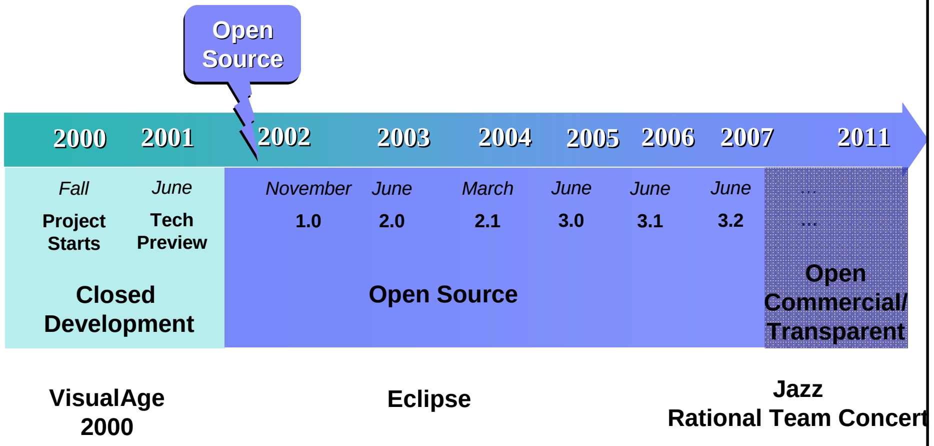
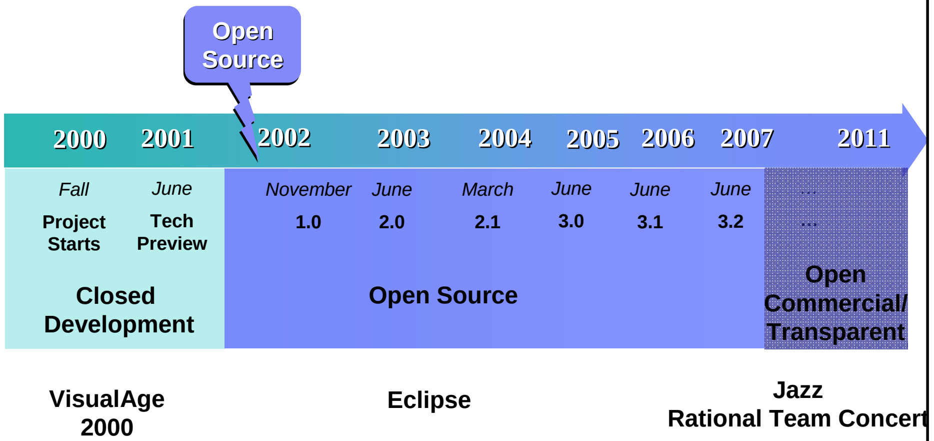
From Closed to Open to Transparent Software Development

Dr. Erich Gamma
IBM Distinguished Engineer
IBM Rational Zurich Research Lab

Eclipse Timeline

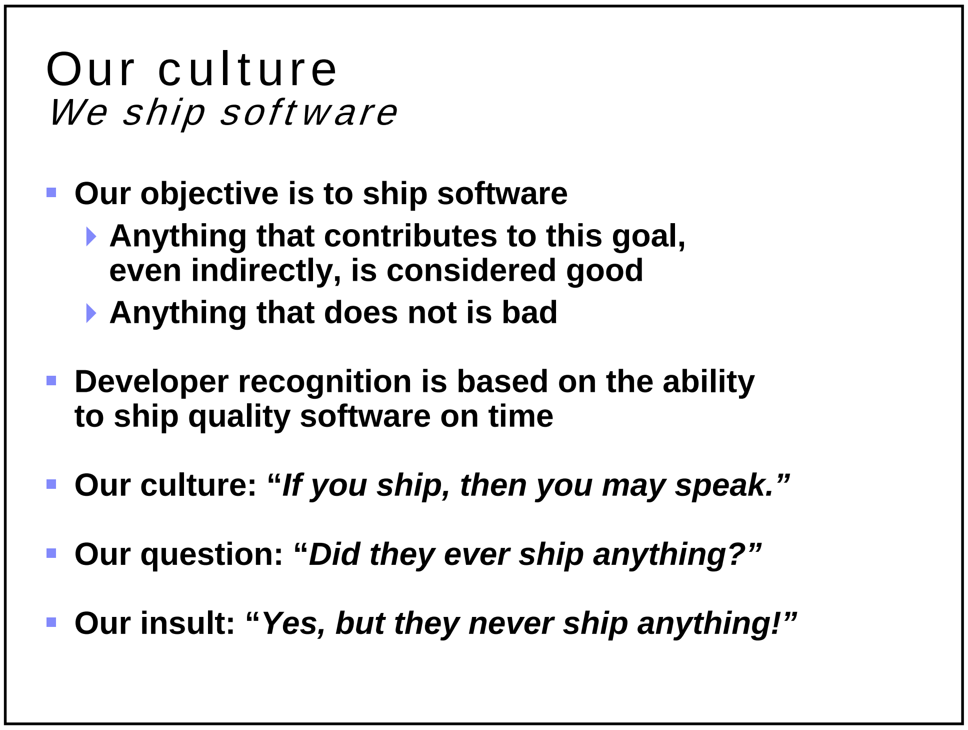


Eclipse Timeline



How we Started: Closed development

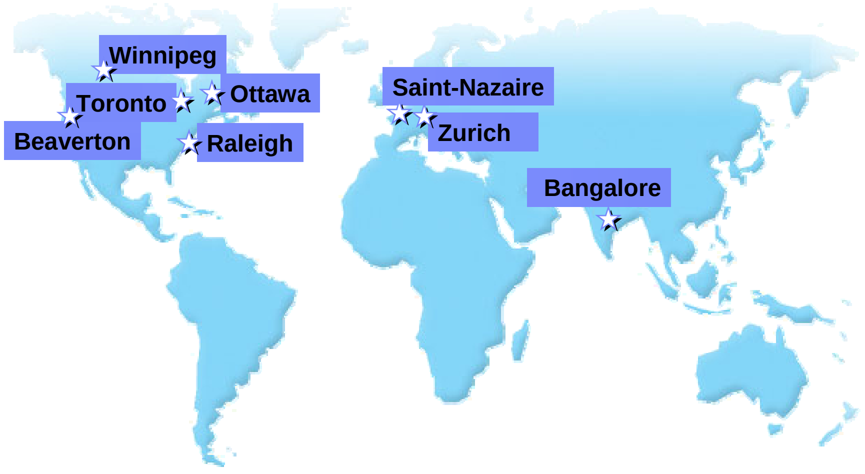
- The Swiss Bank approach to software development
  - If it hasn't shipped it doesn’t exist
- Strong firewall between developers and customers



