

Mylar

A Task Focused UI for Eclipse

For Eclipse users

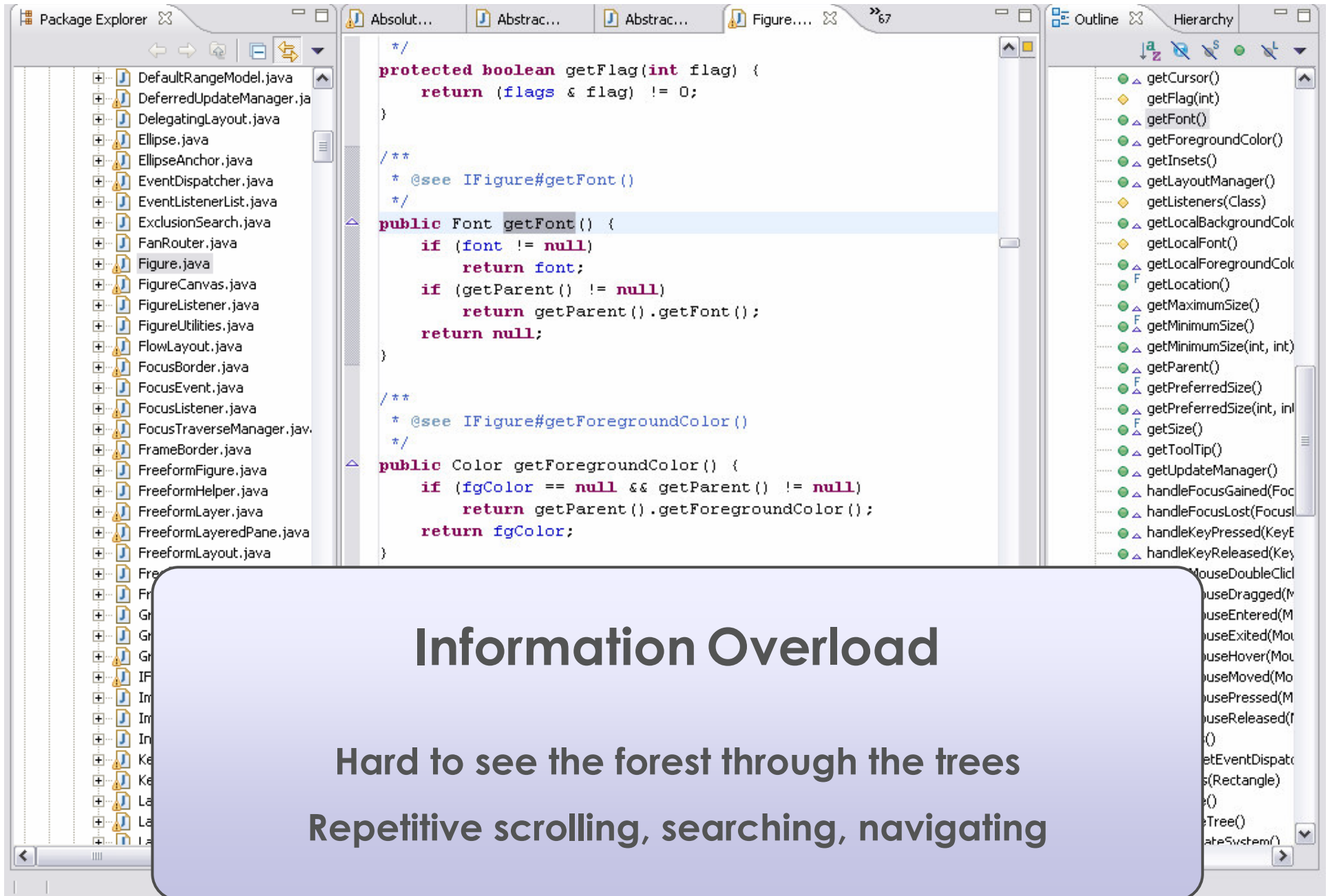
- Integrates task management
- Automates context management

