Rational Team Concert

Introduction to Rational Team Concert

- Rational Team Concert:
  - Is a product that is based on Jazz for developers, architects, and project managers
  - Enables team members to collaborate with integrated iteration planning, work-item management, source control, build management, dashboards, and reports
  - Supports process control and customization
  - Has an Eclipse-based workbench and a Web interface
  - Is a key component of the Rational solution for Collaborative Lifecycle Management (CLM)

Project and team areas

- A project administrator or team lead sets up a project area in Rational Team Concert:
  - Creates a project area and optional team areas
  - Defines the process for this project
  - Defines the project iterations and plans
  - Defines the teams
  - Creates work items

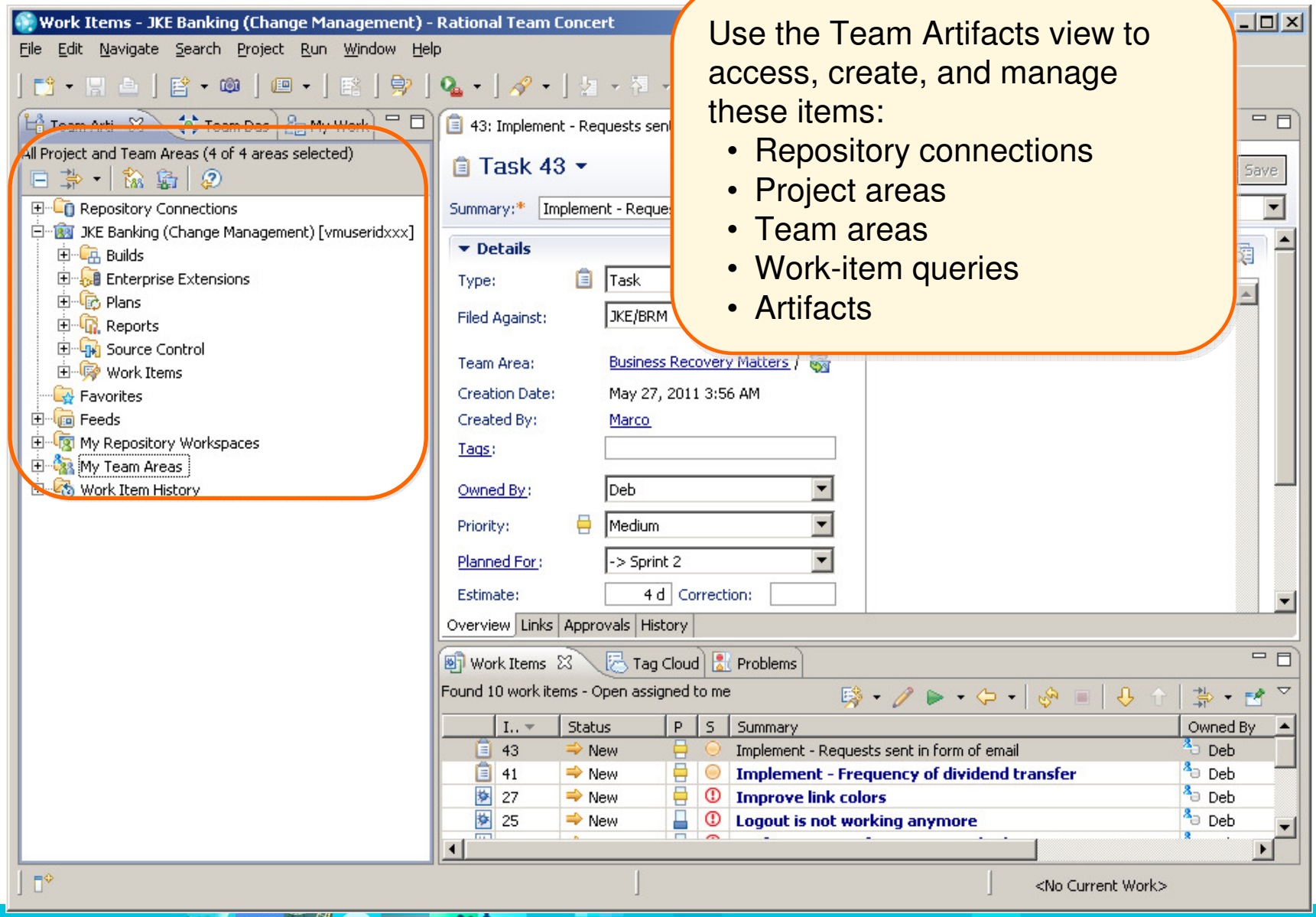
Work items and Iteration plans

- **Work items** capture planned work for a project:
  - Describe requirements, defects, and feature improvements
  - Identify other tasks that are related to project development
- The types of work items available in a project area or team area are defined in the process configuration.
- The project administrator or team lead organizes work into a series of iterations, called an iteration plan.
  - An iteration is generally bound to a specific time period and has a well-defined scope of work items to completed.
  - The process can be adjusted, based on iteration plans. Additional approvals might be required to introduce a new feature late in the development cycle.