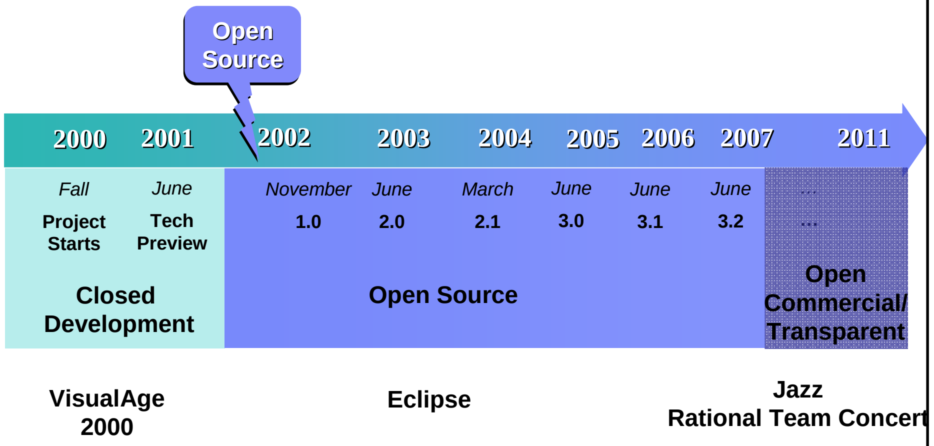
Our culture
We ship software

- **Our objective is to ship software**
  - **Anything that contributes to this goal, even indirectly, is considered good**
  - **Anything that does not is bad**
- **Developer recognition is based on the ability to ship quality software on time**
- **Our culture: “*If you ship, then you may speak.*”**
- **Our question: “*Did they ever ship anything?*”**
- **Our insult: “*Yes, but they never ship anything!*”**

Our Team
Globally Distributed



Eclipse Timeline



Eclipse vision

2001

Industry open source tool platform

Seamless integration of previously unintegrated tools

A plug-in for everything

Users can assemble tools from different suppliers to make a tool environment the way *they* want it.

Why did you open source it?

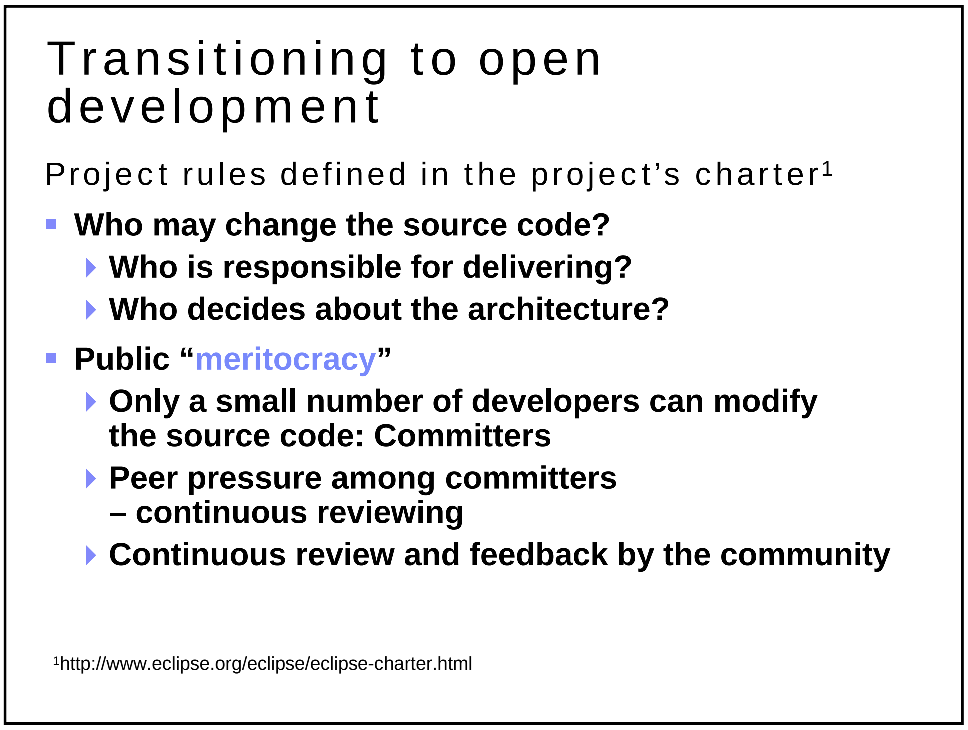
TOP 5 REASONS

5. No one company can deliver all the tool function that customers need
4. Customers don't want to be locked in to one proprietary technology
3. Tool builders are reluctant to invest in building tools for a proprietary technology they have no control over
2. Open source is a great way to reach developers directly
1. The large blue company realized the only way eclipse would reach its full potential as an industry platform was if the tool builders and users had the opportunity to influence and contribute

Conclusion: Release eclipse under an open source license and establish an open source project as a community focal point to continue to evolve the technology.

November 2001
Reaction from the development team

You want us to do what?
Why are we doing this again?
Code in public?
Answer all those dumb questions?
Have technical discussions in public?

