Eclipse Project 3.2 Release Review

Eclipse Project PMC

3.2 Highlights

- Logical model integration
- Runtime refactored
- Standalone OSGi implementation and community
- Improved and extended SWT widget set
- Enhanced internationalization support (ICU4J)
- JUnit 4
- Java SE 6.0 compliance
- Improved Update performance
- More customizable UI
- Universal Welcome
- Improved Java and PDE tooling
- Enhanced debug platform and text editor
- Many other enhancements

Themes and Plan Items

- Scaling up
  - Provide more support for large scale workspaces
  - New source lookup for debugging Eclipse applications
  - Split refactoring
  - Improve NLS tooling
- Enterprise Ready
  - Flexible workspaces
  - Update Enhancements
  - Improve PDE build
  - Deliver support for ICU4J with the Eclipse Platform
  - Enable compiler participation
- Design for Extensibility: Be a Better Platform
  - Logical model integration
  - Improve/extend SWT widget set
  - Enhance the text editor
  - Enhance the debug platform
  - Help System enhancements
  - Support dynamic and reusable content in User Assistance components
  - Create Universal Welcome
  - Address new window system capabilities
  - Support arbitrary Java content types

Themes and Plan Items (cont'd)

- Simple to Use
  - Provide more customizable UI
  - Capabilities/Perspectives/Components
  - Improve cheat sheet support
  - Improve NLS tooling
  - Improve target support
  - Improve task assistance in text fields
  - Improve user experience in presence of syntax errors
  - More refactorings
  - Support for execution environments
  - Code style clean ups
  - Improve JUnit support
- Appealing to the Broader Community
  - Improve and extend the SWT widget set
  - Address new window system capabilities
  - Add support for Java SE 6 features
  - More static analysis
  - Enhance Java infrastructure
- Rich Client Platform
  - Give OSGi a first class presence on eclipse.org
  - Refactor the runtime

New and Noteworthy - Platform

- Integrated progress in splash screen on startup
- Problems, Tasks and Bookmarks view supports multiple filters
- Marker Limits moved to preferences
- Categorization added to Import and Export wizards
- Improved hyperlink navigation
- Window-level working set selection
- Filtering in Show view dialog and New, Import, and Export wizards
- Import Existing Projects copying contents into workspace
- Draggable window trim
- Problems view filters use Window Working Set by default
- Linked resources at any depth within a project
- Install/Update enhancements
- Mac OS X on X86/Intel
- Alternative file systems
- Multiple problems can be fixed at once
- Close unrelated projects menu command
- Improved help in dialogs (using tray)
- Direct text search action
- Multiple search views
- Open cheat sheet from a content file
- New look for welcome
- Live news in the Eclipse SDK Welcome
- Animated message area for displaying errors and warnings
- System Default theme
- Import Team Project Set improvements
- Commit comment templates
- Improved patching support
- Improved conflict handling for CVS Team>Update operation
- CVS shows model content in synchronizations
- Expand All in CVS Repositories view
- History view improvements
- Local and remote history operations combined for CVS projects
- Date categories in CVS history
- Proxy support for CVS pserver connections
- CVS Quick Diff annotations
- Ant launch configuration refactoring
- Java problem markers from a "javac" build
- Breakpoint import and export
- Launch dialog enhancements
- Filter launch configurations
- Centralized Run/Debug perspective settings
- Display debug variables using columns or tree

New and Noteworthy (cont'd)

