IBM Rational Software
Development Conference
2008

Bring Your Process to Life
Process Enactment in IBM Rational Team Concert

Kai-Uwe Maetzel
Jazz Process Lead, Jazz PMC, IBM Rational
kai-uwe_maetzel@us.ibm.com

SDP 27

Rational. software

Go to IBM

What you learned about Jazz

→

Story 1: The Morning Routine

- Is all about maximizing sleep and still being in time
- Becomes more complex the more people are involved
- Establishes over time
- Becomes more and more effective
  - Change the layout your utensils around the sink
  - Change the layout in the wardrobe so that you don't need light
  - ….
- Avoids obstacles
  - What to say; what not to say
- Gets shattered by life changing events: hotel rooms, a new house, a baby

Story 2: Sharing the Pleasure of Riding the Bicycle

- Groups of unfamiliar riders can be highly efficient
  - Drafting
  - Rotating the lead
  - Setting the pace
  - Indicating obstacles
- Rules are established upfront
- Rules help bridges cultural differences, different levels of experience, and different levels of skills

- The better skilled team is still the faster one
- The experienced team can handle exceptional situations better

Story 3: Football, Soccer, Basketball, …

- Teams practice to establish repertoires of moves and their timing
- Coaches try to detect the opponent's patterns
- Coaches and team decide situationally which moves to make

- The game of a team changes over time
  - Some moves and their timing change quickly and often
  - Others become 'signature' moves

- Successful team are both stable and nimble
  - Their stable foundation allows them to quickly evolve

Story 4: Rules

PRIVATE
PARKING
UNAUTHORIZED VEHICLES WILL BE WORKED OVER WITH A SLEDGEHAMMER, FLIPPED OVER BY AN ANGRY MOB, SET ON FIRE, AND SPRAY PAINTED WITH RUDE SLOGANS IMMEDIATELY AFTER BEING USED AS A GETAWAY CAR IN AN INCREDIBLY DARING DAYLIGHT ROBBERY

CAUTION
THIS SIGN HAS
SHARP EDGES
DO NOT TOUCH THE EDGES OF THIS SIGN
ALSO, THE BRIDGE IS OUT AHEAD

Collaboration

- All collaborations have underlying **context specific rules** and **patterns**
  - **Rules** of engagement
  - Agreement on **behavior patterns**
  - Shared and divided **responsibilities**
  - Agreement on **boundaries for improvising**
- Rules and patterns make collaborators **predictable** for each other
- Rules and patterns exist on all levels
  - Micro level (e.g., the communication dance)
  - Macro level (e.g., international, cooperate rules)

Collaboration Rules and Patterns

- Are **goal specific**
- **Emerge** over time (1 & 3) **or** are agreed on **up-front** (2 & 3)
- **Evolve** over time based on feedback (1, 2, & 3)
- Incorporate **situational** sub-patterns and rules
  - the wardrobe is locked starting around 4 weeks before Christmas
  - don't party after 10 pm at night
  - don't make open fire during summer
- Range from **generic to actor specific**
- Decide about success and failure of collaborations

- What works best for you might not work for others but is worth sharing.

Taking the Next Step

- Make tools smarter: We did it before, we continue to do so
- Make explicit what you care about.

- Integrate the higher level concepts of collaboration into collaborative software development tools.
  - **Reify** the concept of **collaboration rules** and **patterns** and their parts
  - **Emphasize artifacts** as they are in the middle of collaborations
  - **Honor the diversity** of teams and their collaborations

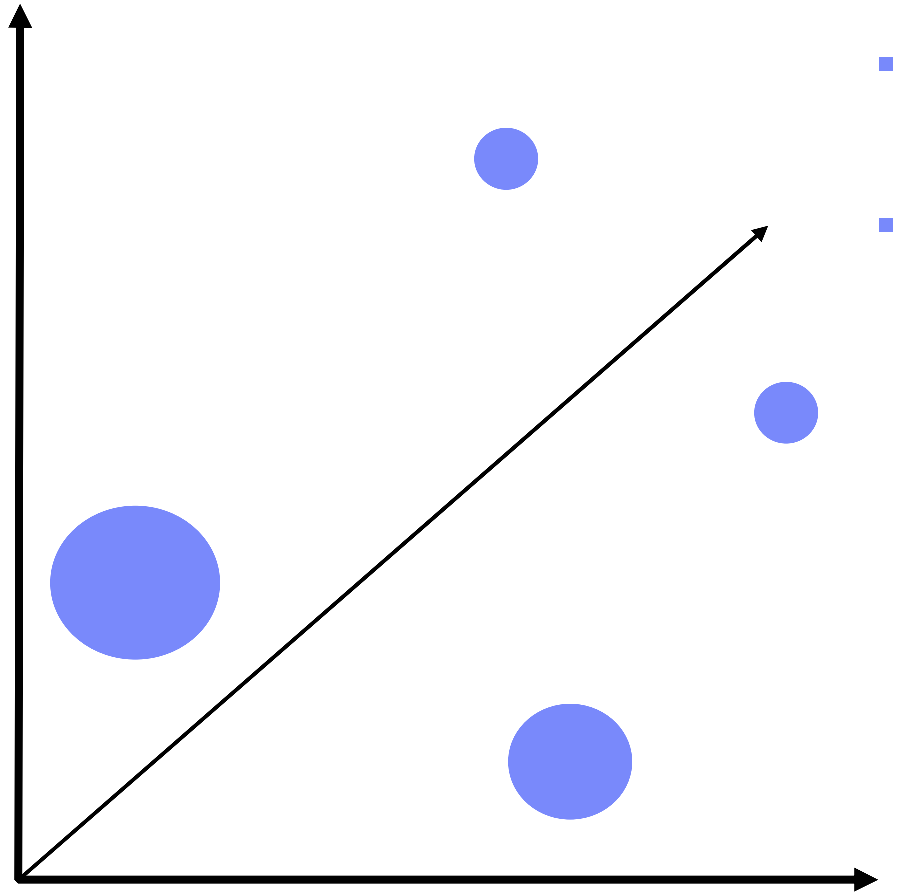
- If the **tools** know they can be way **more helpful**.
- Don’t be presumptuous about the ‘right way’.

Free the Mind for Creativity

- Give a helping hand to people like me who can not or don’t want to remember all the details.
- **Enable** higher **productivity**

- Why do you like refactoring in JDT?
  - it takes away the burden to do it manually
  - update all references
  - reminds you that references are not only in code but also in plugin.xml files, comments, …
- Why do you like garbage collection in Java?
- Why do you like reminders in calendars?
- …

Not a single 'Right Way' but many Good Places



- There is **more than one** good place
- Good places are
  - Team specific
  - Goal specific
  - Culture specific
  - Experience specific
  - Application domain specific
  - Depending on regulations
  - ….

Why is this a good place?

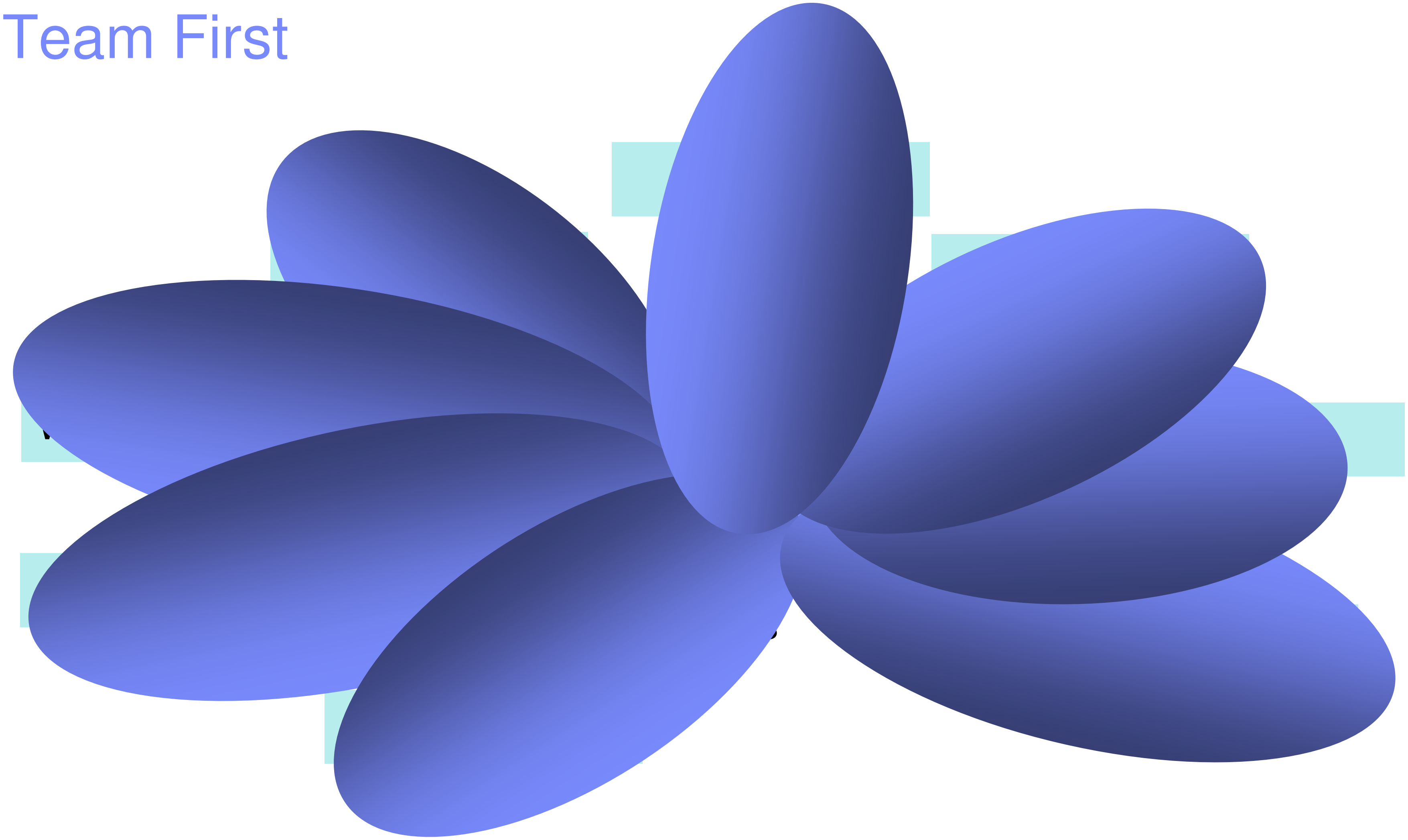
- “This is a good place.”
- “This is our place.”
- “This is the ‘distance’ between our place and a good place.”
- “This is how we get from our place to a good place.”