Rational Team Concert workbench

- The Rational Team Concert workbench refers to the Eclipse-based desktop development environment.
- The Workbench consists of these elements:
  - **Perspectives**, a group of views and editors
  - **Views**, the element that you use to navigate a list or hierarchy of information, or display properties for the active editor
  - **Editors**, the element that you use to edit or browse a resource

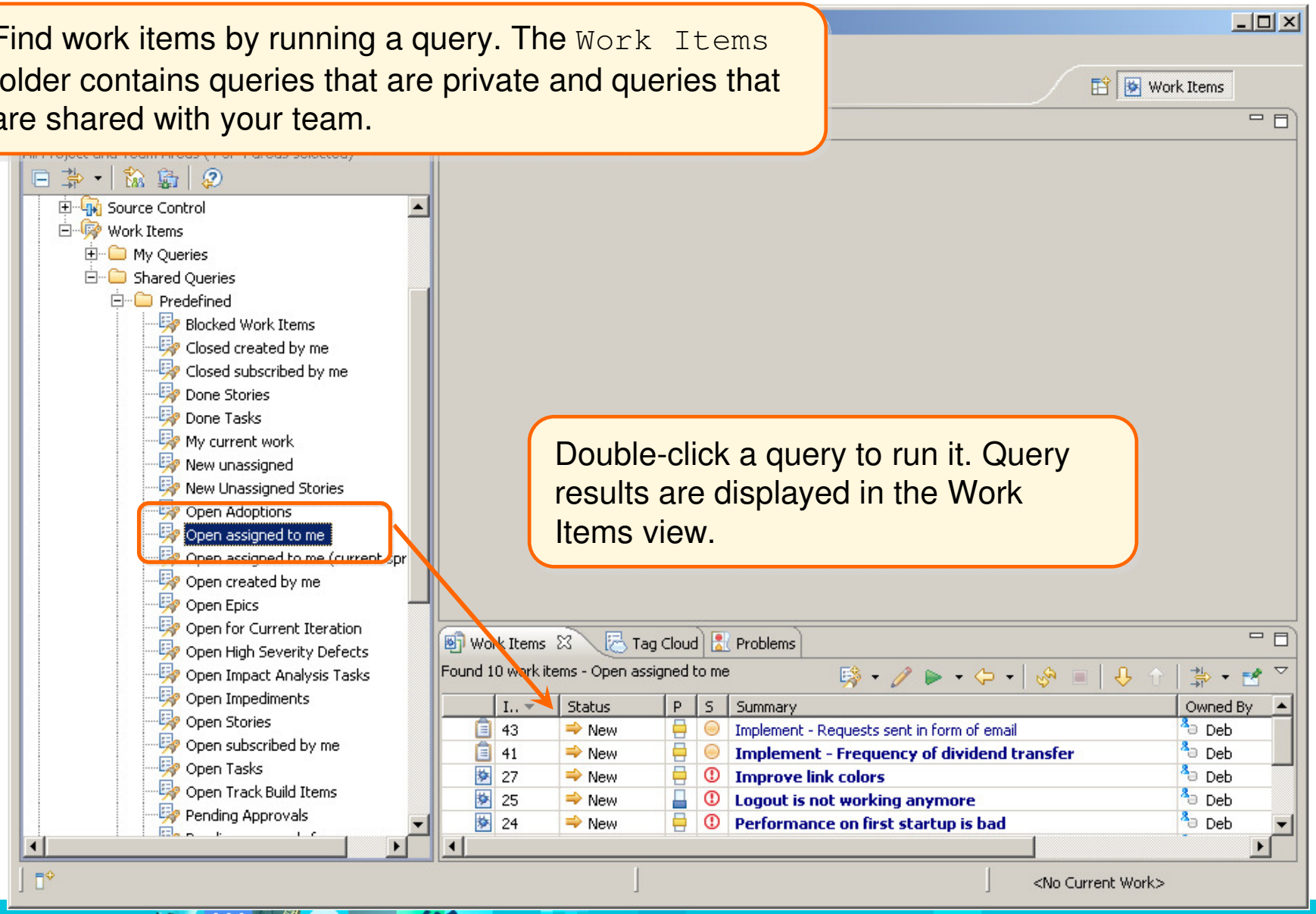
Team Artifacts view



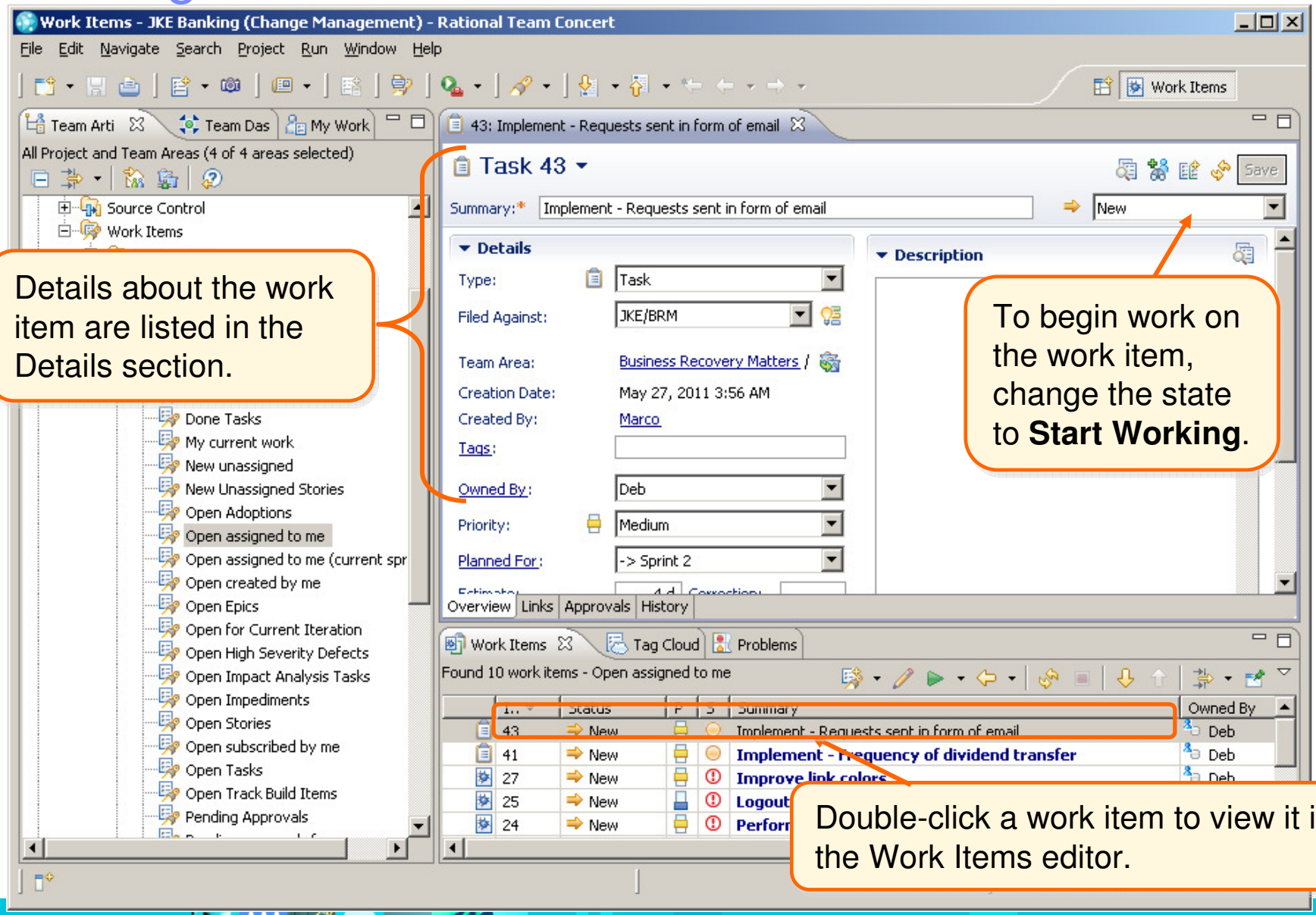
Use the Team Artifacts view to access, create, and manage these items:

- Repository connections
- Project areas
- Team areas
- Work-item queries
- Artifacts

Find work items by running a query. The `Work Items` folder contains queries that are private and queries that are shared with your team.