- Tooltips for annotations in Text editors
- Annotation navigation in Text editors
- JFace field assistance
- Support for dynamic Help content
- User assistance search enhancements
- Tabbed properties framework
- Problems can be grouped
- Embed command links in User Assistance content
- Support for launching commands from cheat sheets
- Completion message in cheat sheets
- Composite cheat sheets
- Help keyword index
- XHTML help docs conversion tool
- Universal welcome
- Support for configuration and themes in Welcome
- Namespace support for the Ant extension points
- Sort indicator in tables and trees
- Reorderable columns in trees
- Improved XP theme support
- Animated GIFs
- Desktop appearance notification
- Dispose notification
- Tool tips for columns
- Vertical CoolBar
- OpenGL support
- Virtual tree (style)
- Buttons with image and text
- HIView on Mac OS X
- Embedding objects in text
- Indent, align and justify text
- Text baseline rise
- HSB color support
- StyledText alignment, indentation, justification
- Embedded objects in StyledText
- Background image
- Native image loading
- Dragging text
- New ExpandBar widget
- Bullets in StyledText
- Custom draw Tree and Table
- Drag over effects for Table and Tree
- New Tooltip class

New and Noteworthy - JDT

- Create and apply refactoring scripts
- JAR file export with refactorings
- Refactoring History
- JUnit 4 support
- JUnit view history
- Clean Up Wizard to help fix multiple problems and establish a code style.
- Extract superclass refactoring
- Introduce indirection refactoring
- Rename Type updates similarly named elements
- Category support in Javadoc comments (@category tag)
- API aware refactorings
- Rename Package refactoring renames subpackages
- Find Broken Externalized Strings
- Generate hashCode() and equals()
- "Surround With" quick menu
- Paste type declaration creates new compilation unit
- File content decorator
- Filter for refactoring preview dialog
- Command line code formatter
- Support for Java-like extensions
- New Quick Fixes
- CamelCase support in code completion
- Customizable Content Assist
- Ruler support and outline for Java files outside workspace
- All members can be folded (collapsed) in the Java editor
- Improved support for Eclipse's string externalization mechanism
- Javadoc view and tool tips from attached Javadoc
- Completion in Javadoc comments
- Java SE 6.0 compliance
- Null reference analysis by Java compiler
- Improved syntax recovery
- Treat configurable errors as non fatal
- Detection of unnecessary $NON-NLS$ tags
- Detection of raw type usage
- Detection of method parameter assignments
- Detection of unused labels
- Detection of switch case fall through
- Improved performance for large .jar files
- Recreating modified .class files
- Execution environments preference page
- System property launch variable
- Evaluation support for arrays in debugger
- Show Java thread groups in debugger
- Runtime classpath exported entries
- Suspend Thread versus Suspend VM

New and Noteworthy - PDE

- A target can be defined in a .target file
- Targets can be contributed to an Eclipse product via the **org.eclipse.pde.core.targets** extension point
- Hierarchical view of plug-inson the Plug-in Development > Target Platform preference page
- Plug-ins for any OSGi framework
- Equinox OSGi framework launcher
- Java search hits in manifest files
- Plug-in manifest files participate in refactoring
- NLS wizard for plug-in manifest files
- Organize plug-in manifest files
- New processing instruction in plugin.xml files
- Bundle execution environment
- Automated management of dependencies
- Structural compare and syntax highlighting for manifest.mf files
- Validate build.properties files
- Quick fixes for plug-in manifest files
- Automatic Javadoc attachment to libraries on plug-in build path
- New extension point schema editor
- Headless RCP application template
- Form validation in product editor
- Integrated progress monitor in product splash screen
- Platform-specific launcher arguments for cross-platform product export
- Add a Welcome page to your product
- Sharable and portable PDE launch configurations.
- Templates for launching arguments
- Enhanced and automatic plug-in validation prior to launching
- New source lookup for debugging Eclipse applications
- Plug-in level custom Ant targets
- Building products from a .product file in a headless automated build
- Multiple repository support (use the org.eclipse.pde.build.fetchFactories extension point)