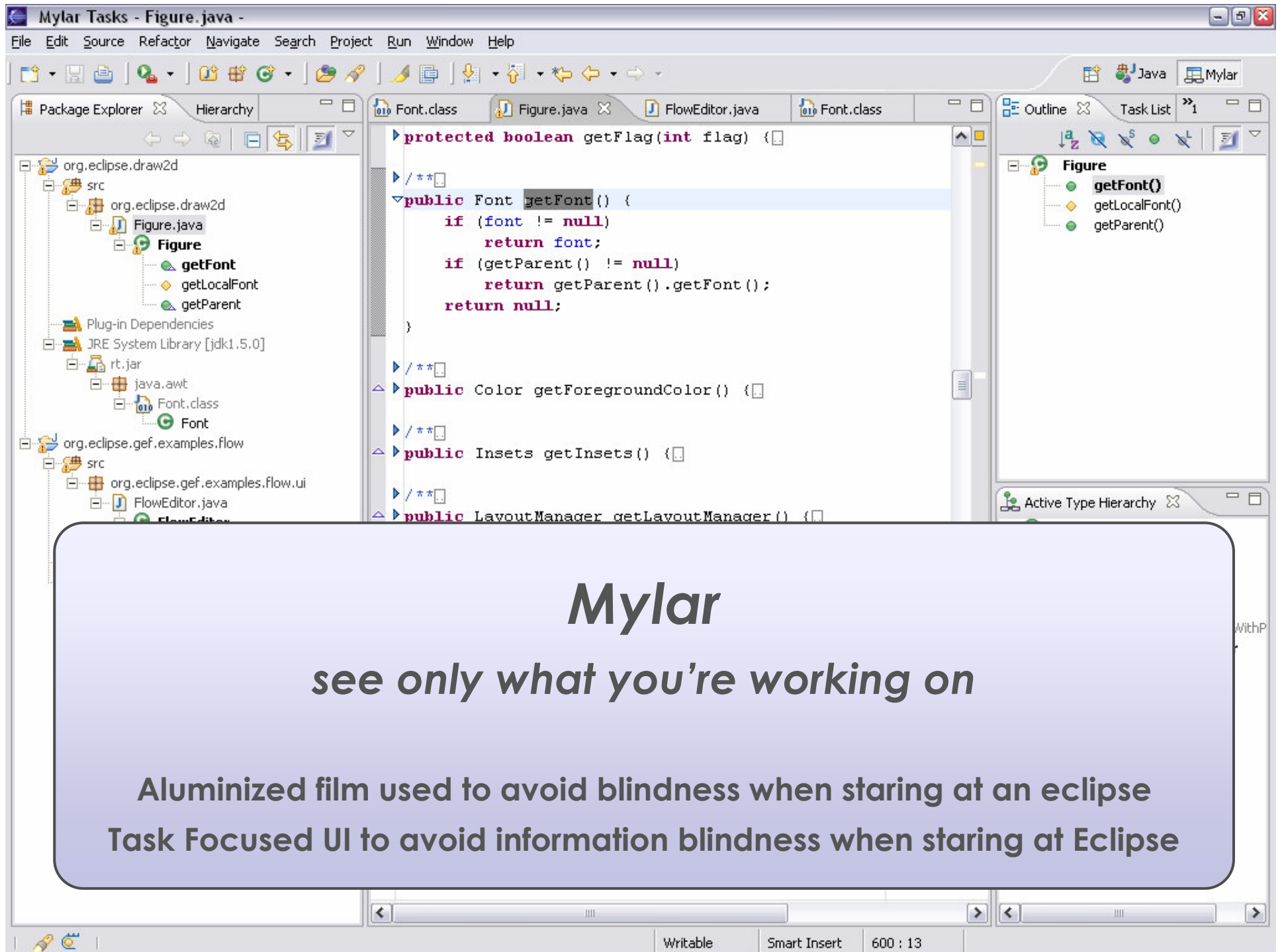
For developers and integrators

- Provides framework for tasks and contexts

Mik Kersten, project lead

Eclipse Webinar, October 25, 2006, made available under the EPL v1.0





Overview

Working with Mylar

- Tasks and contexts make working with large systems easier
- Bugs/issues/tickets/tasks are integrated and easy to manage

Demos highlighting key features

- Working with task context
- Eclipse integration (SDK)
- Repository integration (Bugzilla)

Building on Mylar

- Internals & architecture
- Framework & APIs

Demo 1: Tasks

Without Mylar

- Work with various web UIs to manage bugs/issues/tasks

With Mylar

- Task management is integrated
- Similar to source repositories
- Get persistence, offline editing, notifications