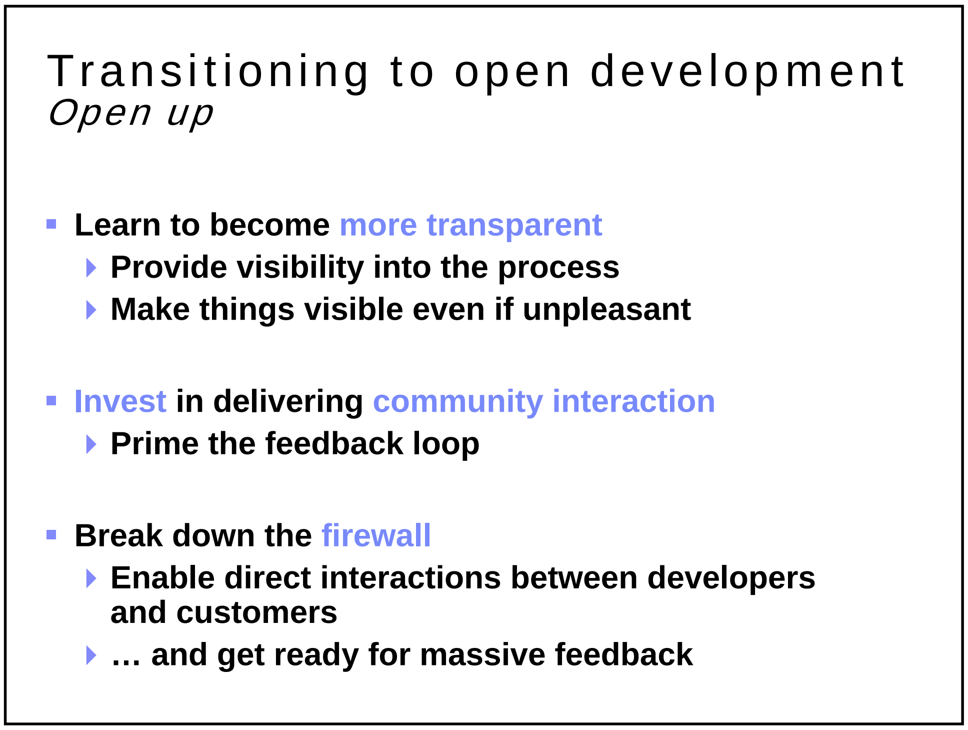


Transitioning to open development

Project rules defined in the project's charter[1]

- Who may change the source code?
  - Who is responsible for delivering?
  - Who decides about the architecture?
- Public "meritocracy"
  - Only a small number of developers can modify the source code: Committers
  - Peer pressure among committers – continuous reviewing
  - Continuous review and feedback by the community

[1]http://www.eclipse.org/eclipse/eclipse-charter.html



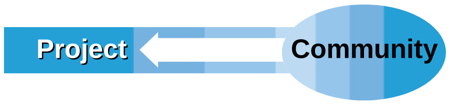
Transitioning to open development

Open up

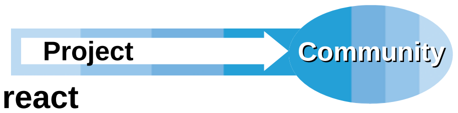
- Learn to become more transparent
  - Provide visibility into the process
  - Make things visible even if unpleasant
- Invest in delivering community interaction
  - Prime the feedback loop
- Break down the firewall
  - Enable direct interactions between developers and customers
  - … and get ready for massive feedback

Open development

Having a community is cool!



- Community gives project
  - Answer user questions
  - Report defects and request features
  - Validate technology by extension
  - Validate technology in a new configuration
  - Submit patches and enhancements

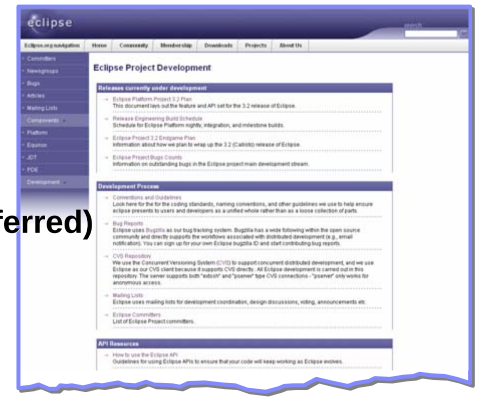


- Project gives community
  - Listen to feedback and react
  - Demonstrate continuous progress
  - Transparent development

Open development

Interacting with the community

- What is going on in the project?
  - All discussion in defects/work items
  - Mailing lists (but work items are preferred)
- How can I use the project to do XYZ?
  - Newsgroups
  - A wiki
- Where do I start?
  - Project portal