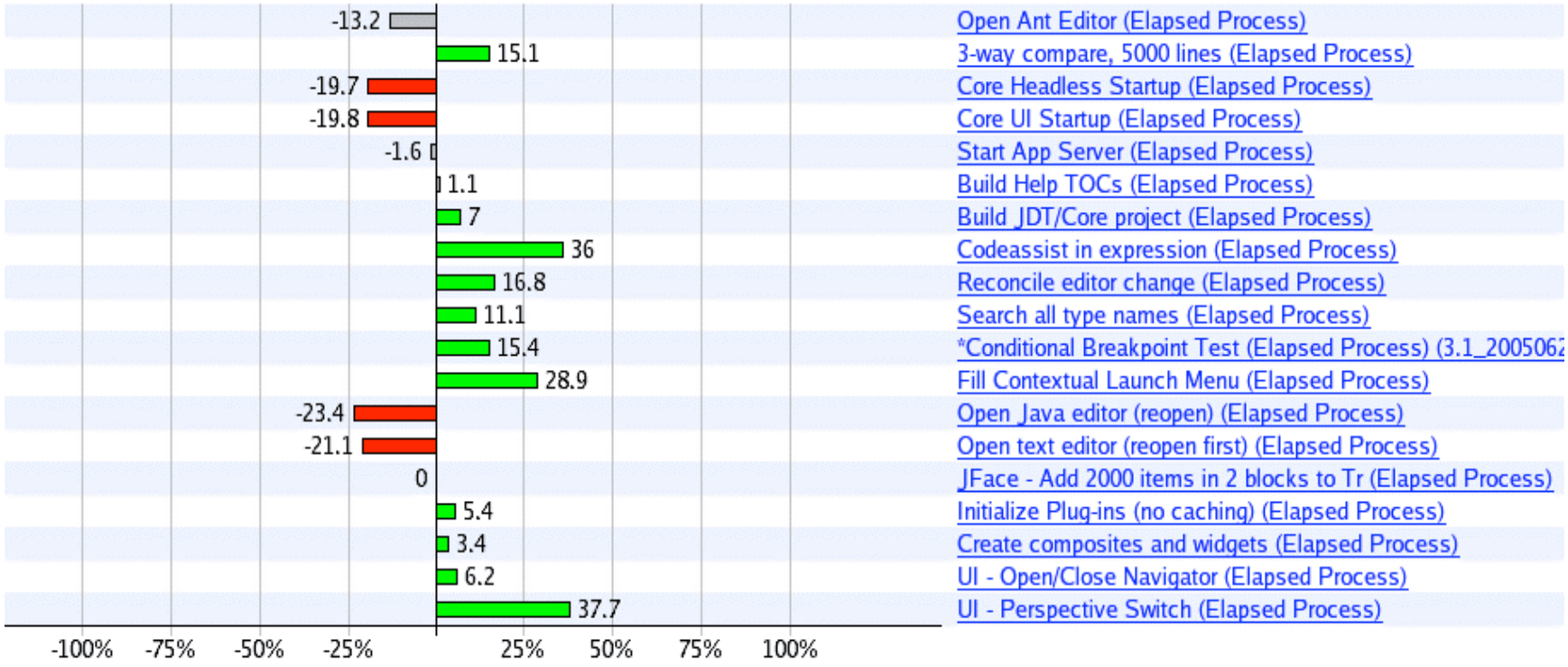
New and Noteworthy - Equinox

- Equinox is now a subproject of Eclipse.
- Equinox is the OSGi R4 specification reference implementation.
- Refactored runtime: the **org.eclipse.core.runtime** plug-in was refactored into several smaller plug-ins so clients depend on only the function they require.
- Extension registry can now be used outside of Eclipse and OSGi.
- The extension registry implementation is now pluggable.
- Clients are now allowed to make dynamic contributions to the extension registry.
- Decoupling of the plug-in name and the definition of extensions and extension points.
- Put commonly useful classes (e.g. Assert, ListenerList) into a common bundle for other bundles to require.
- Most classes in the common bundle can be used with or without OSGi running.
- In Eclipse 3.2 a new hookable adaptor has been included that is used by default as the framework adaptor. A new implementation of the adaptor API is now included which provides hooks that others can implement to provide additional functionality to the framework adaptor implementation.