Tasks

Connectors

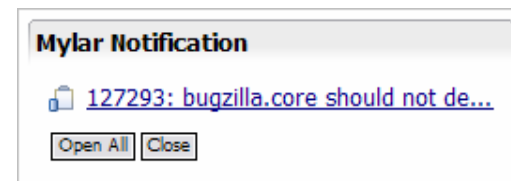
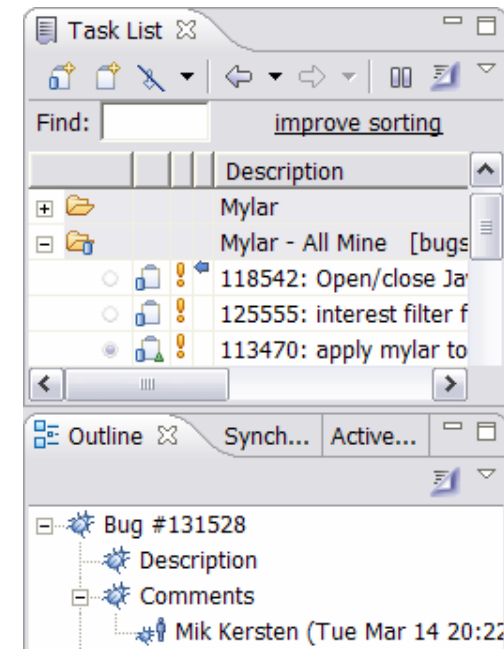
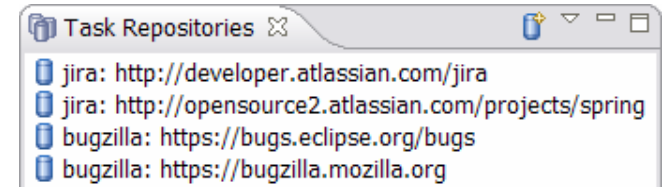
- Similar to source repositories
- Support Bugzilla JIRA and Trac

Tasks

- Local
- Web linked
- Repository queries
- Authoring, offline editing

One integrated task list

- Personalized notes, reminders
- Archive, filters, notifications



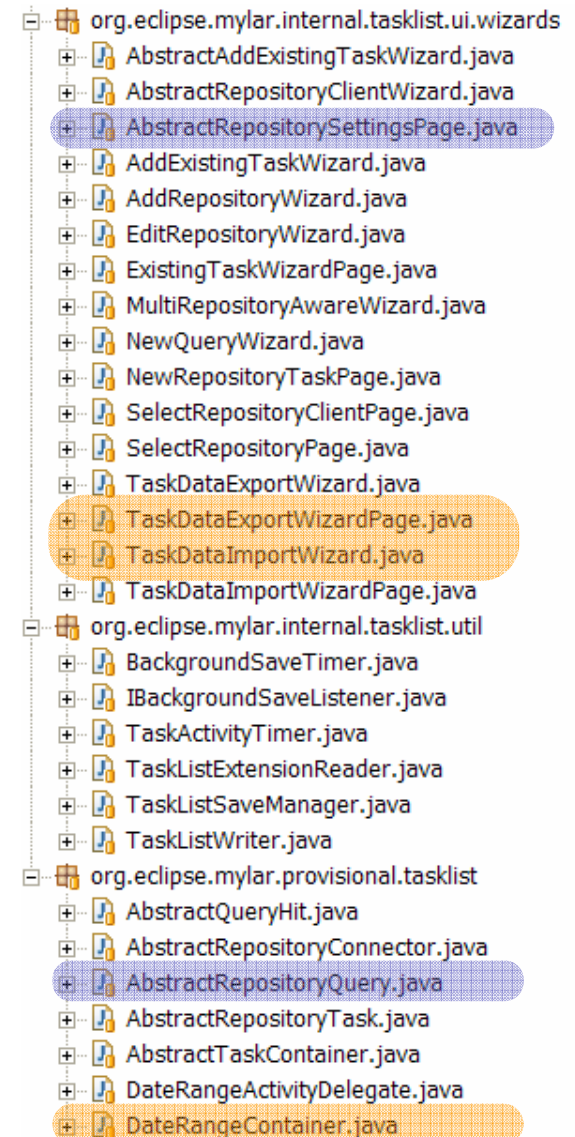
Demo 2: Task context

Without Mylar

- Manually manage context
- Use working sets, filters

With Mylar

- Indicate what task you're working on
- Programming activity forms context for that task
- Context becomes explicit in the UI



