



Work Item and Process Customization

André Weinand
RTC Change Management Architect
andre_weinand@ch.ibm.com

Customizations: Scope

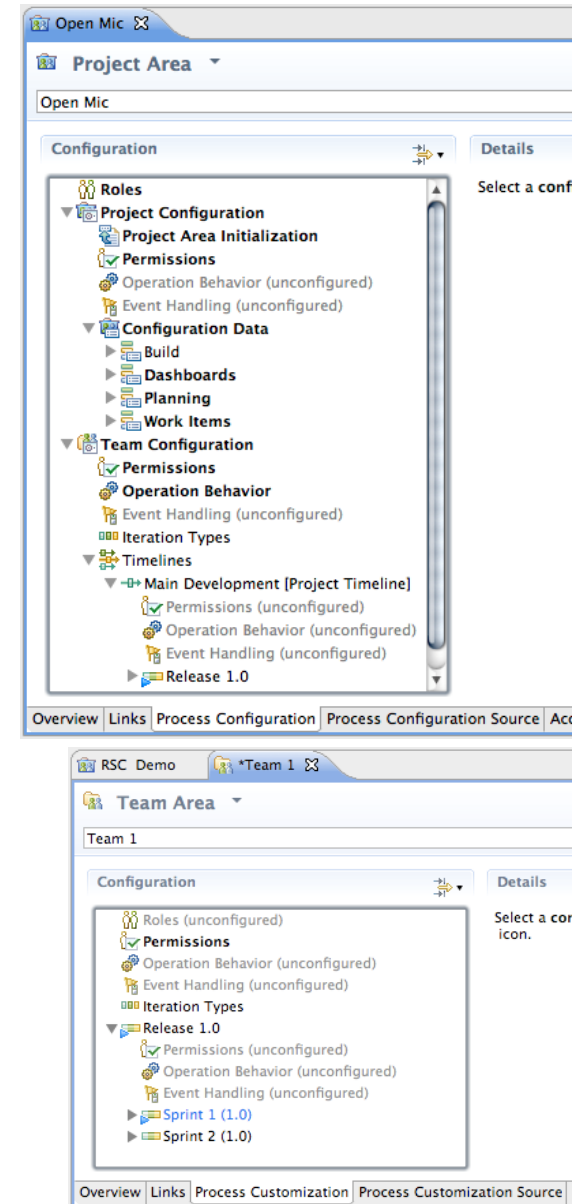
- RTC supports customizations for these capabilities:
 - ▶ Process (Jazz Foundation)
 - ▶ Work Items
 - ▶ Planning
 - ▶ Dashboards
 - ▶ SCM
 - ▶ Build

- Today's focus:
 - ▶ Only common customizations
 - ▶ Components:
 - Process
 - Work Items