- Tools need to allow for **transparency**, **monitoring**, and **introspection**.
- It needs subjective and objective measures to effectively self improve.
  - Reports and dashboards

Jazz and Collaboration

- We call collaboration rules and patterns **Process**.
- Process is part of the **core** of the Jazz platform.
- Jazz is process **neutral**.
- Jazz means ‘enactment’ when it says ‘**enactment**’.
  - Guide and advise
  - Enforce defined rules
  - Live process

Team First



Jazz Process Support

- Support different degrees of flexibility and formalism
- Allows for **predefined** processes
- Allows for **emerging** processes
- Allows for **variations**
- Allows for **exceptions**
- Allows for process **consolidation**
- Allows for process **evolution** in general
- Allows for **extensions**
- Put knowledgeable human in the center

- Comprises runtime, authoring, and inspection support

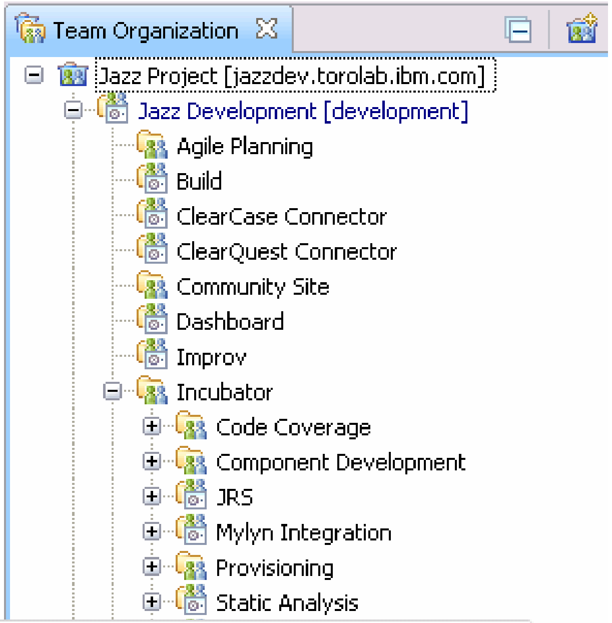
The Basic Model of Process

- Teams work on projects
- Each project follows a process
- Each team is unique and thus can work differently
- Work inside the scope of a team follows the team’s process
- Cross-team work follows the process of the broader team
- Team members play roles defined by the process
- Process manifests itself through artifacts types, operations manipulating the artifacts, and artifact change events.

Organizational Structure

- **Project areas** reify the notion of project.
  - Multiple project areas per Jazz server

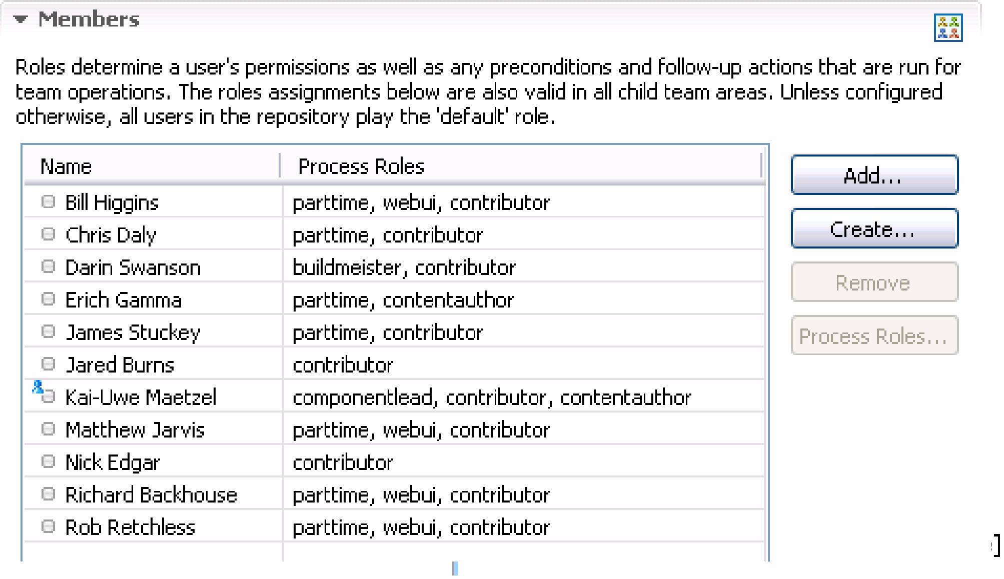
- **Team areas** reify the notion of team.
  - A project area contains a hierarchy of team areas.



- Jazz Project [jazzdev.torolab.ibm.com]
  - Jazz Development [development]
    - Agile Planning
    - Build
    - ClearCase Connector
    - ClearQuest Connector
    - Community Site
    - Dashboard
    - Improv
    - Incubator
      - Code Coverage
      - Component Development
      - JRS
      - Mylyn Integration
      - Provisioning
      - Static Analysis

- Team areas manage team **membership** and **roles** assignments.

- Team **artifacts** are owned by the team area.

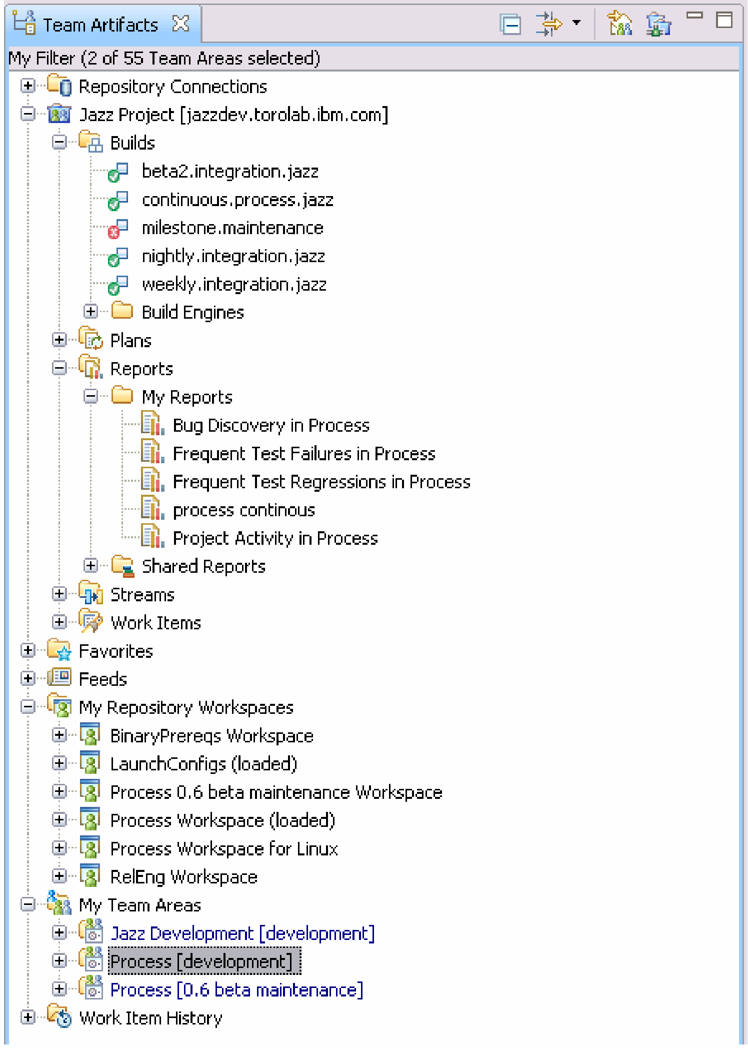


Roles determine a user's permissions as well as any preconditions and follow-up actions that are run for team operations. The roles assignments below are also valid in all child team areas. Unless configured otherwise, all users in the repository play the 'default' role.

Name	Process Roles
Bill Higgins	parttime, webui, contributor
Chris Daly	parttime, contributor
Darin Swanson	buildmeister, contributor
Erich Gamma	parttime, contentauthor
James Stuckey	parttime, contributor
Jared Burns	contributor
Kai-Uwe Maetzel	componentlead, contributor, contentauthor
Matthew Jarvis	parttime, webui, contributor
Nick Edgar	contributor
Richard Backhouse	parttime, webui, contributor
Rob Retchless	parttime, webui, contributor