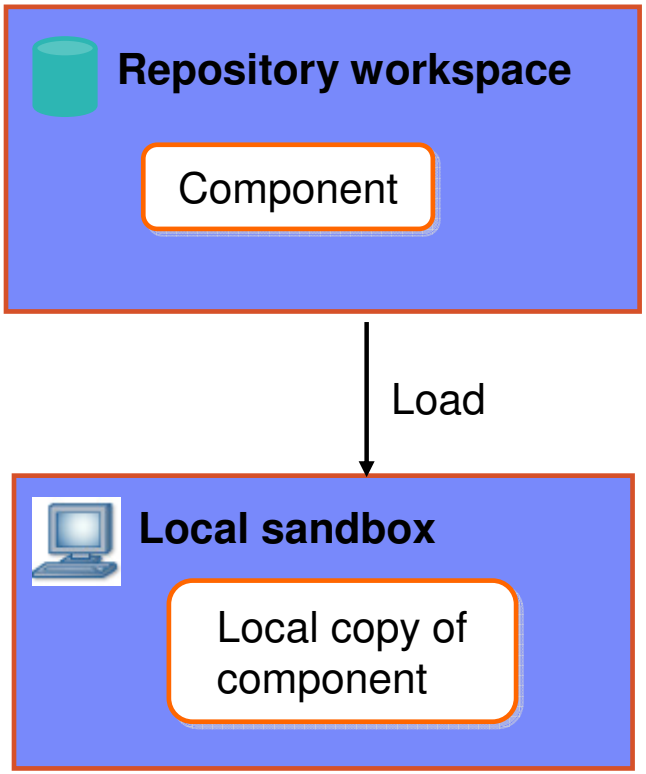


Viewing a work item



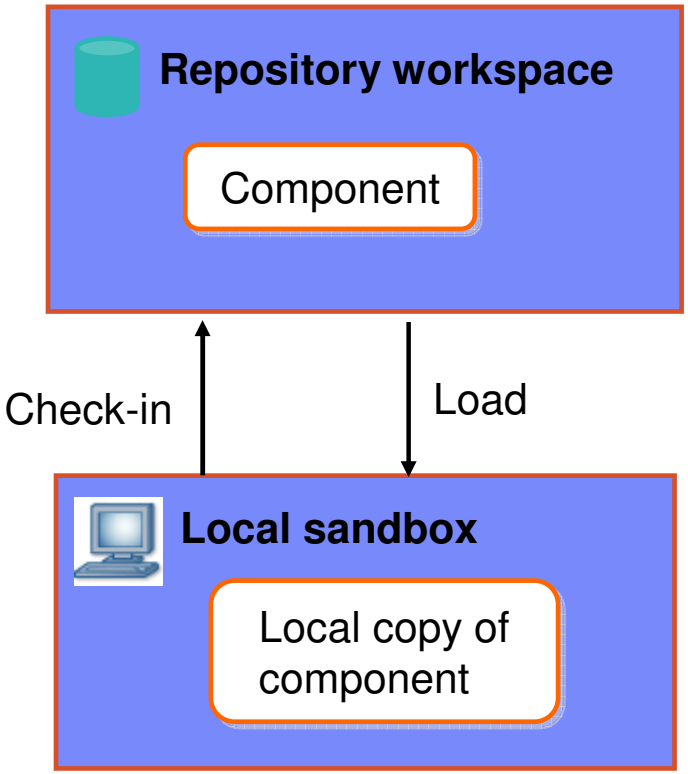
Workspace

- To work on a project, you must have a local **sandbox**, which is a directory in your computer file system where you can work with files and folders under source control. (In an Eclipse environment, this is the Eclipse workspace.)
- You load, or copy, files and folders from a repository workspace on the server to your local sandbox.