Process Customizations

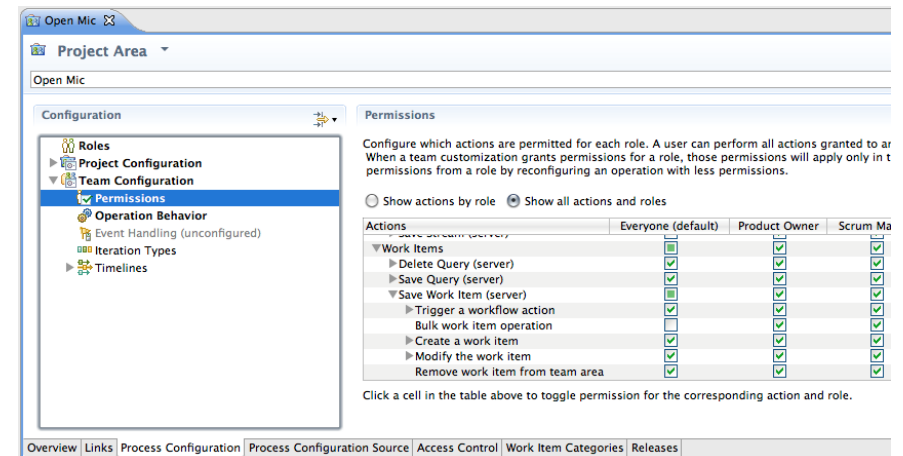
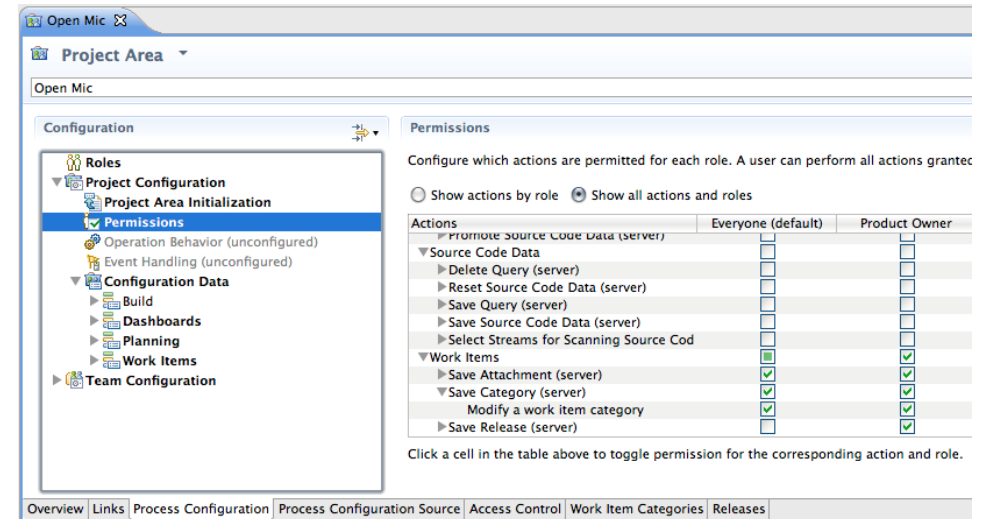
- Project Area:
 - ▶ Roles
 - ▶ Project configurations
 - Initialization
 - Role based Operations
 - Configurations
 - ▶ Team configurations
 - Role based Operations
 - Timeline/Role based Operations

- Team Area (optional):
 - ▶ Roles
 - ▶ Role based Operations
 - ▶ Timeline/Role based Operations



Process Customization: Operations

- **Example:**
 - ▶ “Save Category”: project level
 - ▶ “Save Work Item”: team level
- **Operation Permissions:**
 - ▶ Based on finer grained “actions”:
 - **Create** Work Item
 - **Modify** Work Item
 - **Modify Attribute “Priority”** of Work Item
- **Operation Behavior:**
 - ▶ Preconditions, e.g. “Required Properties”
 - ▶ Follow-up actions, e.g. “Create Initial Work Items”