Implication: You are interested in…

- Artifact presentation filtered
  - By team membership
  - By ownership

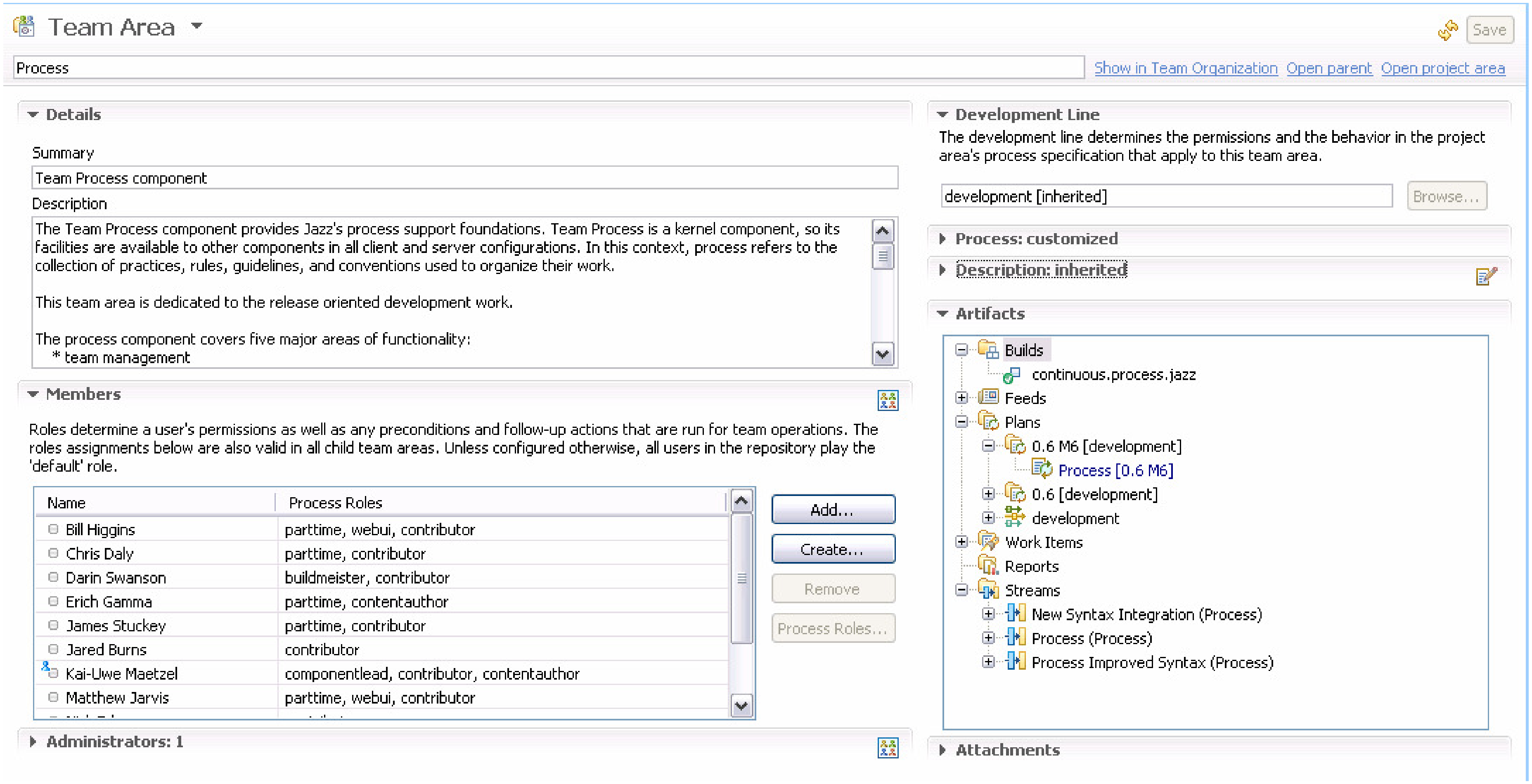


Team Artifacts

My Filter (2 of 55 Team Areas selected)

- Repository Connections
- Jazz Project [jazzdev.torolab.ibm.com]
  - Builds
    - beta2.integration.jazz
    - continuous.process.jazz
    - milestone.maintenance
    - nightly.integration.jazz
    - weekly.integration.jazz
    - Build Engines
  - Plans
  - Reports
    - My Reports
      - Bug Discovery in Process
      - Frequent Test Failures in Process
      - Frequent Test Regressions in Process
      - process continous
      - Project Activity in Process
    - Shared Reports
  - Streams
  - Work Items
- Favorites
- Feeds
- My Repository Workspaces
  - BinaryPrereqs Workspace
  - LaunchConfigs (loaded)
  - Process 0.6 beta maintenance Workspace
  - Process Workspace (loaded)
  - Process Workspace for Linux
  - RelEng Workspace
- My Team Areas
  - Jazz Development [development]
  - Process [development]
  - Process [0.6 beta maintenance]
- Work Item History

Implication: Your team and your artifacts…



Process

Details

Summary

Team Process component

Description

The Team Process component provides Jazz's process support foundations. Team Process is a kernel component, so its facilities are available to other components in all client and server configurations. In this context, process refers to the collection of practices, rules, guidelines, and conventions used to organize their work.

This team area is dedicated to the release oriented development work.

The process component covers five major areas of functionality:
  * team management

Members

Roles determine a user's permissions as well as any preconditions and follow-up actions that are run for team operations. The roles assignments below are also valid in all child team areas. Unless configured otherwise, all users in the repository play the 'default' role.

Name	Process Roles
Bill Higgins	parttime, webui, contributor
Chris Daly	parttime, contributor
Darin Swanson	buildmeister, contributor
Erich Gamma	parttime, contentauthor
James Stuckey	parttime, contributor
Jared Burns	contributor
Kai-Uwe Maetzel	componentlead, contributor, contentauthor
Matthew Jarvis	parttime, webui, contributor

Administrators: 1

Development Line

The development line determines the permissions and the behavior in the project area's process specification that apply to this team area.

development [inherited]

Process: customized

Description: inherited

Artifacts

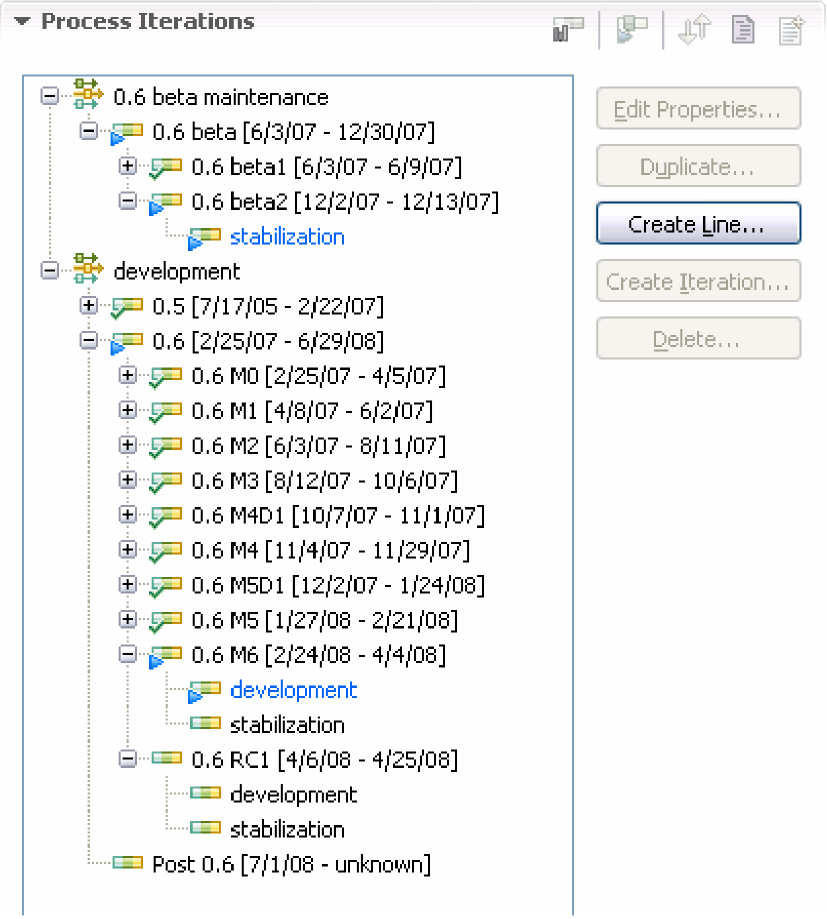
- Builds
  - continuous.process.jazz
- Feeds
- Plans
  - 0.6 M6 [development]
    - Process [0.6 M6]
  - 0.6 [development]
  - development
- Work Items
- Reports
- Streams
  - New Syntax Integration (Process)
  - Process (Process)
  - Process Improved Syntax (Process)

Attachments

Temporal Structure

- **Development lines**
  - Example: development, maintenance, e4
  - Each team area belongs to exactly one development line
  - Contain iterations

- **Iterations**
  - Arbitrarily nested iteration structure
  - Each iteration usually has a start and end date
  - There is one **current iteration** per development line



- 0.6 beta maintenance
  - 0.6 beta [6/3/07 - 12/30/07]
    - 0.6 beta1 [6/3/07 - 6/9/07]
    - 0.6 beta2 [12/2/07 - 12/13/07]
      - stabilization
- development
  - 0.5 [7/17/05 - 2/22/07]
  - 0.6 [2/25/07 - 6/29/08]
    - 0.6 M0 [2/25/07 - 4/5/07]
    - 0.6 M1 [4/8/07 - 6/2/07]
    - 0.6 M2 [6/3/07 - 8/11/07]
    - 0.6 M3 [8/12/07 - 10/6/07]
    - 0.6 M4D1 [10/7/07 - 11/1/07]
    - 0.6 M4 [11/4/07 - 11/29/07]
    - 0.6 M5D1 [12/2/07 - 1/24/08]
    - 0.6 M5 [1/27/08 - 2/21/08]
    - 0.6 M6 [2/24/08 - 4/4/08]
      - development
      - stabilization
    - 0.6 RC1 [4/6/08 - 4/25/08]
      - development
      - stabilization
  - Post 0.6 [7/1/08 - unknown]

Process Structure

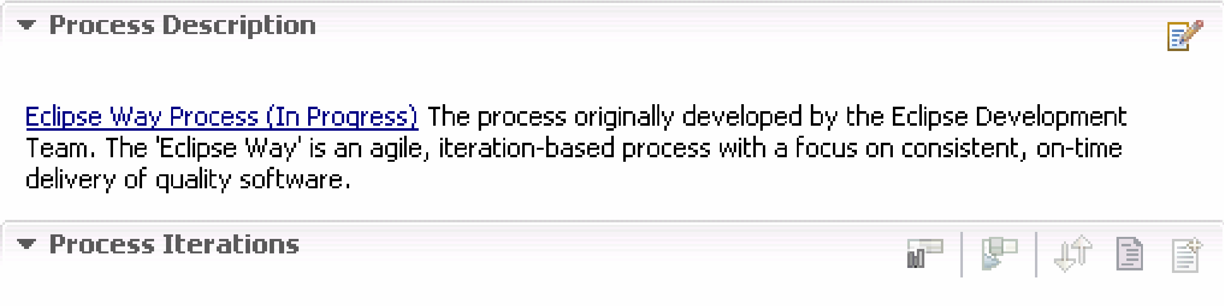
- A project area contains a **process specification** and **process description**.
- Process specification focuses on the formalized aspects.
- Process description focuses on the non-formalized aspects.

- Each team area can have a **process customization** and **process description** or inherit the process from its parent.

- Process customization can extend or partially replace the inherited process.