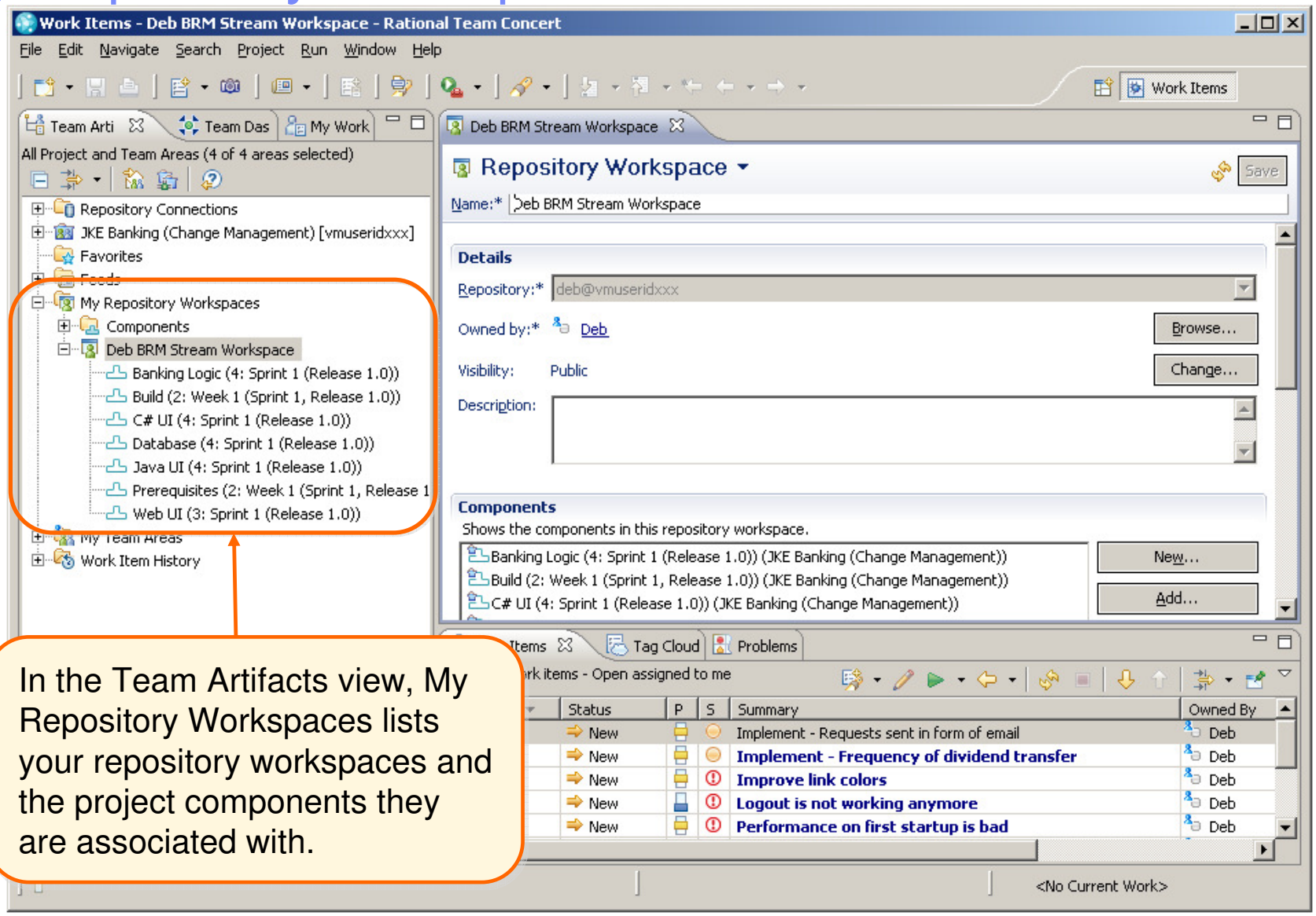


Workspace (continued)

- Changes you make in your local sandbox are stored only on your local file system
- When you check in changes, the modified files and folders are copied to your personal repository workspace on the Jazz Team Server