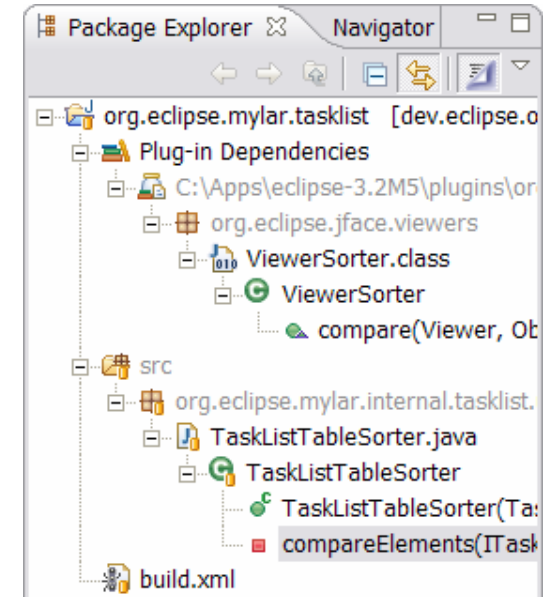
Task context

Tasks

- User-defined unit of work, e.g. bug report

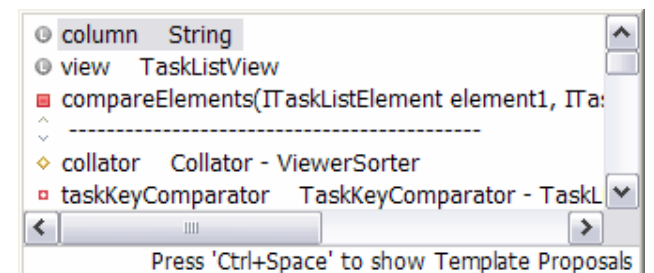
Context

- Mylar monitors your interaction
- Creates degree-of-interest model
- What you touch is in your context
- Actively managed as you work
- Stored and easy to recall



Focused UI

- Views: filtering, decoration
- Editors: folding, content assist
- Context switching, editor management



Demo 3: Integration

Automatic change sets

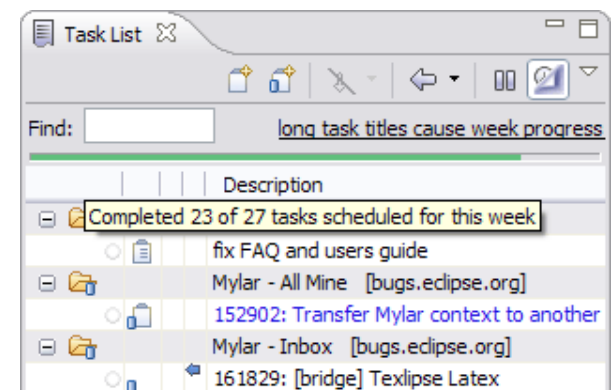
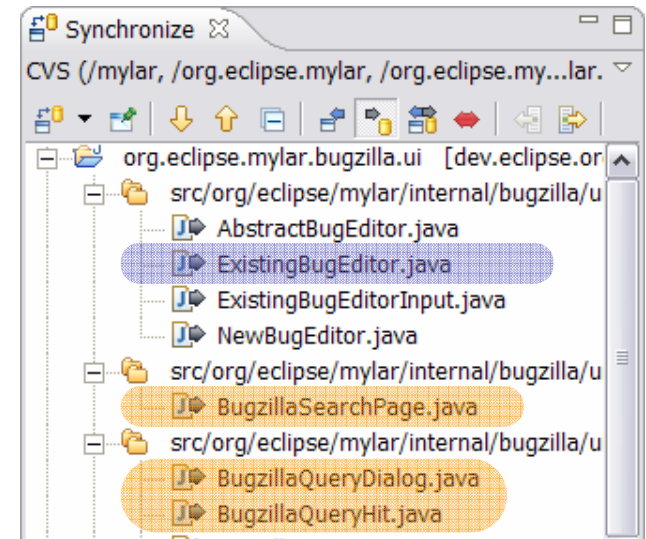
- Sometimes you only want to commit or update a subset