Eclipse Project Development

Releases currently under development

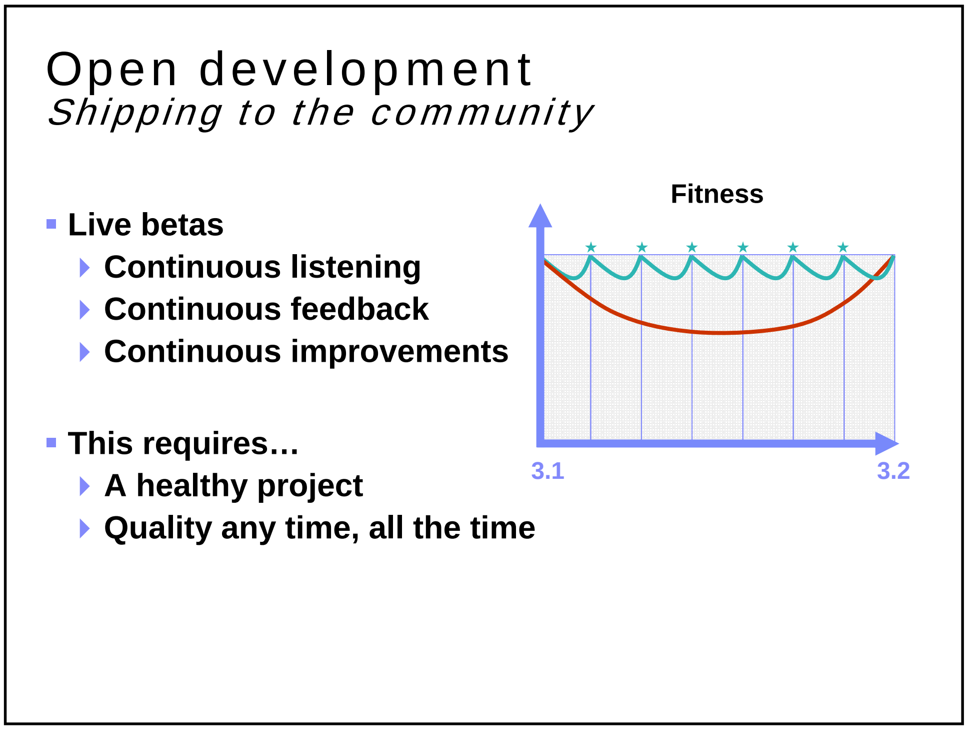
- Eclipse Platform Project 3.2 Plan
  This document lays out the feature and API set for the 3.2 release of Eclipse.
- Release Engineering Build Schedule
  Schedule for Eclipse Platform nightly, integration, and milestone builds.
- Eclipse Project 3.2 Endgame Plan
  Information about how we plan to wrap up the 3.2 (Callisto) release of Eclipse.
- Eclipse Project Bugs Counts
  Information on outstanding bugs in the Eclipse project main development stream.

Development Process

- Conventions and Guidelines
  Look here for the for the coding standards, naming conventions, and other guidelines we use to help ensure eclipse presents to users and developers as a unified whole rather than as a loose collection of parts.
- Bug Reports
  Eclipse uses Bugzilla as our bug tracking system. Bugzilla has a wide following within the open source community and directly supports the workflows associated with distributed development (e.g., email notification). You can sign up for your own Eclipse bugzilla ID and start contributing bug reports.
- CVS Repository
  We use the Concurrent Versioning System (CVS) to support concurrent distributed development, and we use Eclipse as our CVS client because it supports CVS directly. All Eclipse development is carried out in this repository. The server supports both "extssh" and "pserver" type CVS connections - "pserver" only works for anonymous access.
- Mailing Lists
  Eclipse uses mailing lists for development coordination, design discussions, voting, announcements etc.
- Eclipse Committers
  List of Eclipse Project committers.

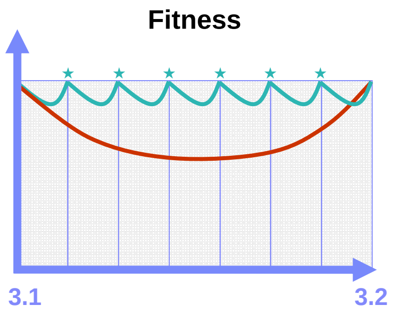
API Resources

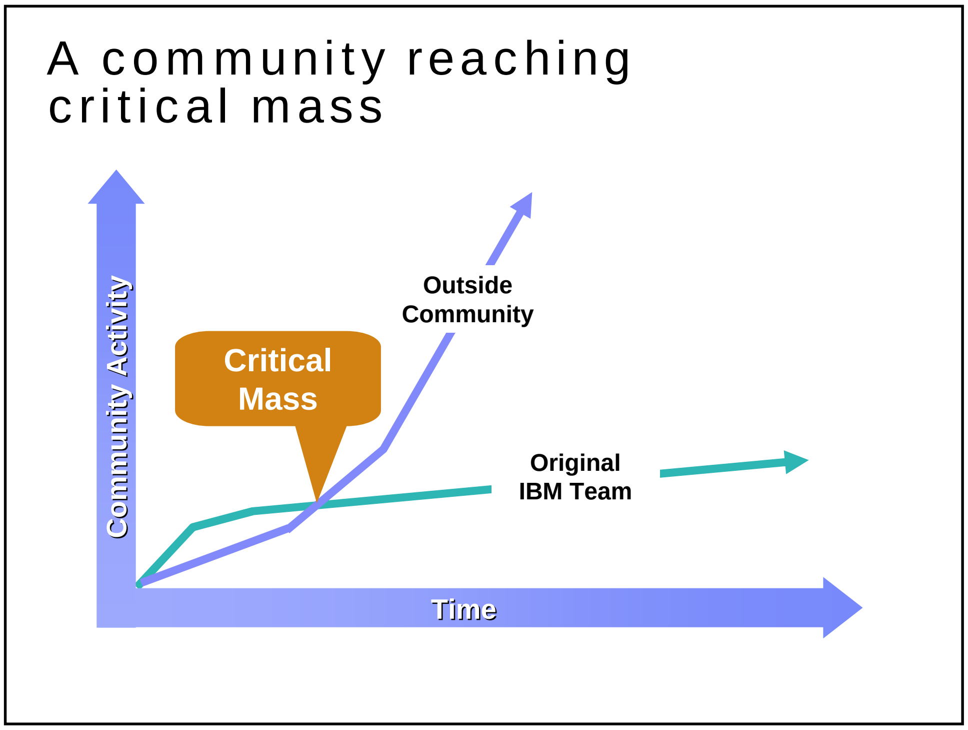
- How to use the Eclipse API
  Guidelines for using Eclipse APIs to ensure that your code will keep working as Eclipse evolves.