Process Description

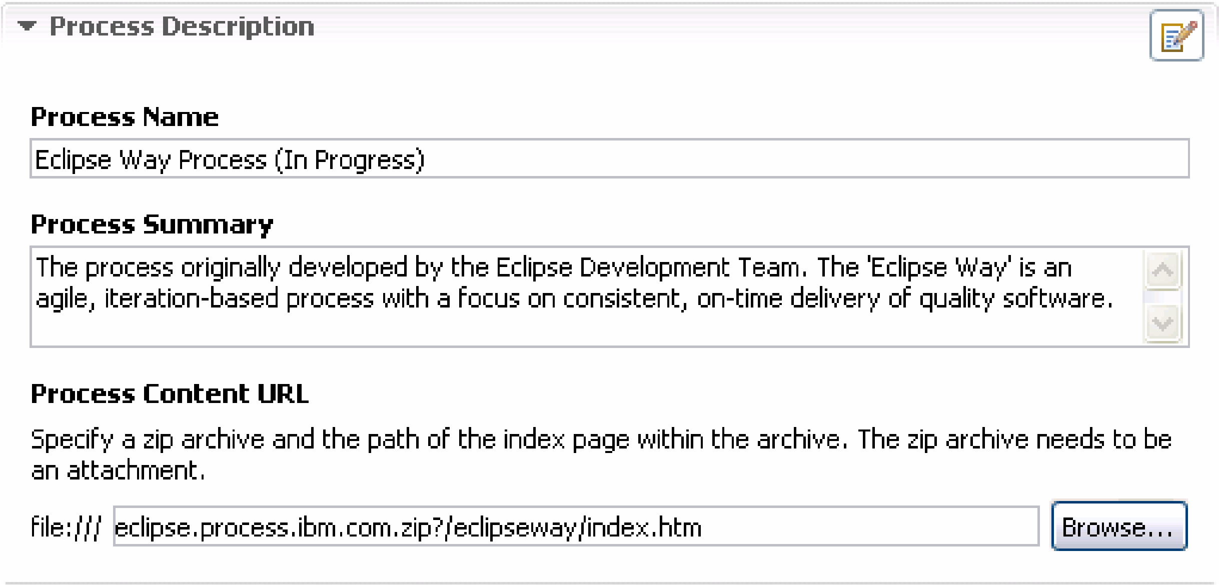
- Presentation mode:



Eclipse Way Process (In Progress) The process originally developed by the Eclipse Development Team. The 'Eclipse Way' is an agile, iteration-based process with a focus on consistent, on-time delivery of quality software.

⊟ 0.6 beta maintenance
⊟ 0.6 beta [6/3/07 - 12/30/
⊞ 0.6 beta1 [6/3/07 - 6/

- Edit mode:



Process Name

Eclipse Way Process (In Progress)

Process Summary

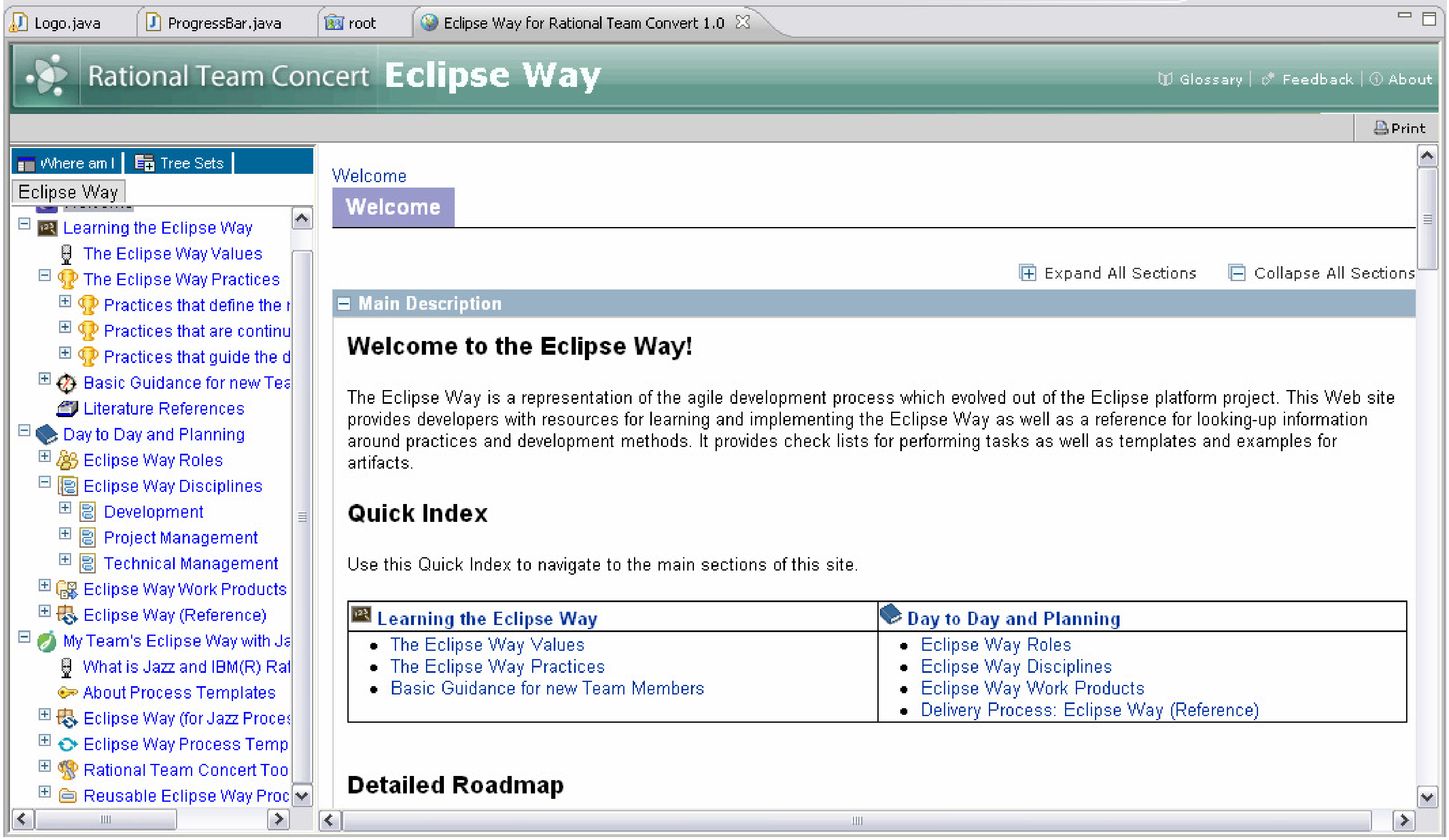
The process originally developed by the Eclipse Development Team. The 'Eclipse Way' is an agile, iteration-based process with a focus on consistent, on-time delivery of quality software.

Process Content URL

Specify a zip archive and the path of the index page within the archive. The zip archive needs to be an attachment.

file:/// eclipse.process.ibm.com.zip?/eclipseway/index.htm

Process Description



Welcome to the Eclipse Way!

The Eclipse Way is a representation of the agile development process which evolved out of the Eclipse platform project. This Web site provides developers with resources for learning and implementing the Eclipse Way as well as a reference for looking-up information around practices and development methods. It provides check lists for performing tasks as well as templates and examples for artifacts.

Quick Index

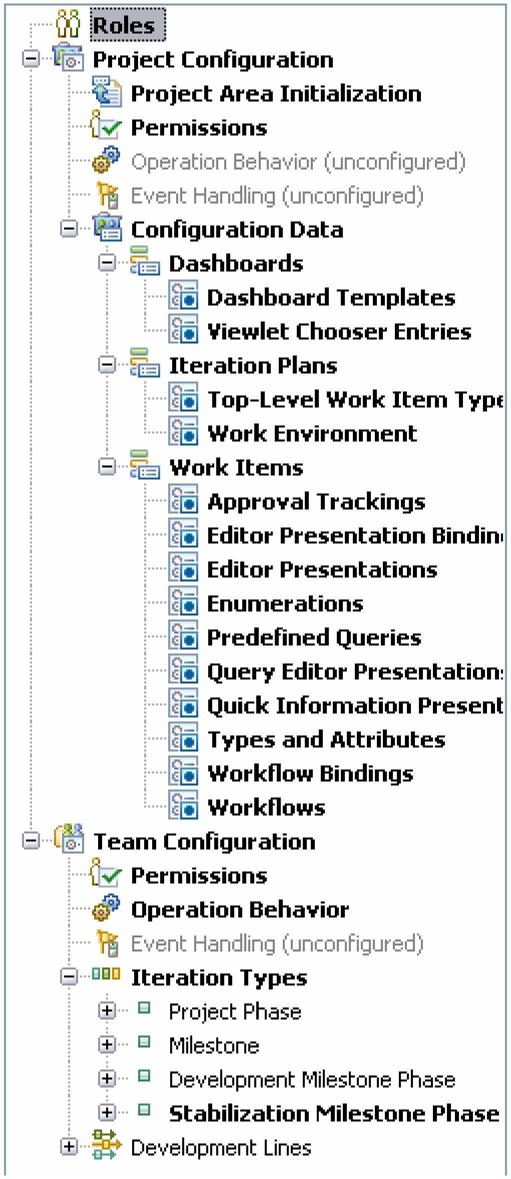
Use this Quick Index to navigate to the main sections of this site.

Learning the Eclipse Way	Day to Day and Planning
• The Eclipse Way Values • The Eclipse Way Practices • Basic Guidance for new Team Members	• Eclipse Way Roles • Eclipse Way Disciplines • Eclipse Way Work Products • Delivery Process: Eclipse Way (Reference)

