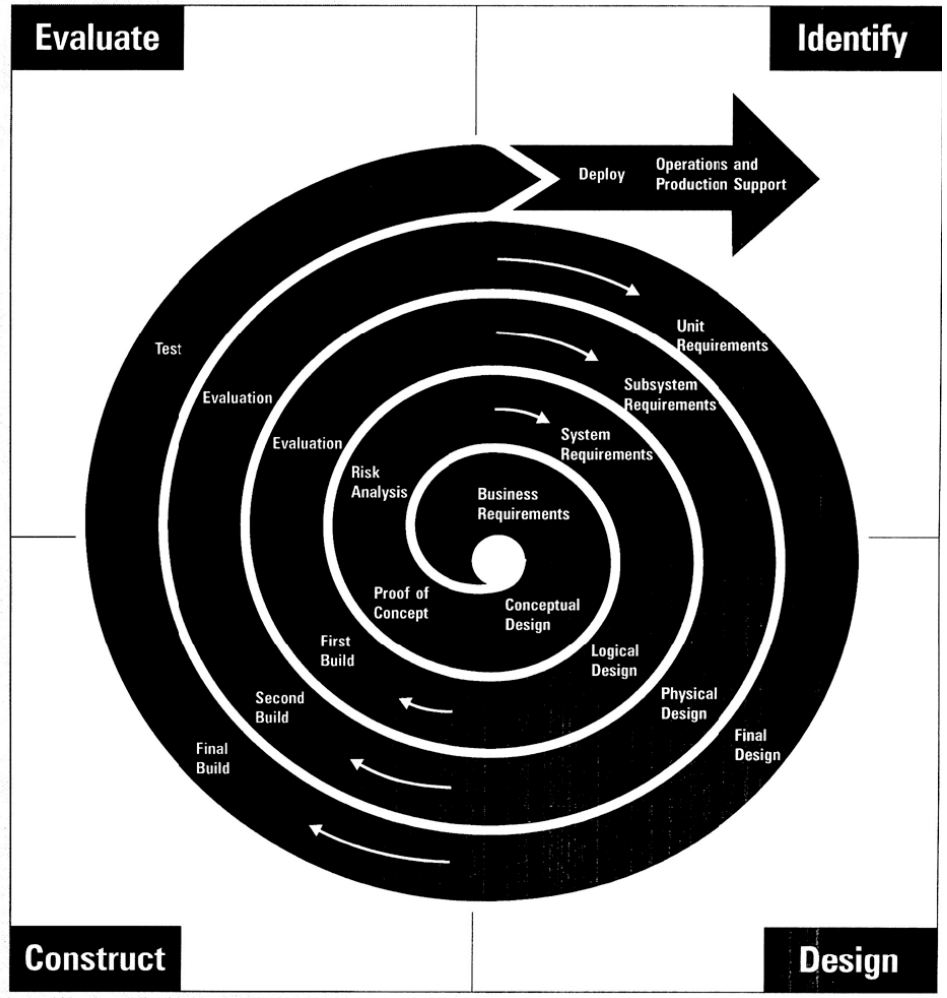
http://danse.us/trac/all/roadmap



- Developer nominated as release manager for each package
- Milestones set as release versions
- Tickets refer to release milestones
- Include short feature summary

DANSE uses Good Development Practices

- Early design & definition
- Use cases & workflow
- Prototyping
- Iterative code growth
- Minimize refactoring
- Plan for “plug-in” development
- Utilize abstraction
- Document all interfaces
- Monitor and contribute to 3rd party solutions



Benefit of Good Development Practices

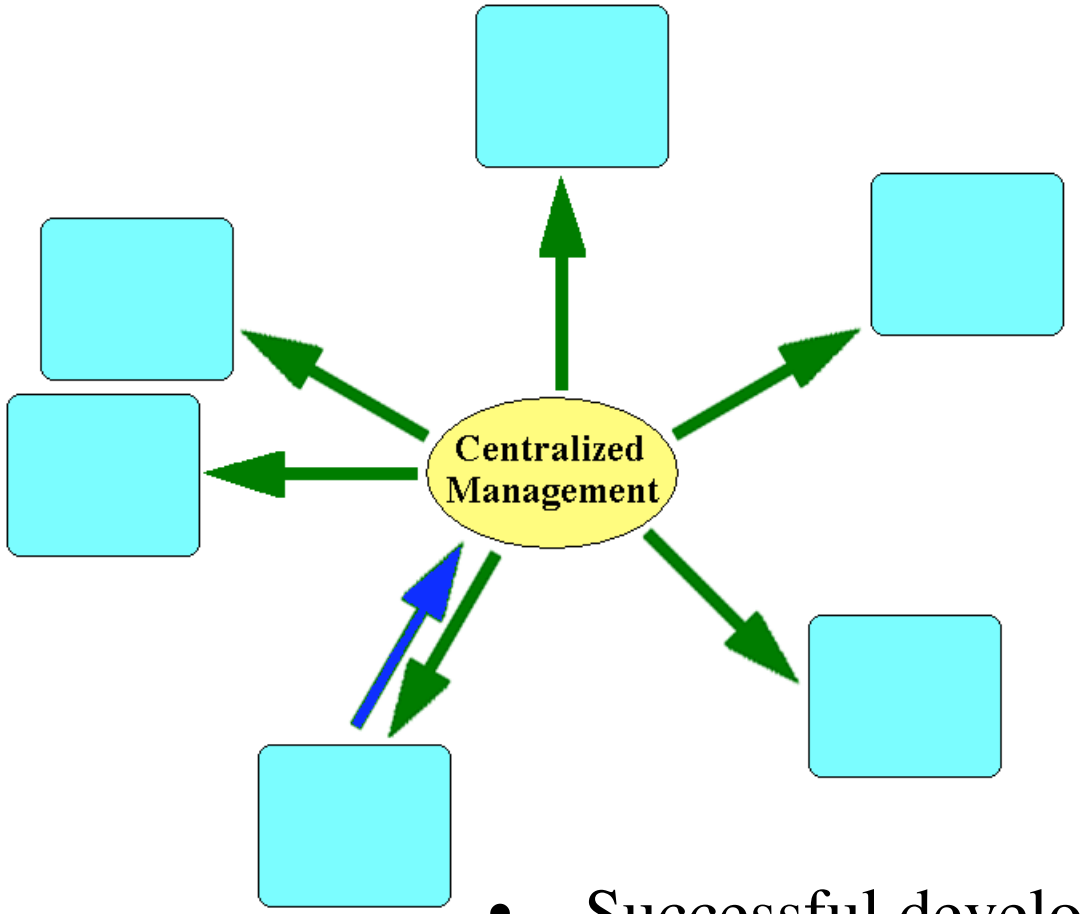
- Better software engineering enables greater reusability and flexibility, which in turn enables DANSE to be more efficient in building robust and durable software that will enable new science.

Summary: Risk in Software Projects

- Misperceptions of specifications are the biggest source of risk in software projects
- Good specifications and software design in the WBS (revisited as appropriate -- at least every 6 months)
- Subproject leaders know what science they want, our developers write use cases to flesh-out the specifications
- Reviews, audits, & quality assurance further minimizes risk
- Adopt commercial practices for scientific software development and management

End Presentation

Resource Management Tools



- centralized code repository
- email ticketing system
- systematic tracking
- structured design process
- build/testing infrastructure
- release management
- standardization
- quality assurance
- reporting mechanisms
- review & design steering

- 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