Deferred 3.2 Plan Items

- Embedding editors and views
- Improve UI forms
- Provide a single user-assistance content delivery mechanism
- Improve serviceability
- Support GB-18030-2
- Increase flexibility when building RCP applications
- Debug platform enhancements
- Provide pervasive user-assistance search capabilities
- Implement library projects
- API aware tools

Performance of 3.2RC5 Relative to 3.1



3.2 Plug-in Changes from 3.1

Added Plug-ins

- com.ibm.icu
- com.jcraft.jsch
- org.eclipse.core.contenttype
- org.eclipse.core.filesystem
- org.eclipse.core.jobs
- org.eclipse.core.runtime.compatibility.auth
- org.eclipse.equinox.common
- org.eclipse.equinox.preferences
- org.eclipse.equinox.registry
- org.eclipse.jdt.apt.core
- org.eclipse.jdt.apt.ui
- org.eclipse.jdt.core.manipulation
- org.eclipse.jdt.junit4.runtime
- org.eclipse.jface.databinding
- org.eclipse.ui.intro.universal
- org.eclipse.ui.navigator
- org.eclipse.ui.navigator.resources
- org.eclipse.ui.views.properties.tabbed
- org.junit4

Removed Plug-ins

- none

Unchanged Plug-ins

- org.apache.ant
- org.junit_3.8.1
- org.eclipse.core.boot

Non-Code Aspects

- The 3.2 release will contain updated User and ISV documentation
- Books authored by members of the Eclipse community:
  - Eclipse Rich Client Platform: Designing, Coding, and Packaging Java Applications by Jeff McAffer and Jean-Michel Lemieux
- Community is very active
  - Mailing lists and newsgroups have steady activity
  - Blogs dedicated to Eclipse are active e.g.
    - http://www.planeteclipse.org
  - Wiki content is growing
    - http://wiki.eclipse.org/index.php/Eclipse_Project

Non-Code Aspects

- Internationalization
  - Latin1 and Latin2 locales are supported in all operating environments
  - DBCS locales are supported on Windows, GTK, and Motif window systems
  - BIDI locales supported on Windows
  - GB18030-1 Chinese codepage standard is supported on Windows and GTK
- Localization
  - The Eclipse project is fully localized and externalized
  - Updated message catalogs will be provided for 3.2
- Accessibility
  - We are unaware of any non-compliance with section 508 accessibility standards in the user interface

Non-Code Aspects

- Articles, examples, and tutorials
  - New and updated articles and tutorials on eclipse.org (about 14)
  - Some of the new/updated articles and tutorials were provided by the Eclipse community
  - Some older articles need to be reviewed and updated for 3.2, if applicable

Platform Quality API

- API quality is a collaborative effort that involves the experience of the developers working on the Eclipse project, and feedback from consumers.
- API changes and proposed API additions are often broadcast to mailing lists to raise awareness of the changes and encourage discussion and feedback.
- The 3.2 migration guide identifies 5 changes:
  - For each, a description of the change, what code is affected, and the action that needs to be taken is described.
  - API changes between 3.1 and 3.2 are checked by component teams and checked again during API reviews, using an automated diff tool.
  - We are not aware of any API compliant plug-ins breaking as a result of these changes.
- The 3.2 migration guide also describes changes required to adopt mechanisms and APIs that are new in 3.2.
- The PMC is comfortable supporting the API that is in the Eclipse project 3.2

3.2 APIs

New API

Platform

- Support for logical model integration (merge, decorators, history, etc)
- Expansion of commands API
- Extended File System (EFS) support for flexible workspaces (in org.eclipse.core.filesystem)
- Field assist support for wizards and dialogs
- Common Navigator view
- Tabbed Properties view
- Additional marker support
- Support for adding trim elements to the Workbench
- Added UI elements (new menu items, filtered tree) to support large scale workspaces
- Added namespace support for Ant extension points
- New shared document undo manager, quick assist, and revision information display
- Allow launch configurations to be associated with workspace resources
- Custom drawing of TableItems and TreeItems
- Tree improvements to match Table feature set
- StyledText support for embedding of objects and additional text layout options
- New GLCanvas for drawing in SWT via OpenGL
- Support for setting background images on Controls
- New command line arguments to run update site performance enhancement tool standalone
- New attributes added to site.xml DTD to support update site performance enhancement tool
- New parameters to allow server side mirror sorting by country with option to auto-select best mirror