Detailed Roadmap

Process Specification Talks About

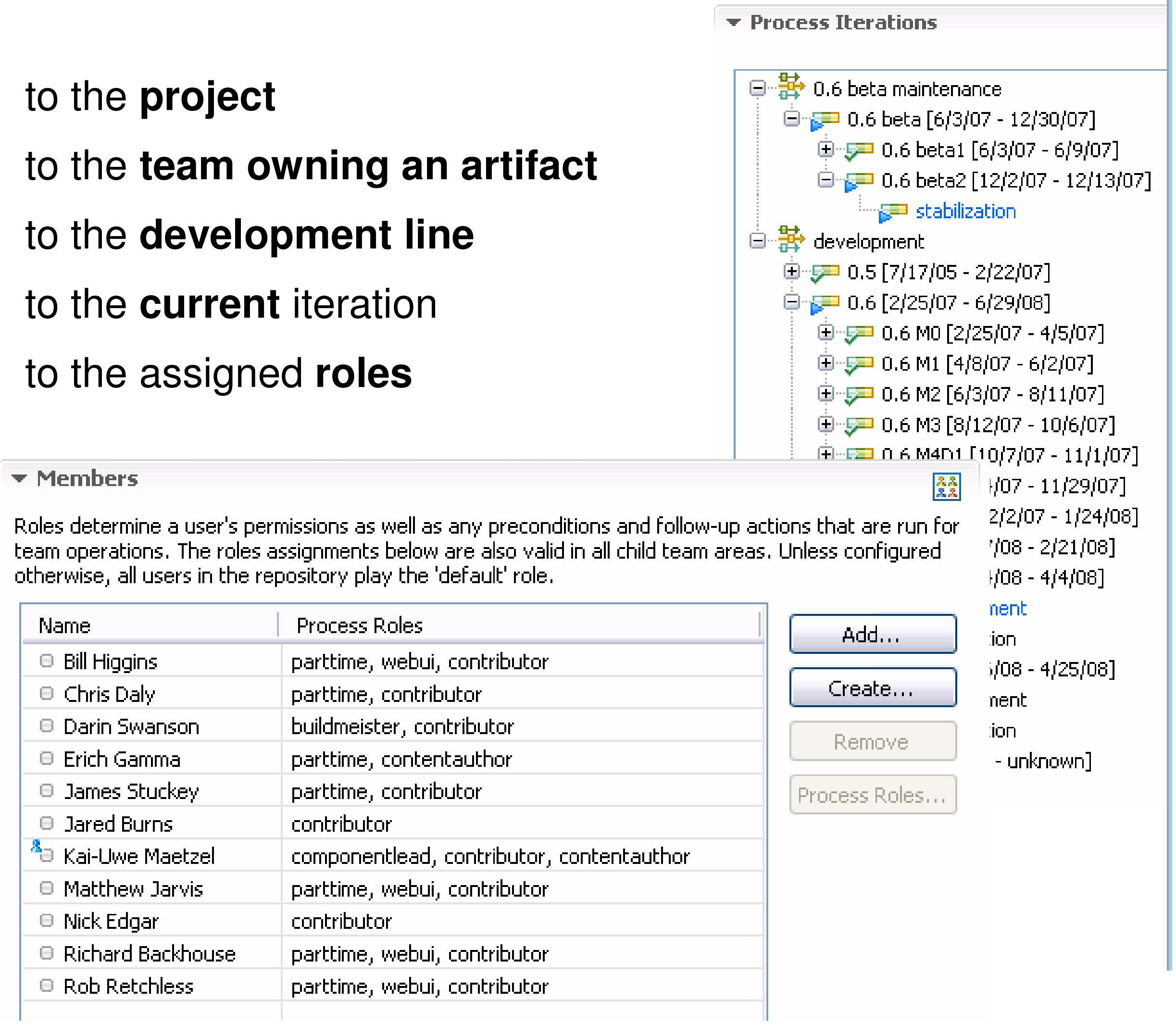
- Roles
- Project configuration
  - permissions for project operations
  - behavior
    - preconditions for project operations
    - follow-up actions for project operations
    - event handlers for project events
  - configuration data
- Team configuration
  - Iteration specific permissions for team operations
  - Iteration specific behavior for team operations
  - Iteration specific behavior for team events



- Roles
- Project Configuration
  - Project Area Initialization
  - Permissions
  - Operation Behavior (unconfigured)
  - Event Handling (unconfigured)
  - Configuration Data
    - Dashboards
      - Dashboard Templates
      - Viewlet Chooser Entries
    - Iteration Plans
      - Top-Level Work Item Type
      - Work Environment
    - Work Items
      - Approval Trackings
      - Editor Presentation Bindin
      - Editor Presentations
      - Enumerations
      - Predefined Queries
      - Query Editor Presentation
      - Quick Information Present
      - Types and Attributes
      - Workflow Bindings
      - Workflows
- Team Configuration
  - Permissions
  - Operation Behavior
  - Event Handling (unconfigured)
  - Iteration Types
    - Project Phase
    - Milestone
    - Development Milestone Phase
    - Stabilization Milestone Phase
  - Development Lines

The Effective Process is Specific…

- to the **project**
- to the **team owning an artifact**
- to the **development line**
- to the **current** iteration
- to the assigned **roles**



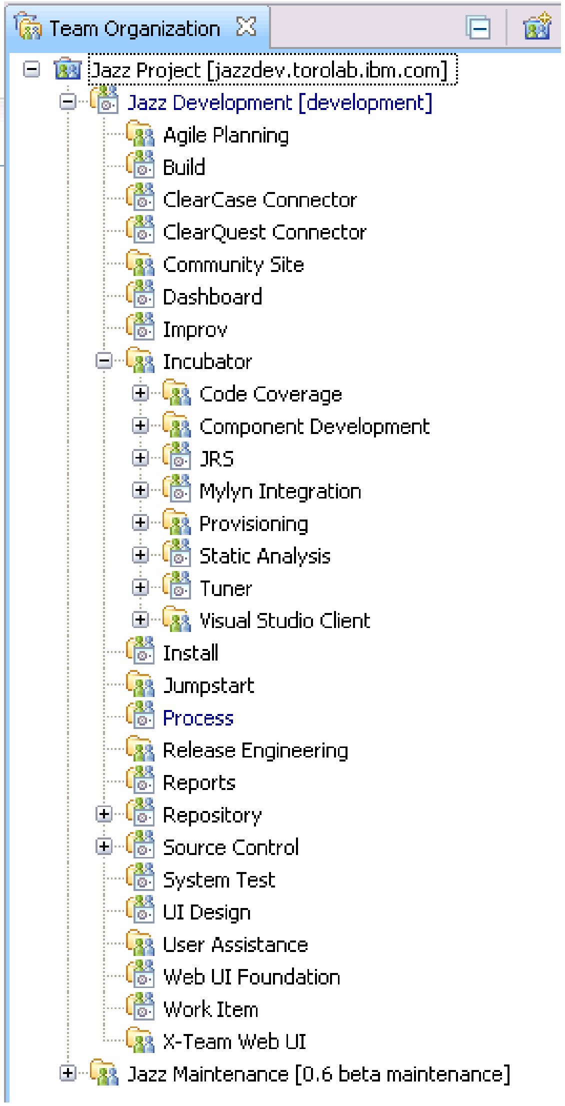
Process Iterations

```
⊟ 0.6 beta maintenance
   ⊟ 0.6 beta [6/3/07 - 12/30/07]
      ⊞ 0.6 beta1 [6/3/07 - 6/9/07]
      ⊟ 0.6 beta2 [12/2/07 - 12/13/07]
         stabilization
⊟ development
   ⊞ 0.5 [7/17/05 - 2/22/07]
   ⊟ 0.6 [2/25/07 - 6/29/08]
      ⊞ 0.6 M0 [2/25/07 - 4/5/07]
      ⊞ 0.6 M1 [4/8/07 - 6/2/07]
      ⊞ 0.6 M2 [6/3/07 - 8/11/07]
      ⊞ 0.6 M3 [8/12/07 - 10/6/07]
      ⊞ 0.6 M4D1 [10/7/07 - 11/1/07]
        /07 - 11/29/07]
        2/2/07 - 1/24/08]
        /08 - 2/21/08]
        /08 - 4/4/08]
        nent
        ion
        /08 - 4/25/08]
        nent
        ion
        - unknown]
```

Members

Roles determine a user's permissions as well as any preconditions and follow-up actions that are run for team operations. The roles assignments below are also valid in all child team areas. Unless configured otherwise, all users in the repository play the 'default' role.

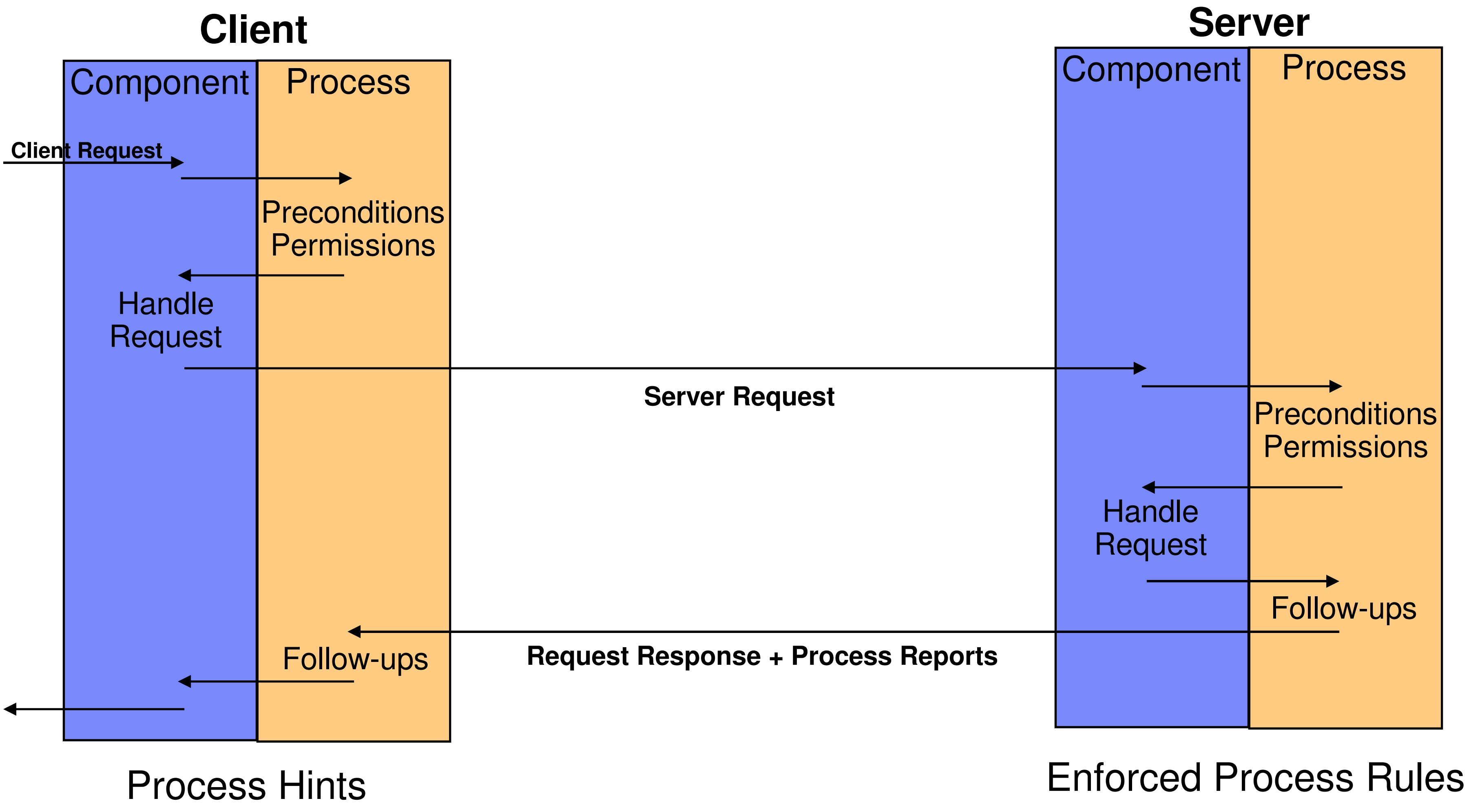
Name	Process Roles
Bill Higgins	parttime, webui, contributor
Chris Daly	parttime, contributor
Darin Swanson	buildmeister, contributor
Erich Gamma	parttime, contentauthor
James Stuckey	parttime, contributor
Jared Burns	contributor
Kai-Uwe Maetzel	componentlead, contributor, contentauthor
Matthew Jarvis	parttime, webui, contributor
Nick Edgar	contributor
Richard Backhouse	parttime, webui, contributor
Rob Retchless	parttime, webui, contributor