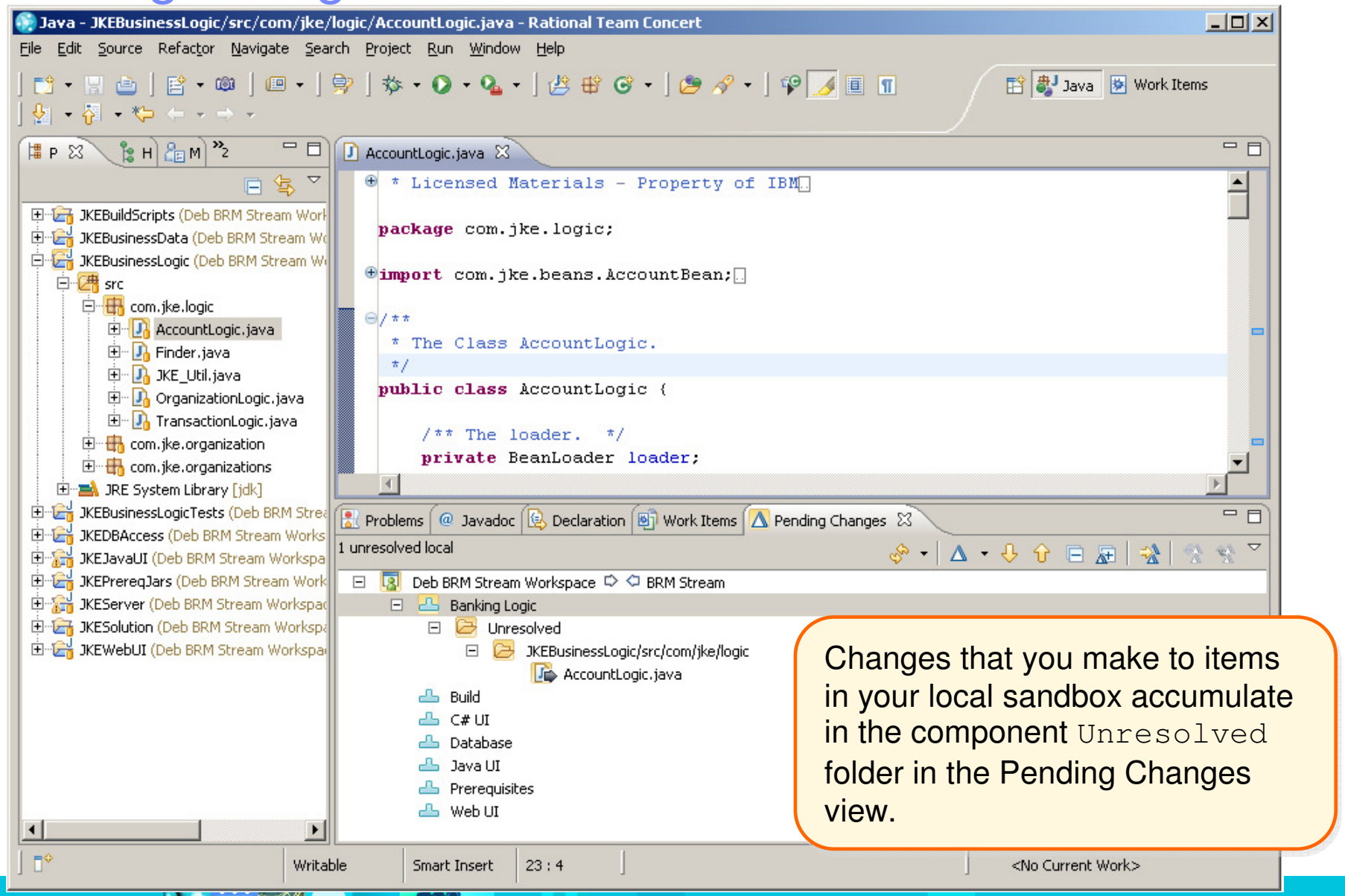


My Repository Workspaces



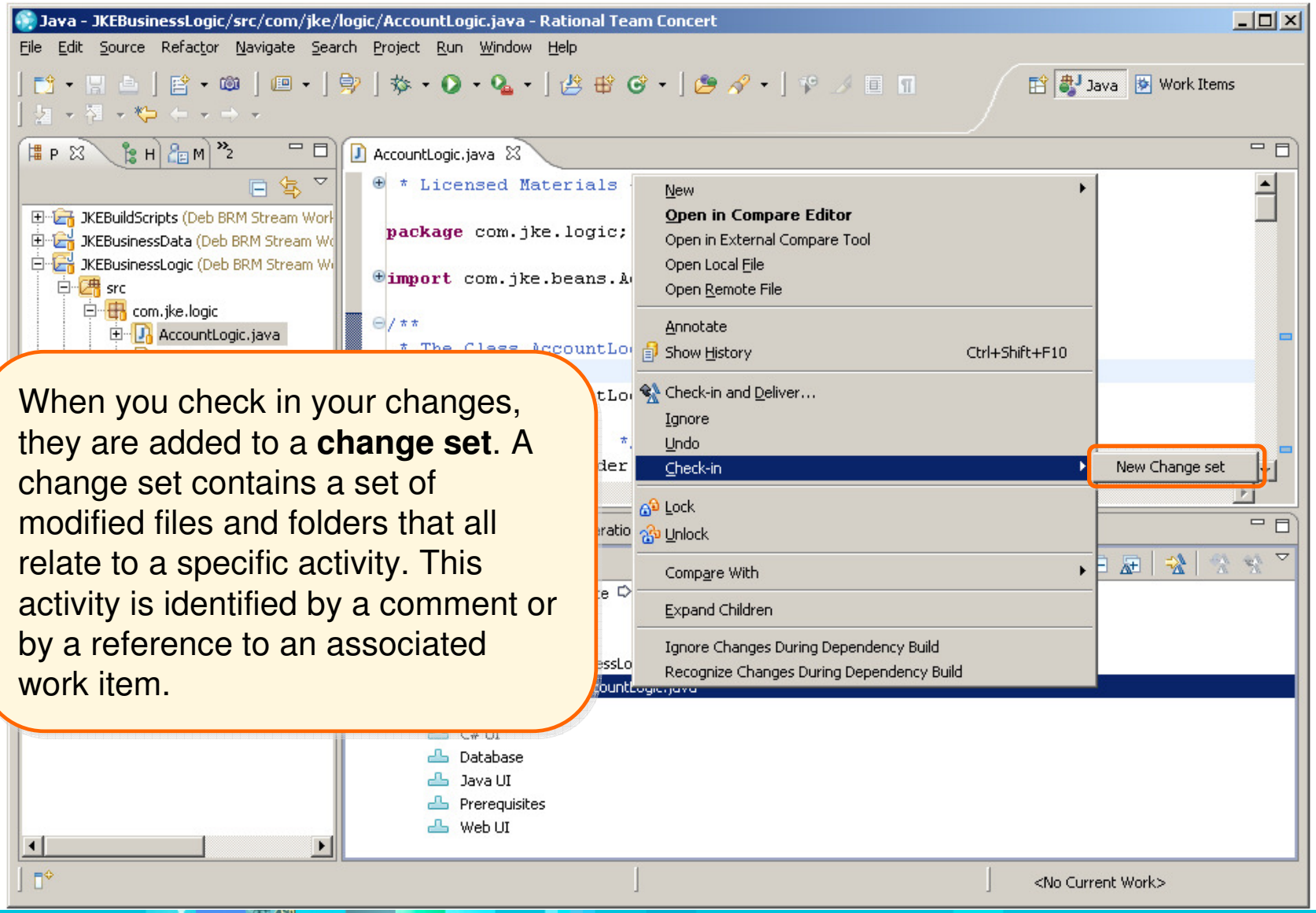
In the Team Artifacts view, My Repository Workspaces lists your repository workspaces and the project components they are associated with.