Platform (cont'd)

- Universal welcome API added (including theme and generic welcome API enhancements
- New markup annotations for dynamic content
- Markup and extension point added for defining and contributing keyword index for content help
- Tray API added to JFace dialog
- Command serialization syntax and enablement for embedding in documents
- Search participant API for searching any content type
- Interfaces added to provide for custom handling of open action and image for search results
- Composite cheat sheets
- Added elements to allow commands to be launched from or a completion message to be added to a step in a cheat sheet

Equinox

- The org.eclipse.core.runtime plug-in was re-factored into several smaller plug-ins for clients to depend on only the function that is required
- Decoupling of the plug-in name and the definition of extensions and extension points
- The extension registry implementation is now pluggable
- Clients are now allowed to make dynamic contributions to the extension registry

PDE

- APIs for PDE launch configurations (Eclipse application and Equinox framework launch configurations)
- Launcher tabs and delegates are API
- End-to-end support to automate headless building of RCP applications
- New API to plug repositories into the headless build process

3.2 APIs (cont'd)

JDT

- New extension points for completion proposal computers and sorters
- Compilation participant API added to enable contributing to reconcile and build operation of Java files
- DOM AST now provides bindings for all annotation references
- New preferences for new compiler diagnostics
- JavaModel now allows defining working copies for class files and exposes method/field categories from Javadoc
- Classpath entries can be marked as optional
- Access restrictions are now inherited and can optionally be ignored if a better match is found downstream on classpath
- Camel case syntax is leveraged into code assist and search engine
- Support for Java SE 6 compliance
- New API for performance (ExpressionInfo)
- New plug-in jdt.core.manipulation contains advanced JDT core APIs for re-factoring participants
- Added API in the JUnit wizards for JUnit 4
- New API for re-factoring history and scripting support
- New API for model integration of re-factorings
- New API for element mapping
- New API for re-factoring preview filtering and participant filtering support
- New API for operation validation

Deprecated

- Content assist for widgets (replaced by field assist)
- File buffer extension point to specify document creation
- Javadoc completion processor
- objectClass attribute of org.eclipse.ui.propertyPages extension point
- Class PluginVersionIdentifier – now using OSGi's Version class
- Methods on the Platform and Plugin classes
- #getNamespace on interfaces IExtension, IExtensionPoint, IConfigurationElement
- In bundle manifest, Eclipse-AutoStart is now Eclipse-LazyStart
- IProjectDescription.getLocation() is replaced by getLocationURI()
- IResource.isLocal/setLocal()
- handle attribute of org.eclipse.ui.intro.config extension point
- DOM CompilationUnit#lineNumber(int) is replaced by #getLineNumber(int)

Tool Usability

- Eclipse is a superior IDE for Java tooling and plug-in development
- Many usability enhancements made in 3.2 to continue this tradition
  - More tightly integrated help display, using tray
  - String externalizing (NLS), code clean-up
  - Help keyword index
  - Launch templates
  - Hierarchical view of plug-ins
  - New welcome theme with live news and universal welcome
  - Improved NLS tooling
  - Extension point schema editor
  - Code clean up wizard
  - Multi-quick fix
  - Refactoring scripts
  - Much more…

Tool Usability (cont'd)

Awards

- **2005 SOA Web Services Journal Readers' Choice Awards**
  - Eclipse awarded Best GUI for SOA
- **2005 Java Developer's Journal Readers' Choice Awards**
  - Eclipse awarded Best Java Application
  - Eclipse Rich Client Platform awarded Best Rich Client Platform
  - Eclipse IDE awarded Best Team Development Tool
  - Eclipse IDE awarded Most Innovative Java Product
  - SWT awarded Best Java Class Library
  - Eclipse awarded Best Java Debugging Tool