Team Organization

```
⊟ Jazz Project [jazzdev.torolab.ibm.com]
   ⊟ Jazz Development [development]
      Agile Planning
      Build
      ClearCase Connector
      ClearQuest Connector
      Community Site
      Dashboard
      Improv
      ⊟ Incubator
         ⊞ Code Coverage
         ⊞ Component Development
         ⊞ JRS
         ⊞ Mylyn Integration
         ⊞ Provisioning
         ⊞ Static Analysis
         ⊞ Tuner
         ⊞ Visual Studio Client
      Install
      Jumpstart
      Process
      Release Engineering
      Reports
      ⊞ Repository
      ⊞ Source Control
      System Test
      UI Design
      User Assistance
      Web UI Foundation
      Work Item
      X-Team Web UI
   ⊞ Jazz Maintenance [0.6 beta maintenance]
```

Process Execution Flow



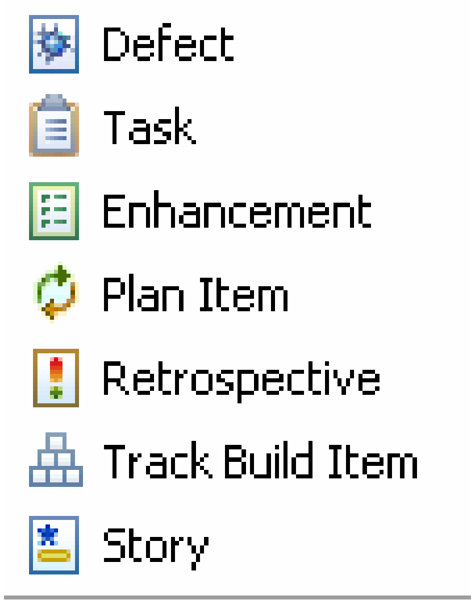
Designing for Process

- Process enablement is **not** a **transparent** platform feature

- Process enablement is **explicit design for the end user** while being process neutral
  - What are the events the component sends out
  - What are the component’s operations
  - What are the permissions for each of the operations
  - Which configuration data does the component need
    - How should it be structured
    - What dependencies are there between the different types of configuration data

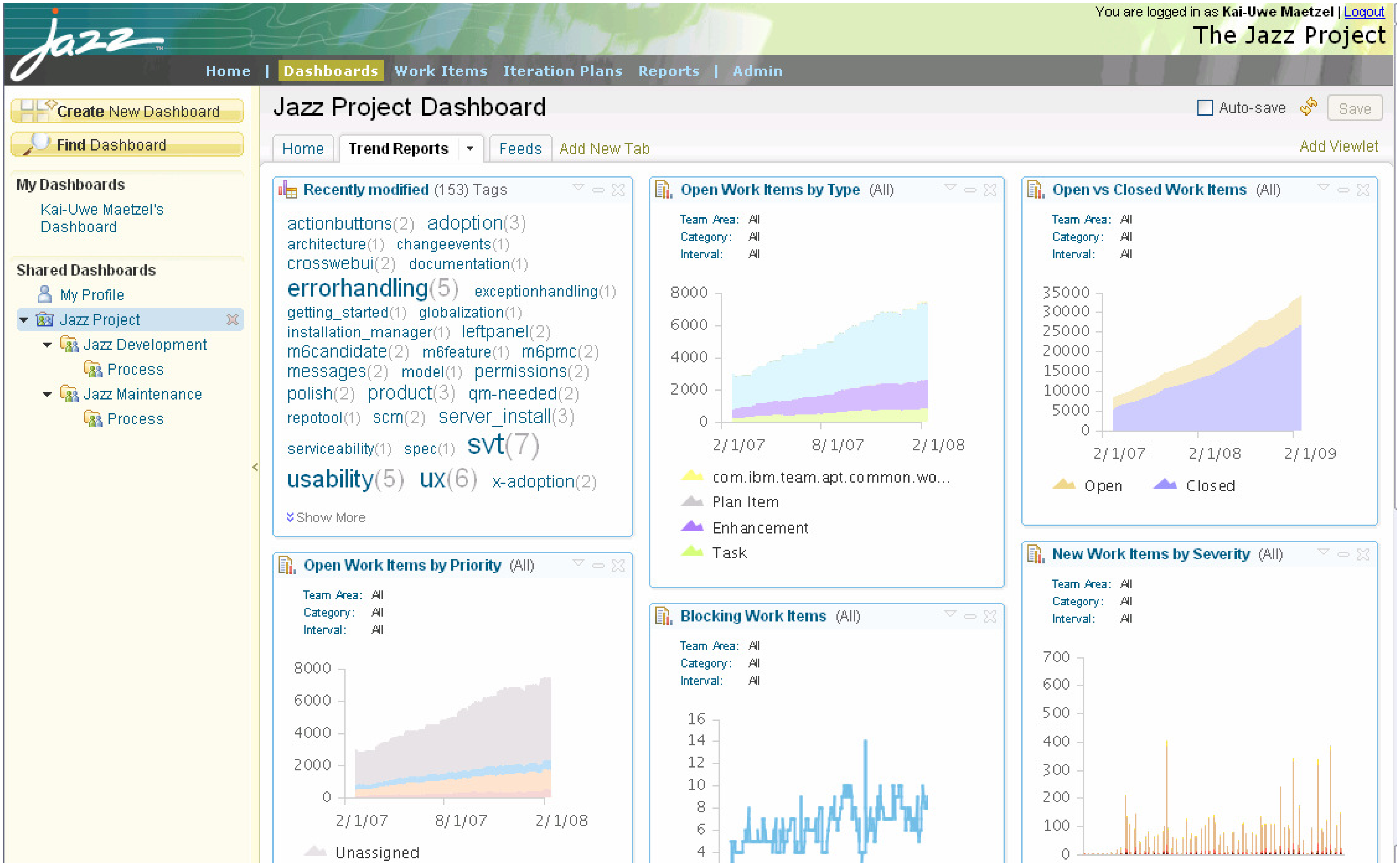
Case Study: ‘Eclipse Way’

- Developed based on our experience with the eclipse platform project
- Covers
  - Basic iteration types:
    - project phase, milestone, development and stabilization milestone phase
  - Basic roles: team lead, contributor
  - Appropriate role based permissions
  - Work item types and their workflows:
  - Agile planning support in form of Stories
  - Reports and dashboard templates
  - SCM delivery rules
  - Rules for joining a team



- Defect
- Task
- Enhancement
- Plan Item
- Retrospective
- Track Build Item
- Story

Case Study: ‘Eclipse Way’


You are logged in as **Kai-Uwe Maetzel** | Logout

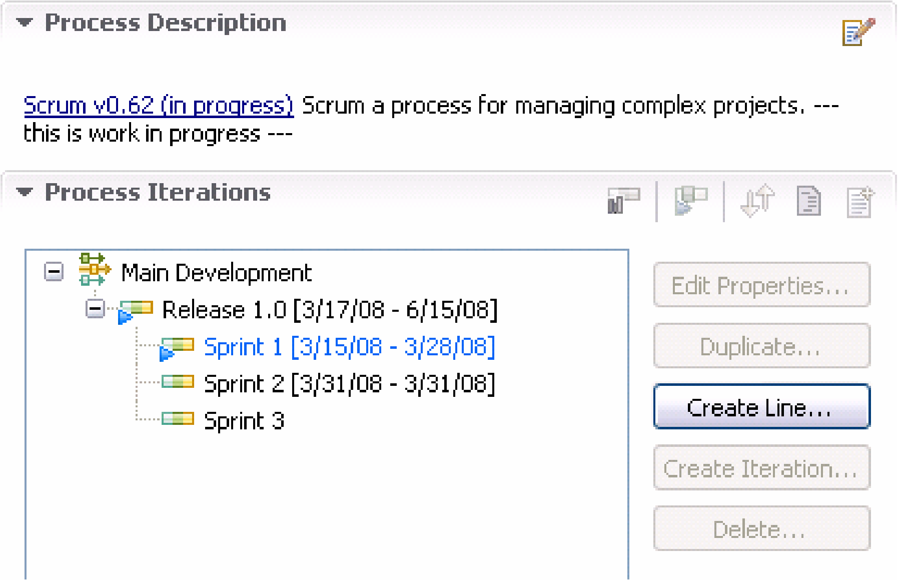
Team Area: All
Category: All
Interval: All

jazz
The Jazz Project
Home | Dashboards Work Items Iteration Plans Reports | Admin
Create New Dashboard
Find Dashboard
My Dashboards
Kai-Uwe Maetzel's Dashboard
Shared Dashboards
My Profile
Jazz Project
Jazz Development
Process
Jazz Maintenance
Process
Jazz Project Dashboard
Auto-save
Save
Home
Trend Reports
Feeds
Add New Tab
Add Viewlet
Recently modified (153) Tags
actionbuttons(2) adoption(3) architecture(1) changeevents(1) crosswebui(2) documentation(1) errorhandling(5) exceptionhandling(1) getting_started(1) globalization(1) installation_manager(1) leftpanel(2) m6candidate(2) m6feature(1) m6pmc(2) messages(2) model(1) permissions(2) polish(2) product(3) qm-needed(2) repotool(1) scm(2) server_install(3) serviceability(1) spec(1) svt(7) usability(5) ux(6) x-adoption(2)
Show More
Open Work Items by Type (All)
8000
6000
4000
2000
0
2/1/07
8/1/07
2/1/08
com.ibm.team.apt.common.wo...
Plan Item
Enhancement
Task
Open vs Closed Work Items (All)
35000
30000
25000
20000
15000
10000
5000
0
2/1/07
2/1/08
2/1/09
Open
Closed
Open Work Items by Priority (All)
8000
6000
4000
2000
0
2/1/07
8/1/07
2/1/08
Unassigned
Blocking Work Items (All)
16
14
12
10
8
6
4
New Work Items by Severity (All)
700
600
500
400
300
200
100
0

Case Study: Scrum

- Early work to provide Scrum support
- First version was developed within a few hours
- Is being continuously improved based on Scrum Master feedback