Pending changes



Changes that you make to items in your local sandbox accumulate in the component `Unresolved` folder in the Pending Changes view.

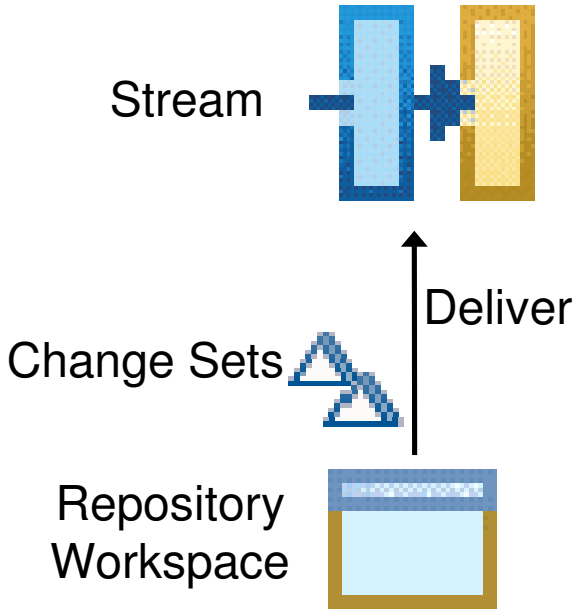
Checking in changes



When you check in your changes, they are added to a **change set**. A change set contains a set of modified files and folders that all relate to a specific activity. This activity is identified by a comment or by a reference to an associated work item.

Delivering changes to a stream

- Checked-in changes are not shared with the team until you **deliver** the change sets from your repository workspace to the team's stream.
- A **stream** is a repository object that is used to integrate the work done in developer's workspaces