Sharing context

- Attaching and retrieving context

Focusing the task list

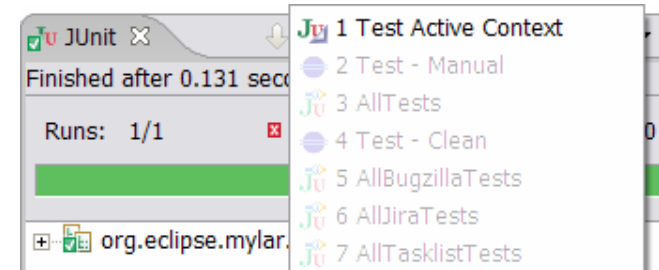
- Scheduling, planning, progress



There's more...

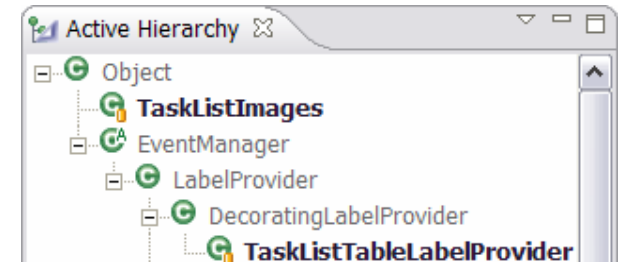
Automation

- Testing context via Active Test Suite



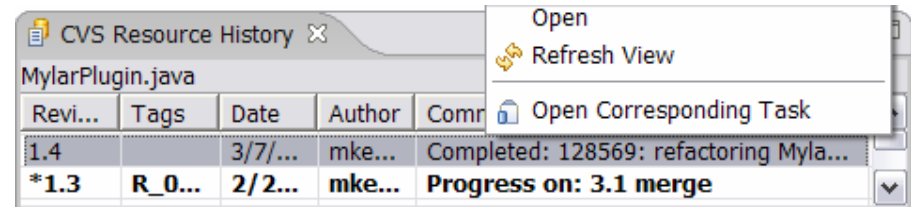
Context views

- Active Type Hierarchy, Active Search



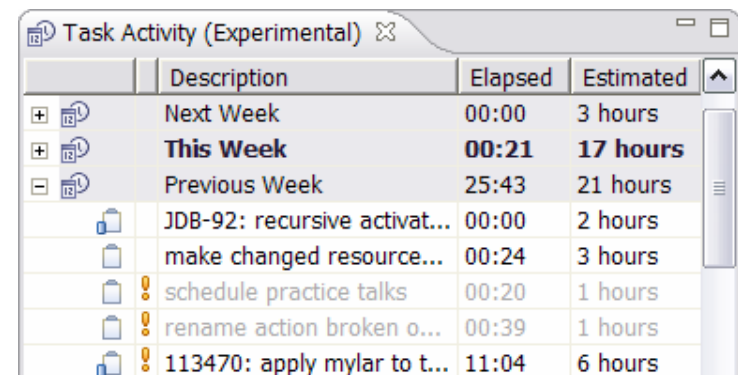
Everything is linked

- Tasks to context to resources



Easy to build on

- XP planning via Task Activity View



Changing how we work with Eclipse

What you need to do

- Buy into working with tasks

Once tasks are explicit

- Dramatic reduction in information overload
- Information you need to get work done is at your finger tips
- Multi-tasking and recalling old tasks become effortless
- UI automation (e.g. working sets, search, commit messages)
- UI for task management is consistent and integrated
- Keeps you in Eclipse and out of your browser and inbox