Scrum: Iteration Structure



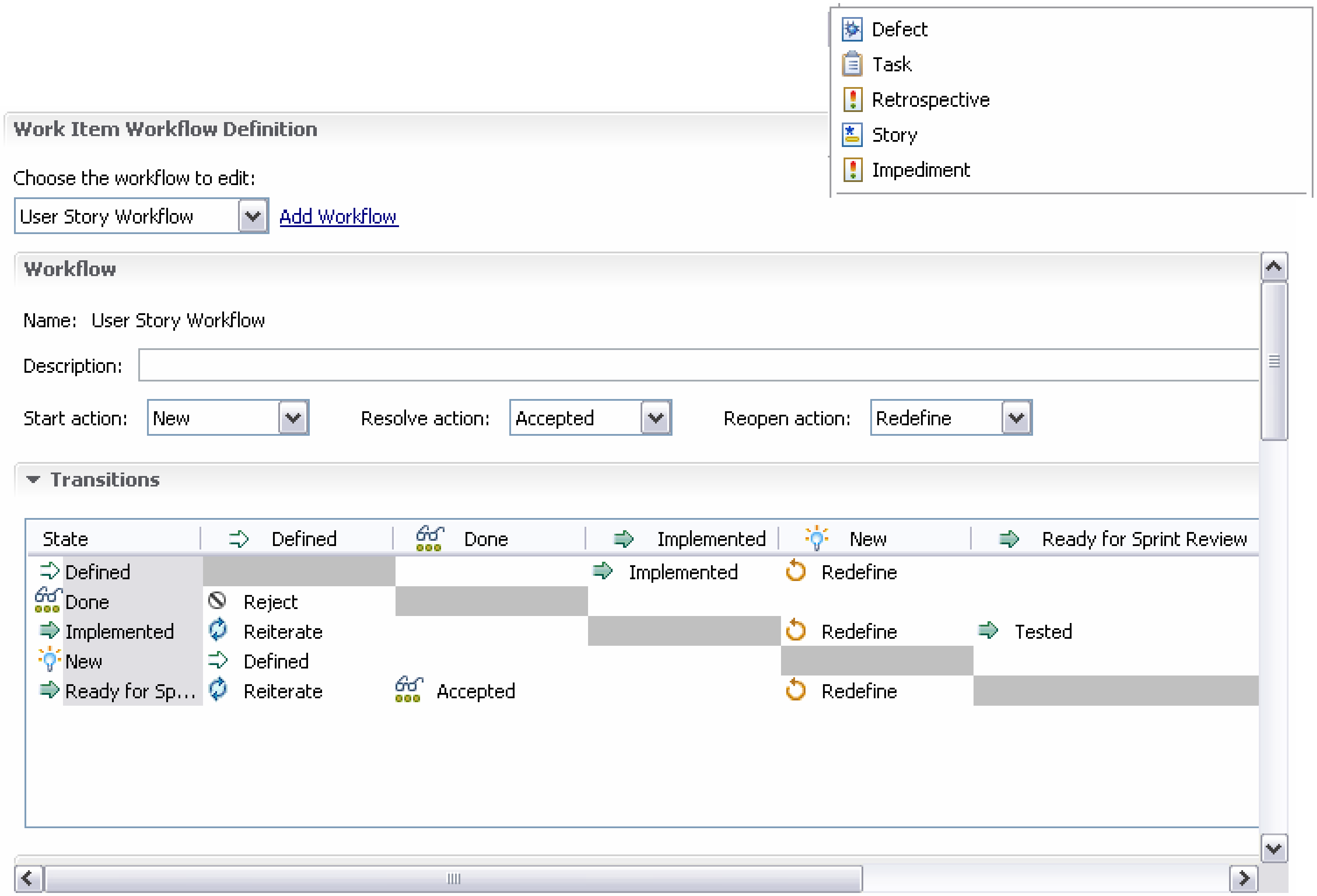
Process Description

Scrum v0.62 (in progress) Scrum a process for managing complex projects. --- this is work in progress ---

Process Iterations

- Main Development
  - Release 1.0 [3/17/08 - 6/15/08]
    - Sprint 1 [3/15/08 - 3/28/08]
    - Sprint 2 [3/31/08 - 3/31/08]
    - Sprint 3

Scrum: Work Item Types



- Defect
- Task
- Retrospective
- Story
- Impediment

Work Item Workflow Definition

Choose the workflow to edit:

User Story Workflow | Add Workflow

Workflow

Name: User Story Workflow

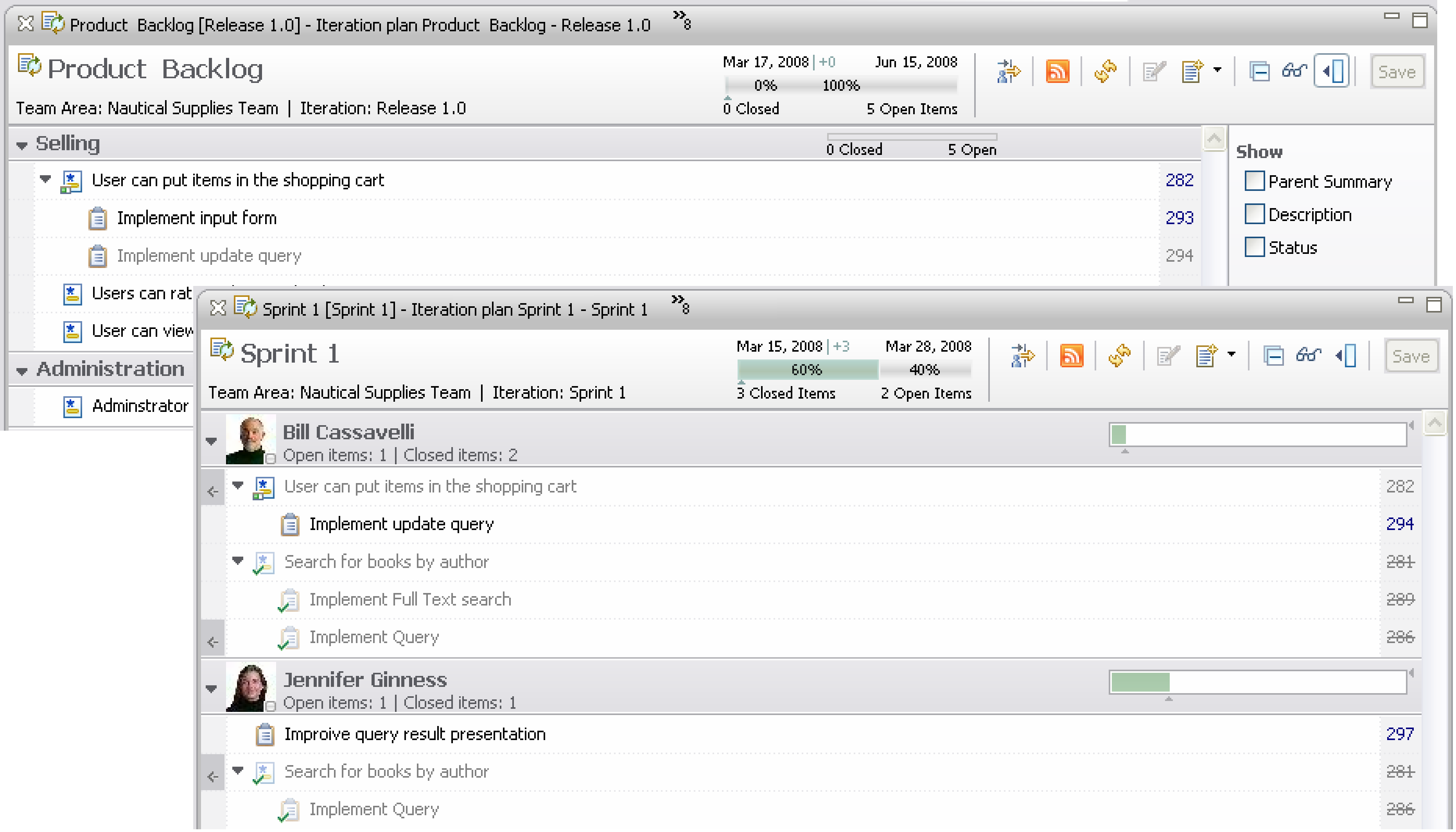
Description:

Start action: New | Resolve action: Accepted | Reopen action: Redefine

Transitions

State	Defined	Done	Implemented	New	Ready for Sprint Review
Defined			Implemented	Redefine	
Done	Reject				
Implemented	Reiterate			Redefine	Tested
New	Defined				
Ready for Sp...	Reiterate	Accepted		Redefine	

Scrum: Using iteration plans for backlogs



Product Backlog [Release 1.0] - Iteration plan Product Backlog - Release 1.0

Product Backlog

Team Area: Nautical Supplies Team | Iteration: Release 1.0

Mar 17, 2008 | +0 Jun 15, 2008
0% 100%
0 Closed 5 Open Items

Selling 0 Closed 5 Open

- User can put items in the shopping cart 282
  - Implement input form 293
  - Implement update query 294
- Users can rat
- User can view

Administration

- Adminstrator

Show

□ Parent Summary
□ Description
□ Status

Sprint 1 [Sprint 1] - Iteration plan Sprint 1 - Sprint 1

Sprint 1

Team Area: Nautical Supplies Team | Iteration: Sprint 1

Mar 15, 2008 | +3 Mar 28, 2008
60% 40%
3 Closed Items 2 Open Items

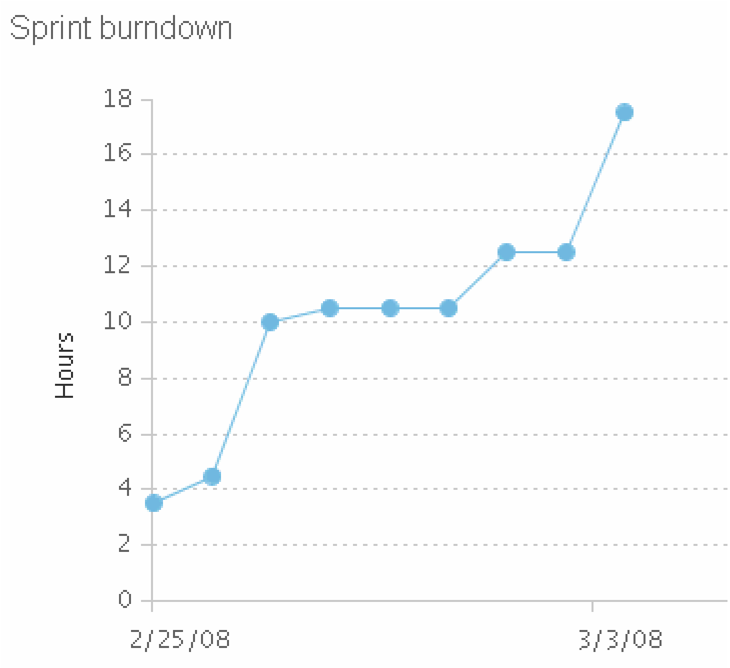
Bill Cassavelli
Open items: 1 | Closed items: 2