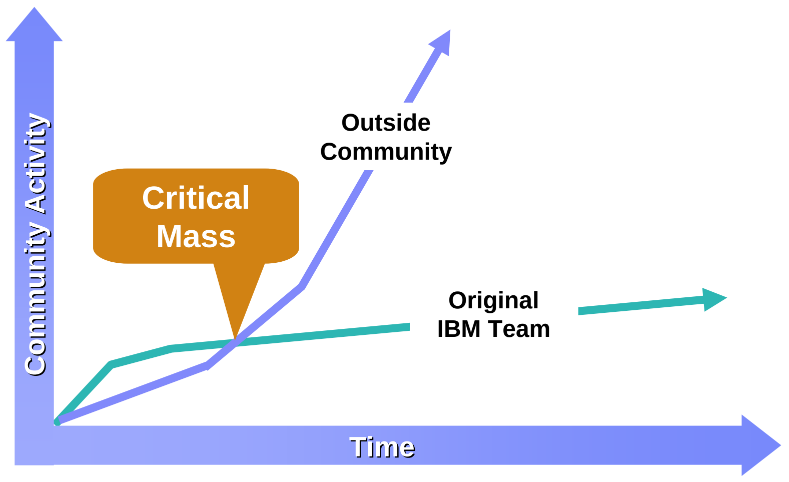
Open development
Shipping to the community

- Live betas
  - Continuous listening
  - Continuous feedback
  - Continuous improvements
- This requires…
  - A healthy project
  - Quality any time, all the time

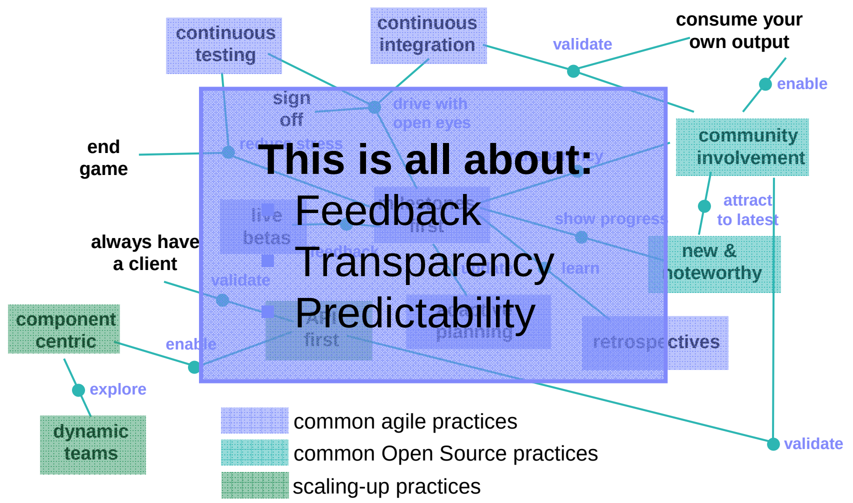




A community reaching critical mass



The Eclipse Development Practices



▨ common agile practices

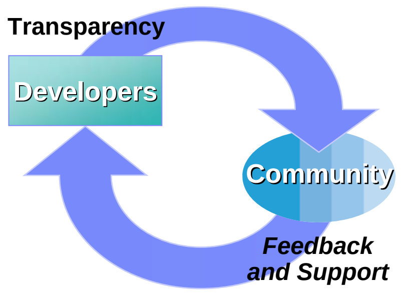
▨ common Open Source practices

▨ scaling-up practices

Lessons learned

Transparency and predictability enable feedback

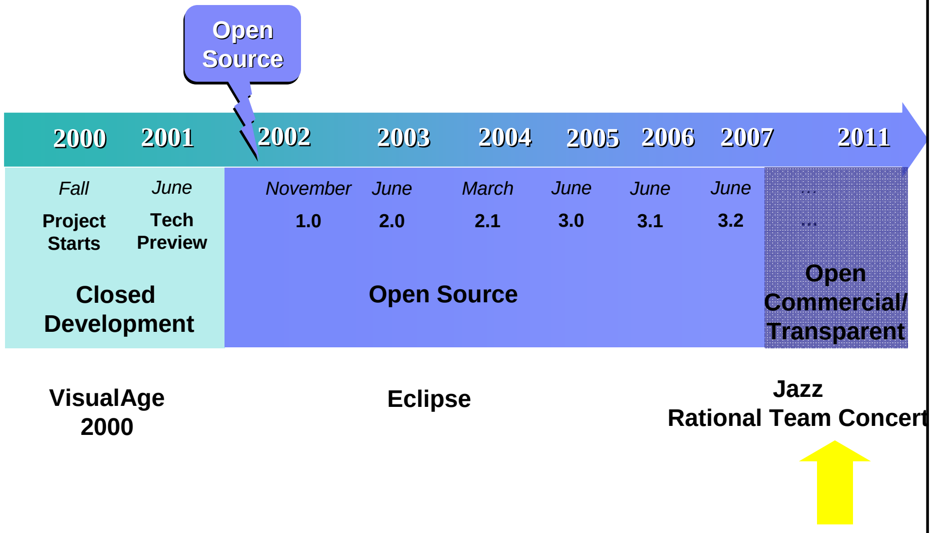
- **Transparency helps existing development**
  - **Better understanding of current status**
  - **Responding to feedback takes time, but pays off**
- **Use same communication channels inside as outside**
  - **Helped communication in our globally distributed team**



Lessons learned
The "village effect"

- A large organization can act like a smaller organization
  - Flat hierarchies
- Visible accountability
- Communication flows, plans, and progress are visible for all to see

Eclipse Timeline



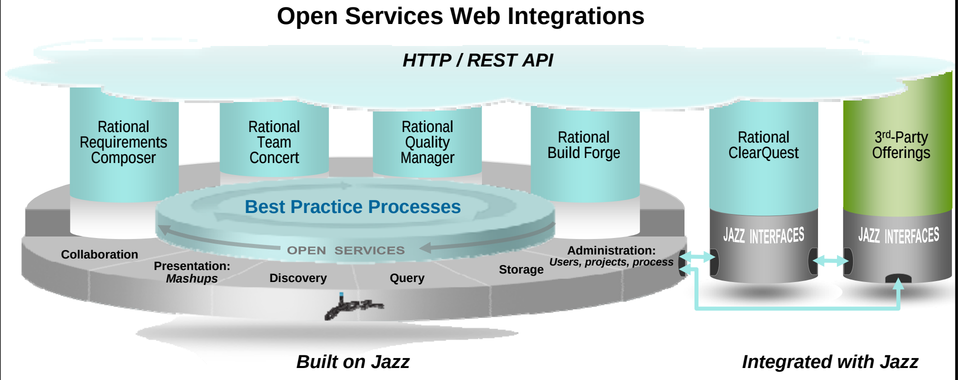
Reflections from an Eclipse PMC Lead

Many Eclipse observers do not realize the amount of work/stress/human factor involved in shipping on time every time.