Architectural Issues

- Pack200 used in Update
  - Jar compression reduces Eclipse delivery footprint
- Metadata digest used in Update
  - Eliminate connections for retrieving feature metadata from update sites
- Equinox is a standalone OSGi implementation
- Runtime refactored
- Unused or under-utilized new features in the Platform
  - JFace data binding, Tabbed properties, Common navigator
- ICU4J integrated as a plug-in and adopted by the Platform
  - Increased footprint by 3MB, added dependencies
  - Replacement plug-in available (190KB with source, 60 KB without)

End of Life Issues

- When evolving API the Eclipse Platform will, whenever possible, deprecate the affected API methods and continue to keep them operational.
- Exceptions to this rule are in the 3.2 migration guide.

Bugzilla

- Between June 1, 2005 and May 29, 2006
  - More than 19,900 reports were created
  - Over 13,800 were resolved
  - Over 6,900 were resolved without changing code (invalid, duplicate, etc)
- Current state is
  - 11 blockers, 80 critical
  - 0 P1, 158 P2
- 3.1 state was
  - 5 blockers, 50 critical
  - 0 P1, 138 P2

Standards

- OSGi Service Platform Core Specification, Release 4
- Elements of the OSGi Service Platform Service Compendium, Release 4
- APT mirror API (com.sun.mirror 1.5)
- User Assistance consumes (parses) a small subset of RSS 1.0 to get news from eclipse.org
- J2SE
  - Tools are build against J2SE 1.4
  - Compiler can generate 1.3, 1.4, 5.0, and 6.0 code
  - Clients can run 1.4 or 1.5

UI Usability

- Strings are externalized to support translation into other languages.
- Extensive use of mnemonics and shortcut keys in the user interface enhances usability.
- Bidirectional support on Windows.
- We are unaware of any non-compliance with accessibility standards in the user interface.

Schedule

- Milestones every 6 weeks, 6 cycle duration
  - Feature and API frozen on March 31, end of M6 cycle
- Tracked schedule
  - All milestones except M5 delivered as promised
    - M5a was produced to ease transition of clients who were broken as a result of not following API specifications for extension ids when namespace extension support was introduced.
- End game (release candidate) milestones for 7 cycles
  - Duration reduced from 2-week to 1-week cycles at RC3 milestone
  - No new features or API allowed without proper approvals
  - Development to end on June 2, 2006
  - Increasingly stringent approval, checking, and change notification requirements in this stage

Process

- The Eclipse project is developed using an open, transparent, and inclusive process
- Teams rely on Bugzilla, mailing lists and newsgroups for input
- Weekly planning calls conducted with the PMC and component leads
  - Meeting minutes posted to the eclipse-dev mailing list
- Component teams have publicly available milestone plans
  - Use project's web space on eclipse.org to broadcast component milestone plan items and provide status on each item, per milestone

Community

- Eclipse team members are active in Bugzilla, newsgroups, and mailing lists
- Blogs started by Eclipse committers are evolving
  - http://www.planeteclipse.org
- Some teams are using the eclipse-dev IRC channel
  - irc.freenode.net#eclipse-dev
- The Eclipse team participates in code camps, conference presentations, and tutorials, including
  - EclipseCon, JavaOne, JavaWorld, JAOO
- The Eclipse team interacts with other open source projects, standards bodies, and other projects on eclipse.org, including
  - OSGi, Apache Ant, JLS, WTP
- Million download challenge for Eclipse 3.1

IP Issues

- All significant and third party contributions have been reviewed and approved by Eclipse legal.
- About files and license files are complete and correct.
- Project log complete, yet to be reviewed by Eclipse legal.
  - http://www.eclipse.org/eclipse/development/eclipse_project_log.html

Project Plan for Eclipse 3.3

- Pending - still in planning stage

Release Review

Version 032