- User can put items in the shopping cart 282
  - Implement update query 294
- Search for books by author ~~281~~
  - Implement Full Text search ~~289~~
  - Implement Query ~~286~~

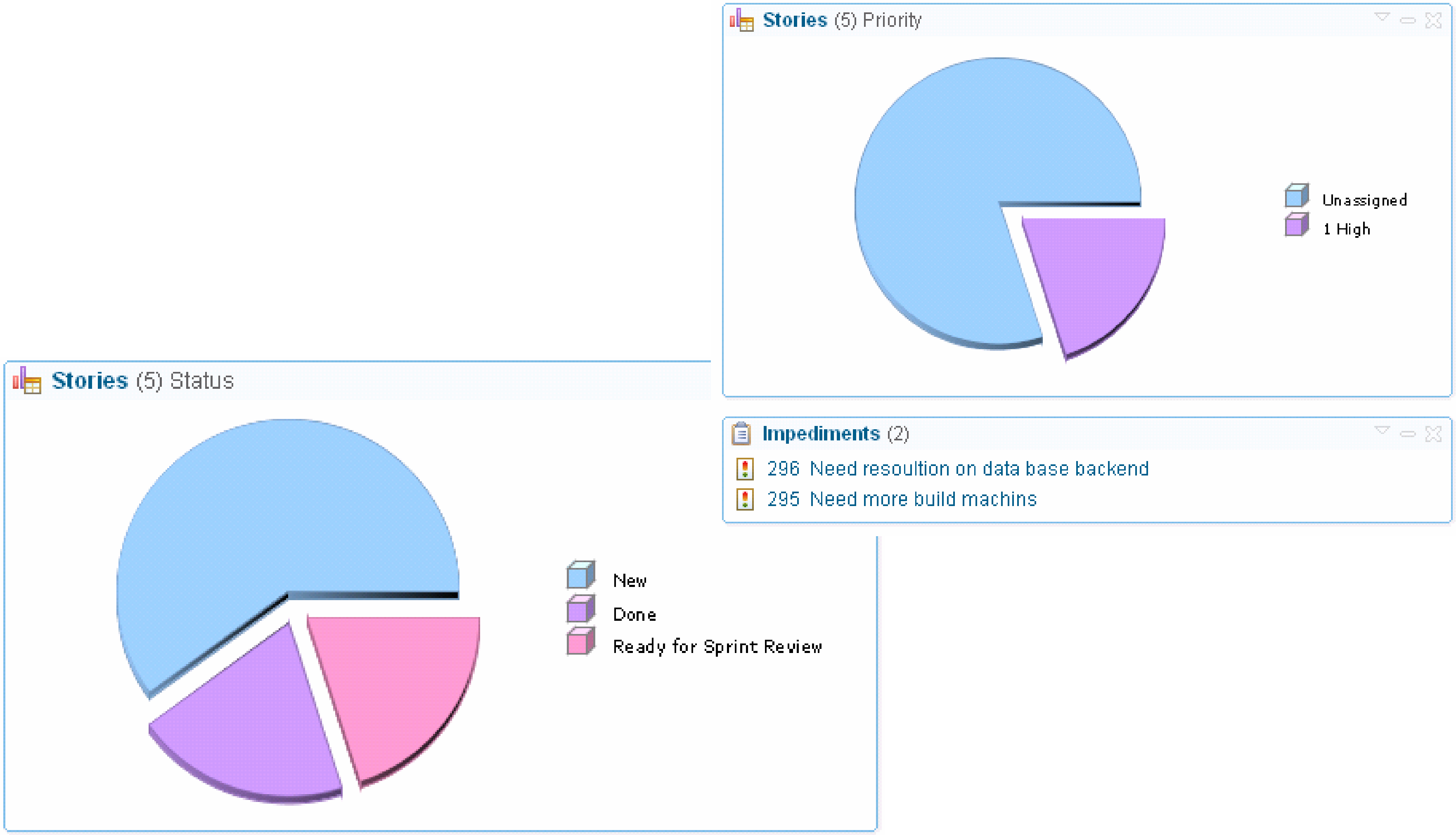
Jennifer Ginness
Open items: 1 | Closed items: 1

- Improive query result presentation 297
- Search for books by author ~~281~~
  - Implement Query ~~286~~

Scrum: Burndown Report



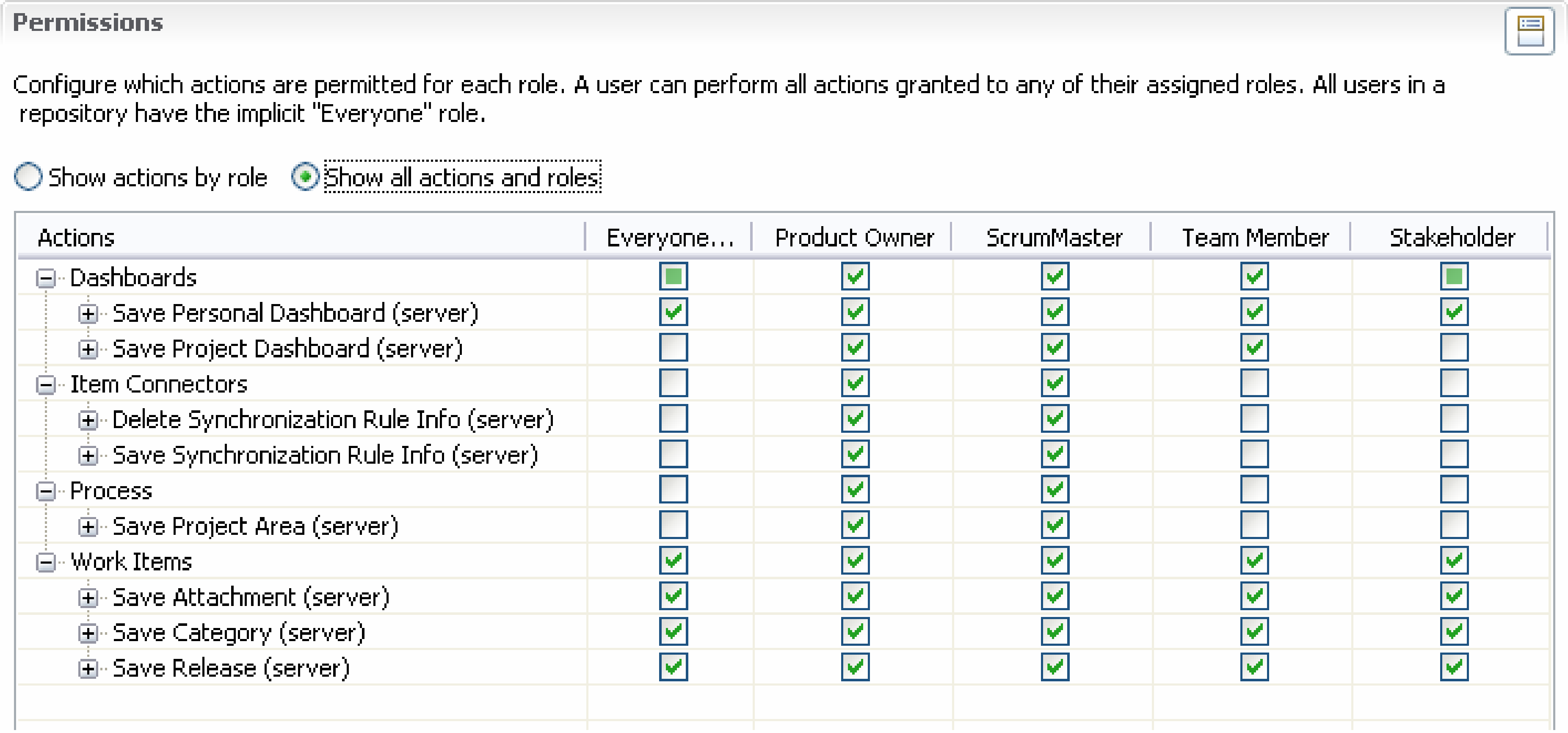
Scrum: Dashboard Viewlets



296 Need resoultion on data base backend

295 Need more build machins

Scrum: Roles and Permissions



Permissions

Configure which actions are permitted for each role. A user can perform all actions granted to any of their assigned roles. All users in a repository have the implicit "Everyone" role.

○ Show actions by role ● Show all actions and roles

Actions	Everyone...	Product Owner	ScrumMaster	Team Member	Stakeholder
⊟ Dashboards	🟩	☑	☑	☑	🟩
⊞ Save Personal Dashboard (server)	☑	☑	☑	☑	☑
⊞ Save Project Dashboard (server)	☐	☑	☑	☑	☐
⊟ Item Connectors	☐	☑	☑	☐	☐
⊞ Delete Synchronization Rule Info (server)	☐	☑	☑	☐	☐
⊞ Save Synchronization Rule Info (server)	☐	☑	☑	☐	☐
⊟ Process	☐	☑	☑	☐	☐
⊞ Save Project Area (server)	☐	☑	☑	☐	☐
⊟ Work Items	☑	☑	☑	☑	☑
⊞ Save Attachment (server)	☑	☑	☑	☑	☑
⊞ Save Category (server)	☑	☑	☑	☑	☑
⊞ Save Release (server)	☑	☑	☑	☑	☑

Evolving and Sharing Processes

- How was the Scrum Template being developed?

- Start with an existing process template.
- Create a project area with the imported template
- Evolve the process as you go
- Customize and consolidate as required
- Create a new process template from the concrete project area
- Export the newly created process template
- Share it with your friends

Process Extensibility

- The set of potential preconditions and follow-up actions is unlimited.
- We include a limited set that is useful out of the box.
- Jazz can be extended with new preconditions and follow-up actions using standard eclipse extension points.

- Real life example:
  - “Externalized Strings” precondition for selfhosting

Summary

- Jazz processes capture the idea and the notion of choreographies of collaboration.
- With Jazz collaboration rules are your friend not something you have to fight. Keep your processes as concrete as possible and as strict as necessary.
- Process sandboxes allow ‘good things’ to happen on all levels.
- Process support in Jazz is an ongoing endeavor