It reminds me of the duck quietly advancing on the pond, but no one realizes that it is paddling like crazy underwater.

Philippe Mulet – former Eclipse PMC Lead

Jazz: Open, extensible, web-centric, integration architecture



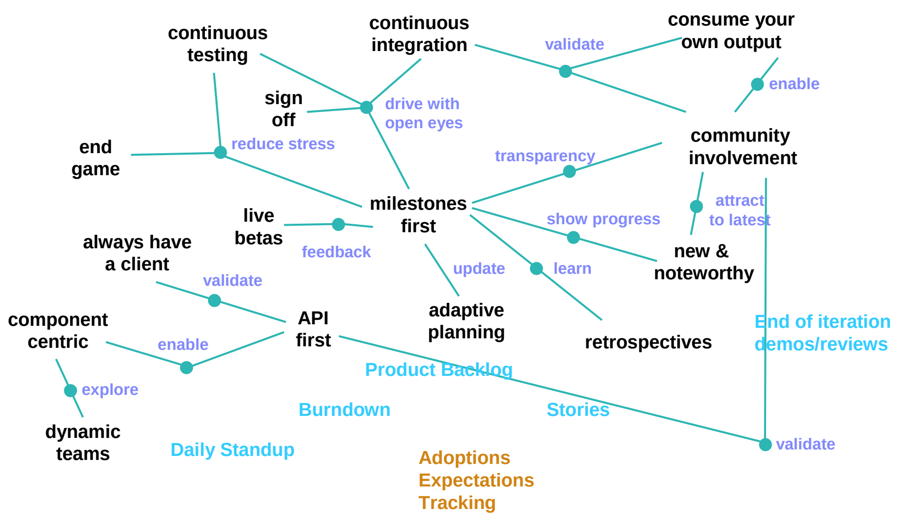
Open Commercial Development

- Products use a commercial license but development is done in the open
- Open, transparent process, from feature requests and planning through delivery
- Same communication channels for inside and outside of the projects
- Direct access to developers

Open Commercial Development

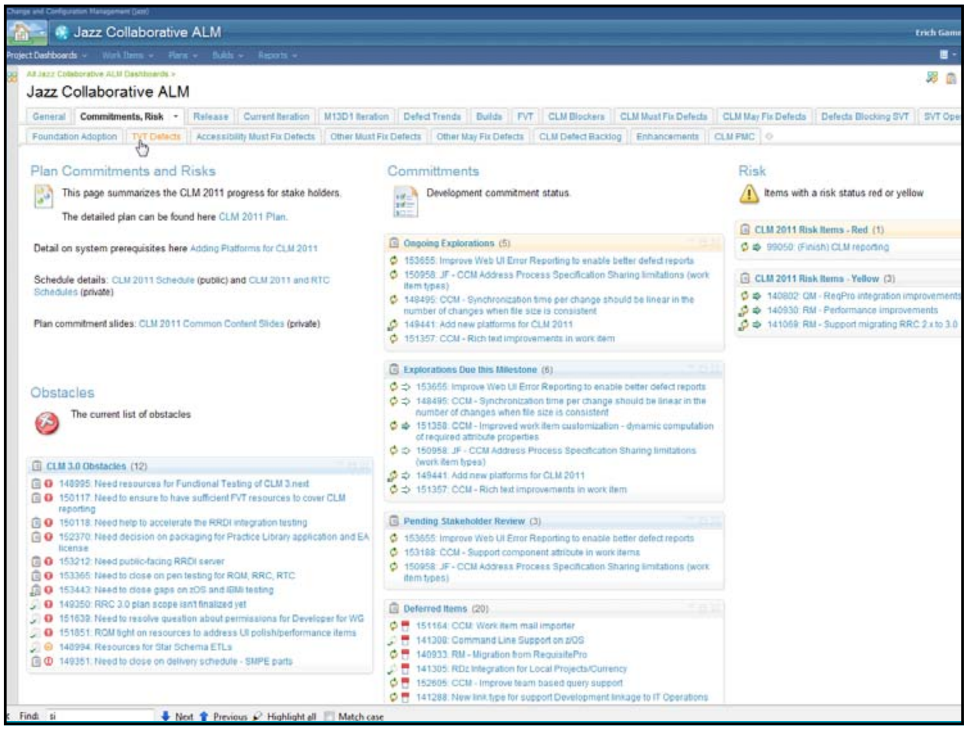
- What can the community members do:
  - Download milestones, try them, and provide feedback on betas and incubators, including source code
  - Access, create, and comment in defects, enhancement requests
  - Access milestone and component iteration plans
  - Access the development wiki
  - Participate in discussions on the development community newsgroups

Our Expanded Practices Today



Demo Transparency

www.jazz.net



Jazz Collaborative ALM

Project Dashboards · Work Items · Plans · Builds · Reports

All Jazz Collaborative ALM Dashboards >

Jazz Collaborative ALM

General | Commitments, Risk | Release | Current Iteration | M13D1 Iteration | Defect Trends | Builds | FVT | CLM Blockers | CLM Must Fix Defects | CLM May Fix Defects | Defects Blocking SVT | SVT Ope
Foundation Adoption | TVT Defects | Accessibility Must Fix Defects | Other Must Fix Defects | Other May Fix Defects | CLM Defect Backlog | Enhancements | CLM PMC

Plan Commitments and Risks

This page summarizes the CLM 2011 progress for stake holders.