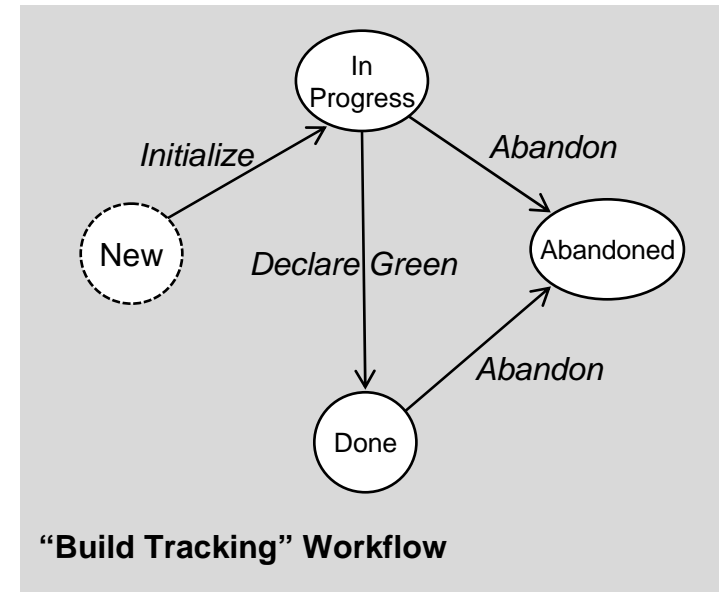
Work Item Types & Custom Attributes

- **Work Item Type** defines:
 - ▶ Display name, Icon
 - ▶ Built-in attributes: e.g. Summary, Description, Owner, Creator, ...
 - ▶ Optional custom attributes
 - ▶ A state transition matrix (aka workflow)
- **Scrum Examples:**
 - ▶ Build Tracking Item
 - ▶ Story
 - Criteria of Acceptance: Large HTML
 - ▶ **Adoption Item**
 - Impact: Custom enumeration
 - Affected Teams: Team Area list

The screenshot shows a web interface for a work item titled "Adoption Item 69". The summary is "Upgrade to 2.0 API". The "Details" section includes a dropdown for "Type" set to "Adoption Item", a dropdown for "Filed Against" set to "Category 2", and a dropdown for "Impact" set to "Significant". The "Affected Teams" section lists "JUnit Team [development]" and "Team 1 [Main Development]" with "Add..." and "Remove" buttons. Below this, the "Team Area" is "Team 1 / Demo9", the "Creation Date" is "May 26, 2009 2:30 PM", the "Created By" is "Andre Weinand", and the "Owned By" is "Unassigned".

Work Item Workflows

- A **Workflow** consist of:
 - ▶ States
 - ▶ Actions
 - ▶ Resolutions (optional)
 - ▶ State Groups
- Scrum Examples:
 - ▶ User Story Workflow
 - ▶ **Build Tracking Workflow**



Customizing the UI – Editor & Attribute Presentations

■ Editor Presentations

- ▶ Define how a work item is presented based on:
 - Tabs, Sections, Attribute presentations
 - Small number of predefined layouts (no free-form layout yet)
- ▶ Can be used in various contexts:
 - Editor, Inline Editors, Dialogs, Hovers, ...

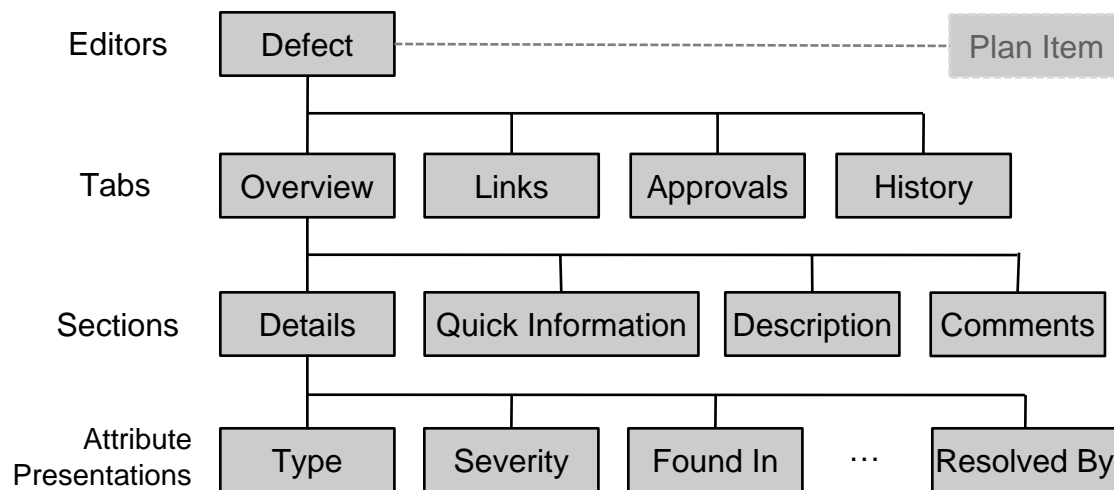
The screenshot shows the 'Types and Attributes' configuration page. The 'Work Item Types' section is expanded to show the 'Defect' type. The configuration fields for 'Defect' are:

- Name: Defect
- ID: defect
- Category: com.ibm.team.workitem.workItemType
- Icon: [Bug icon]
- Aliases: bug

The 'Work Item Editor' field is set to 'com.ibm.team.workitem.editor.default'. Other fields include 'Inline Work Item Editor' (com.ibm.team.workitem.web.inline.default), 'Lightweight Work Item Creation Dialog' (com.ibm.team.workitem.lightweight.editor.section), and 'Plan Editor Preview' (com.ibm.team.apt.planPreview.default). An orange arrow points from the word 'Contexts' to the 'Work Item Editor' field.

Editor Presentation

- Editor's UI element hierarchy:



- RTC 3.0.1 supports WYSIWYG editing
 - but only in Web UI

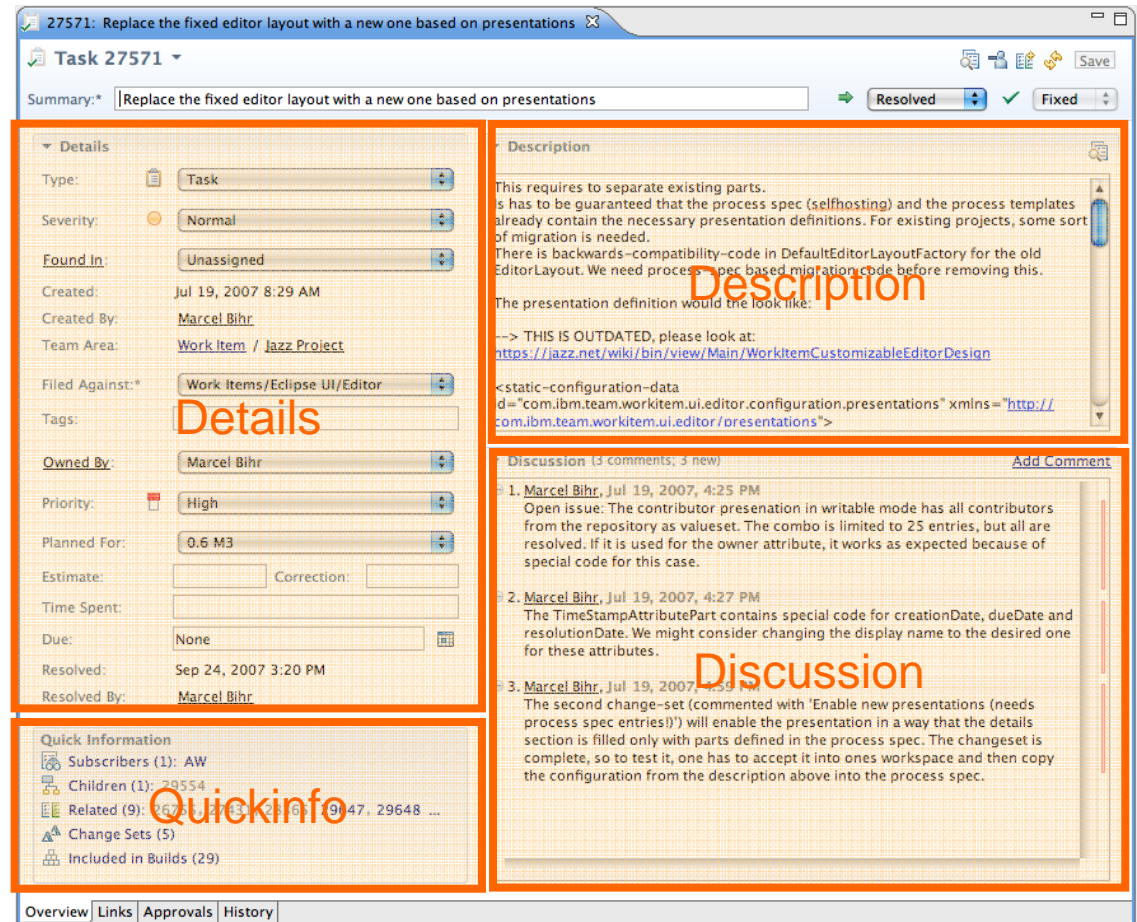


Editor Presentation Layout

- Layout limitations:
 - ▶ Tabs have predefined 'section slots'
 - ▶ Sections can be added to slot but not arbitrarily positioned