under the hood

Mylar's context model

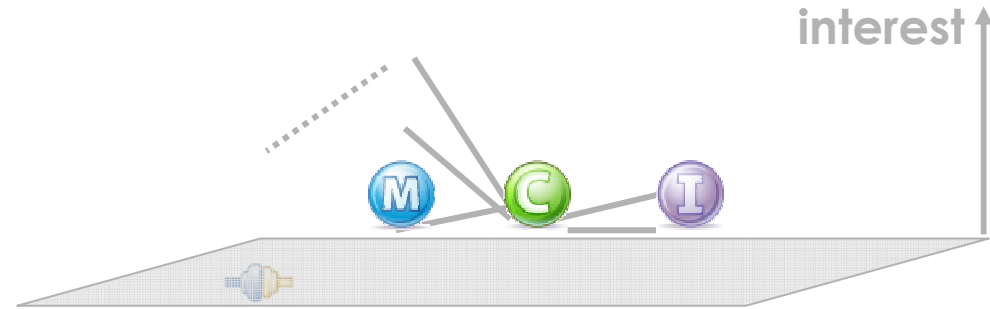
Interaction history

- InteractionEvent stream
- Origin, handle, type, date



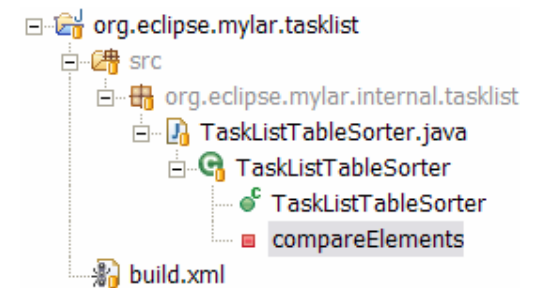
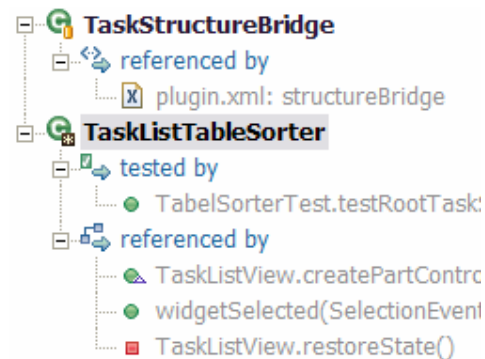
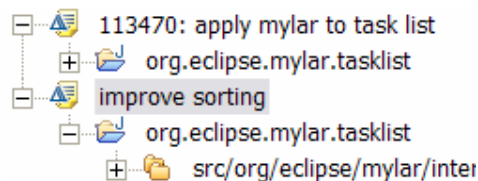
Context (Core)

- Degree-of-interest graph
- Degree-of-separation scope



Display (UI)