The detailed plan can be found here CLM 2011 Plan.

Detail on system prerequisites here Adding Platforms for CLM 2011

Schedule details: CLM 2011 Schedule (public) and CLM 2011 and RTC Schedules (private)

Plan commitment slides: CLM 2011 Common Content Slides (private)

Obstacles

The current list of obstacles

CLM 3.0 Obstacles (12)
- 148995: Need resources for Functional Testing of CLM 3.next
- 150117: Need to ensure to have sufficient FVT resources to cover CLM reporting
- 150118: Need help to accelerate the RRDI integration testing
- 152370: Need decision on packaging for Practice Library application and EA license
- 153212: Need public-facing RRDI server
- 153365: Need to close on pen testing for RQM, RRC, RTC
- 153443: Need to close gaps on zOS and IBMi testing
- 149350: RRC 3.0 plan scope isn't finalized yet
- 151639: Need to resolve question about permissions for Developer for WG
- 151851: RQM tight on resources to address UI polish/performance items
- 148994: Resources for Star Schema ETLs
- 149351: Need to close on delivery schedule - SMPE parts

Committments

Development commitment status.

Ongoing Explorations (5)
- 153655: Improve Web UI Error Reporting to enable better defect reports
- 150958: JF - CCM Address Process Specification Sharing limitations (work item types)
- 148495: CCM - Synchronization time per change should be linear in the number of changes when file size is consistent
- 149441: Add new platforms for CLM 2011
- 151357: CCM - Rich text improvements in work item

Explorations Due this Milestone (6)
- 153655: Improve Web UI Error Reporting to enable better defect reports
- 148495: CCM - Synchronization time per change should be linear in the number of changes when file size is consistent
- 151358: CCM - Improved work item customization - dynamic computation of required attribute properties
- 150958: JF - CCM Address Process Specification Sharing limitations (work item types)
- 149441: Add new platforms for CLM 2011
- 151357: CCM - Rich text improvements in work item

Pending Stakeholder Review (3)
- 153655: Improve Web UI Error Reporting to enable better defect reports
- 153188: CCM - Support component attribute in work items
- 150958: JF - CCM Address Process Specification Sharing limitations (work item types)

Deferred Items (20)
- 151164: CCM: Work item mail importer
- 141308: Command Line Support on z/OS
- 140933: RM - Migration from RequisitePro
- 141305: RDz Integration for Local Projects/Currency
- 152605: CCM - Improve team based query support
- 141288: New link type for support Development linkage to IT Operations

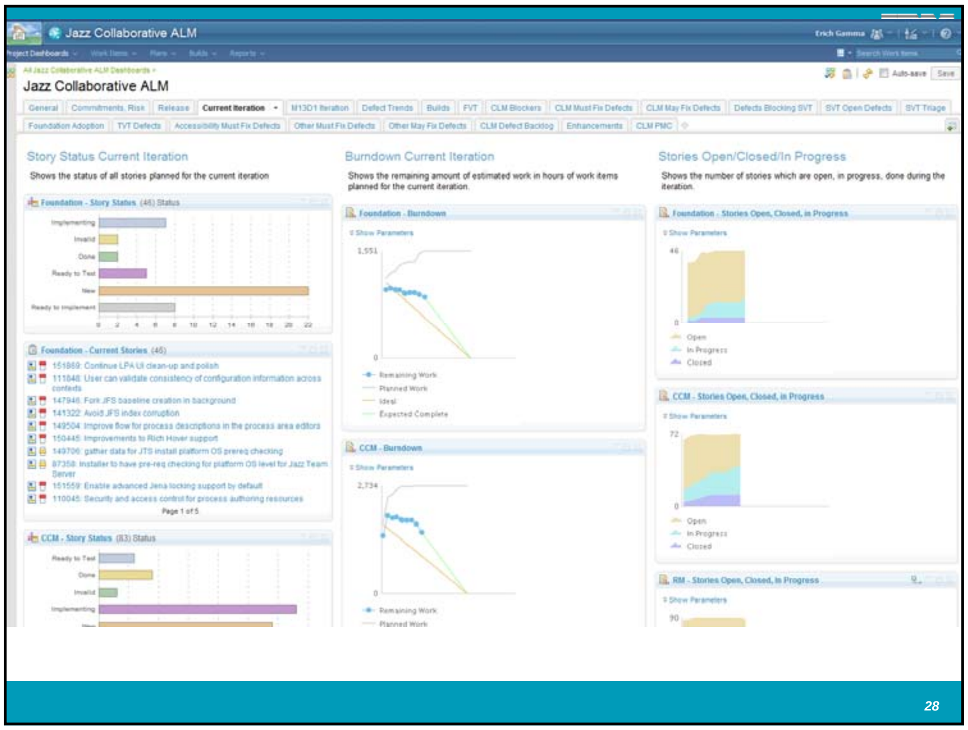
Risk

Items with a risk status red or yellow

CLM 2011 Risk Items - Red (1)
- 99050: (Finish) CLM reporting

CLM 2011 Risk Items - Yellow (3)
- 140802: QM - ReqPro integration improvements
- 140930: RM - Performance improvements
- 141069: RM - Support migrating RRC 2.x to 3.0

Find: si | Next | Previous | Highlight all | Match case



Jazz Collaborative ALM

Project Dashboards · Work Items · Plans · Builds · Reports

All Jazz Collaborative ALM Dashboards >

Jazz Collaborative ALM

Auto-save | Save

General | Commitments, Risk | Release | Current Iteration | M13D1 Iteration | Defect Trends | Builds | FVT | CLM Blockers | CLM Must Fix Defects | CLM May Fix Defects | Defects Blocking SVT | SVT Open Defects | SVT Triage
Foundation Adoption | TVT Defects | Accessibility Must Fix Defects | Other Must Fix Defects | Other May Fix Defects | CLM Defect Backlog | Enhancements | CLM PMC

Story Status Current Iteration

Shows the status of all stories planned for the current iteration

Foundation - Story Status (46) Status

Foundation - Current Stories (46)
- 151869: Continue LPA UI clean-up and polish
- 111848: User can validate consistency of configuration information across contexts
- 147946: Fork JFS baseline creation in background
- 141322: Avoid JFS index corruption
- 149504: Improve flow for process descriptions in the process area editors
- 150445: Improvements to Rich Hover support
- 149706: gather data for JTS install platform OS prereq checking
- 87358: Installer to have pre-req checking for platform OS level for Jazz Team Server
- 151559: Enable advanced Jena locking support by default
- 110045: Security and access control for process authoring resources

Page 1 of 5

CCM - Story Status (83) Status

Burndown Current Iteration

Shows the remaining amount of estimated work in hours of work items planned for the current iteration.