- Web UI uses same slots
 - ▶ but different layout

- Similar limitation for Sections
 - ▶ Attributes can be added but not arbitrarily positioned



More Customization Features

- **Preconditions** for Work Item Save Operation:
 - ▶ “Require Approvals”
 - Require an approval for all transitions into a specific state
 - ▶ “Strict Group Resolution” (generalization of “All Children Resolved” precondition)
 - Requires all children to be resolved before the parent can be resolved, and requires the parent to be open before a child can be added or re-opened.
 - ▶ “Attribute Validation”
 - Verifies that a work item can only be saved if all attribute values are valid
 - ▶ “Implied Attributes”
 - Specific attribute changes can trigger sets and clears of dependent attributes
 - ▶ “Prevent Editing”
 - Prevents a Work Item from being edited in certain states

More Customization Features

■ Default Values

- ▶ Freely configurable default value for attributes
 - Supported Types: Text, Wiki text, Users, User Lists, Iterations, Categories, Numbers

The screenshot shows a configuration form with two main sections: 'Details' and 'Configuration'. In the 'Details' section, there is a 'Name' field containing 'Default Description' and a 'Category' field containing 'Multi-Line HTML'. The 'Configuration' section has a 'Text' field containing the message: 'Please see the defect filing guidelines at: <http://jazz.net/guidelines/bugs.html>'.

■ Attribute Validation

- ▶ Two out-of-the-box configurable validators:
 - Number Ranges: Restrict numeric fields to a specific range, e.g. only positive numbers
 - Regular Expressions: Enter a regular expression that validates your string attribute
 - Example: U.S. Zip codes:

The screenshot shows a configuration form for a regular expression validator. It includes the following fields and options:

- Configuration** section:
- Severity:** A dropdown menu set to 'Error'.
- Message:** A text field containing 'Enter e.g. CA 94105'.
- Regular Expression:** A text field containing the regular expression `^[A-Z]{2} \d{5}$`.
- Case sensitive:** A checked checkbox.
- Match line breaks (for ^ and \$):** An unchecked checkbox.
- Test:** A text input field containing 'CA 94115' and a 'Test' button.
- Result:** A message below the test field stating 'The expression matches the text'.

More Customization Features

▪ Dependent Enumerations

- ▶ Value set of one enumeration depends on the value of another enumeration
- ▶ Example:
 - "Browser" value set depends on a "Platform" attribute:

The screenshot shows a configuration window titled "Configuration" with two columns: "Source Enumeration:" and "Dependent Enumeration:".

Under "Source Enumeration:", there is a dropdown menu labeled "Platform" with a list of options: OS X, Windows, and Linux. "OS X" is currently selected.

Under "Dependent Enumeration:", there is a dropdown menu labeled "Browser" with a list of options: Firefox, Safari, Internet Explorer, and Chrome. All four options are checked with blue checkmarks.

Process Customization: Best Practices

- Process can be easily changed at any time
 - ▶ But there are some constraints (see below)
- At end of iteration reflect on how to improve process
 - ▶ e.g. collect improvements in “Process Improvement” work items
- After enough reflection/discussions customize existing process instance
 - ▶ Start with simple customizations
 - e.g. **add** a new work item type, **add** query to find all items of this type
 - ▶ Plan how to avoid corruption of existing data
 - e.g. **add only**, do not change IDs,
do not remove process definitions that are still used
(use the “check ... usages” action links in the Eclipse UI)
- After thorough testing create a process template of your customizations that other projects can use