- Views, editors, files



Frameworks

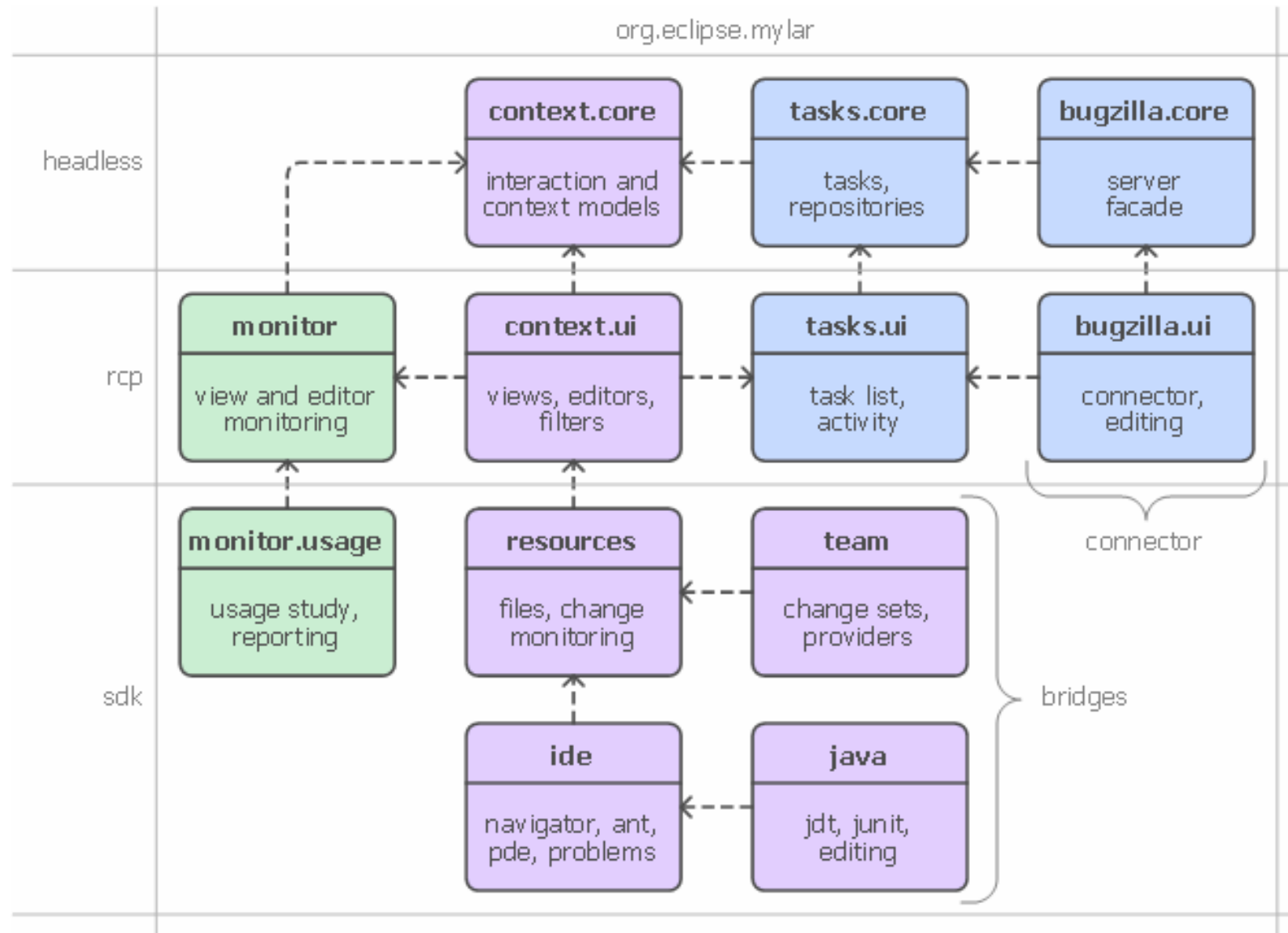
Tasks

- Core facilities: lifecycle, persistence, synchronization
- UI facilities: editing, diffs, notifications
- UI and persistence provided
- Developers access multiple issue trackers
- Provide customizable behavior with consistent integration

Context

- Generic model, context management, persistence
- Structure bridges: map context to existing models: e.g. JavaModel
- UI bridges: selection/edit/refactoring capture, map to UI
- Model scales with interaction, not with workspace size
- Facilitated by Eclipse's modularity and component model

Extensibility



Provisional APIs

Tasks API

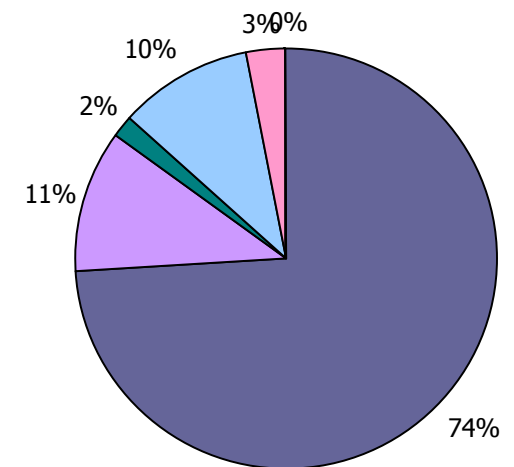
- Currently supports three connectors: Bugzilla, JIRA and Trac
- Extensible to other kinds of trackers and repositories

Context API

- Currently supports: Resources, JDT, PDE, Ant
- Has been stable, but revision coming

Monitor API

- Usage monitoring based on interaction history
- Studied ~100 developers voluntarily using Mylar
- Privacy, obfuscation, reporting



- Package Explorer
- Search
- Type Hierarchy
- Outline
- Call Hierarchy
- Bookmark

Questions

Status

- 1.0 planned for Dec 8th, UI streamlining and other improvements
- Thriving on community feedback on contributions

Questions?

Mylar

Reducing information overload one task at a time

Committers

- Mik Kersten, Rob Elves, Steffen Pingel, Ian Bull, Gail Murphy

Contributors

- Eugene Kuleshov, Jeff Pound, Brock Janiczak, Willian Mitsuda, Nathan Hapke, Raphael Ackermann, Gunnar Wagenknecht, Shawn Minto, Ken Sueda, Wesley Coelho, Leah Findlater

More info

- <http://eclipse.org/mylar/publications.php>