Foundation - Burndown

CCM - Burndown

Stories Open/Closed/In Progress

Shows the number of stories which are open, in progress, done during the iteration.

Foundation - Stories Open, Closed, in Progress

CCM - Stories Open, Closed, in Progress

RM - Stories Open, Closed, In Progress

28

Summary - Our Journey from Eclipse to Jazz/RTC and ALM

We developed Eclipse…

Started to reflect practices that worked for us

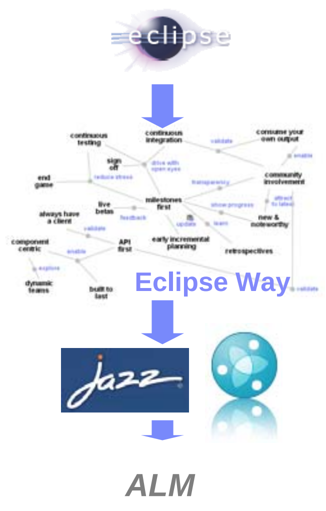
⇒ **the "*Eclipse Way*"**

Then we started to tool the Eclipse Way

⇒***Jazz & Rational Team Concert***

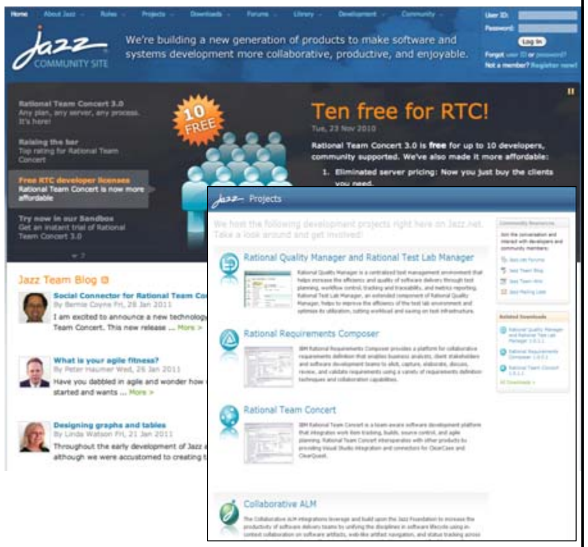
and integrated other Jazz tools across disciplines

⇒***Application Lifecycle Management***



See it live at jazz.net

- Transparent development
  - Jazz architecture
  - Jazz products
- Self-hosting
  - Using Jazz products…
  - … to develop Jazz products
- Learn about Jazz at jazz.net
- Try it
  - Sandbox available
  - Free small teams



We're building a new generation of products to make software and systems development more collaborative, productive, and enjoyable.

Rational Team Concert 3.0
Any plan, any server, any process. It's here!

Raising the bar
Top rating for Rational Team Concert

Free RTC developer licenses
Rational Team Concert is now more affordable

Try now in our Sandbox
Get an instant trial of Rational Team Concert 3.0

Ten free for RTC!
Tue, 23 Nov 2010
Rational Team Concert 3.0 is **free** for up to 10 developers, community supported. We've also made it more affordable:
1. Eliminated server pricing: Now you just buy the clients you need.

Jazz Team Blog

Social Connector for Rational Team Co
By Bernie Coyne Fri, 28 Jan 2011
I am excited to announce a new technology
Team Concert. This new release ... More >

What is your agile fitness?
By Peter Haumer Wed, 26 Jan 2011
Have you dabbled in agile and wonder how
started and wants ... More >

Designing graphs and tables
By Linda Watson Fri, 21 Jan 2011
Throughout the early development of Jazz
although we were accustomed to creating t

We host the following development projects right here on Jazz.net. Take a look around and get involved!

Rational Quality Manager and Rational Test Lab Manager
Rational Quality Manager is a centralized test management environment that helps increase the efficiency and quality of software delivery through test planning, workflow control, tracking and traceability, and metrics reporting. Rational Test Lab Manager, an extended component of Rational Quality Manager, helps to improve the efficiency of the test lab environment and optimize its utilization, cutting workload and saving on test infrastructure.

Rational Requirements Composer
IBM Rational Requirements Composer provides a platform for collaborative requirements definition that enables business analysts, client stakeholders and software development teams to elicit, capture, elaborate, discuss, review, and validate requirements using a variety of requirements definition techniques and collaboration capabilities.

Rational Team Concert
IBM Rational Team Concert is a team-aware software development platform that integrates work item tracking, builds, source control, and agile planning. Rational Team Concert interoperates with other products by providing Visual Studio integration and connectors for ClearCase and ClearQuest.

Collaborative ALM
The Collaborative ALM integrations leverage and build upon the Jazz Foundation to increase the productivity of software delivery teams by unifying the disciplines in software lifecycle using in-context collaboration on software artifacts, web-like artifact navigation, and status tracking across

Community Resources
Join the conversation and interact with developers and community members:
Jazz.net Forums
Jazz Team Blog
Jazz Team Wiki
Jazz Mailing Lists

Related Downloads
Rational Quality Manager and Rational Test Lab Manager 1.0.1.1
Rational Requirements Composer 1.0.1.1
Rational Team Concert 1.0.1.1
All Downloads >