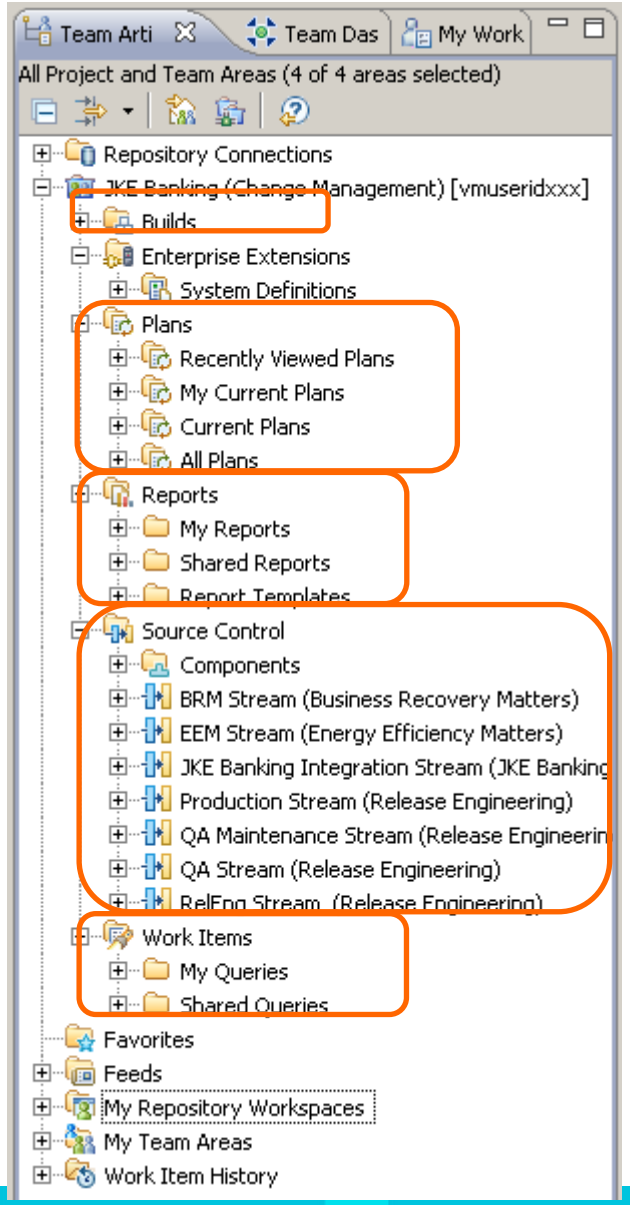


Rational Team Concert user interfaces

- Rational Team Concert client for Eclipse
- Rational Team Concert client for Microsoft Visual Studio
- Rational Team Concert web client
- “scm” command-line interface

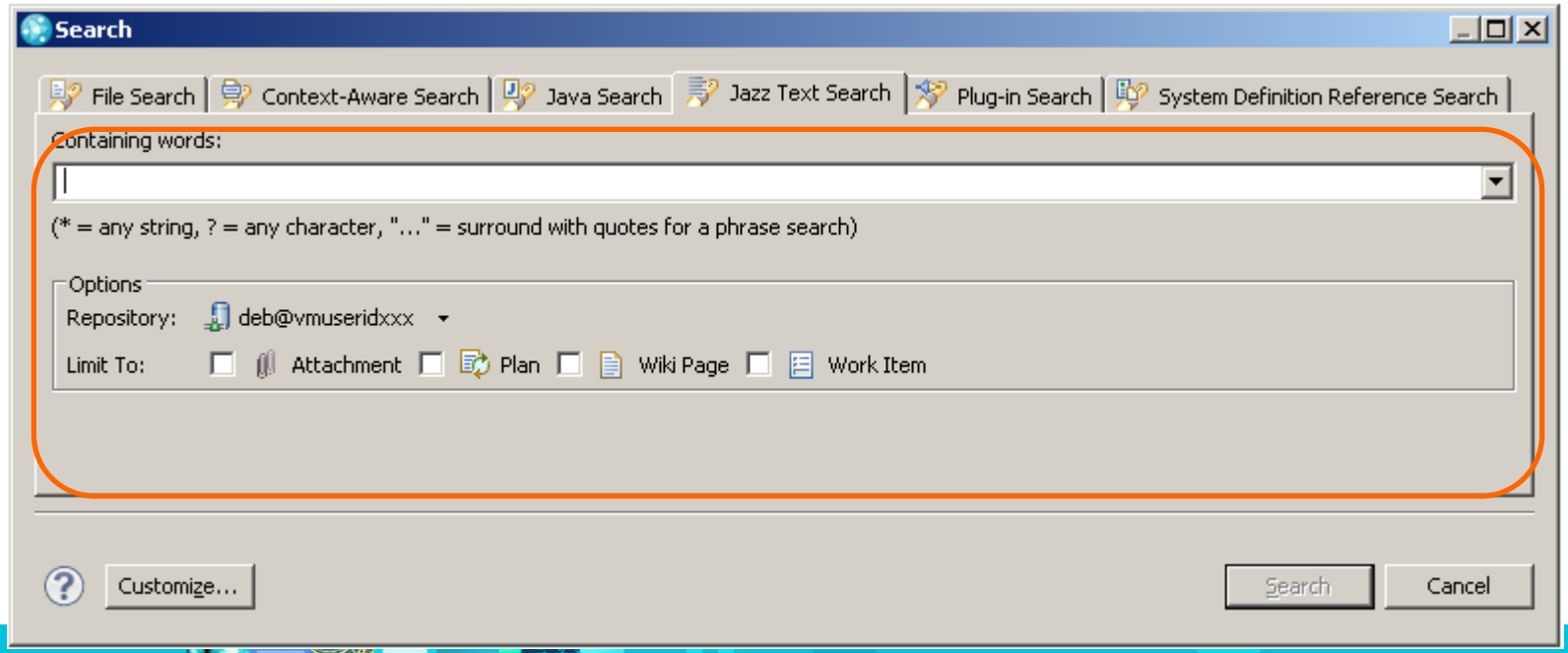
Jazz/RTC capabilities in the Eclipse client

- Artifact management
  - Build management
  - Project planning
  - Reporting
  - Source control
  - Work-item management



Jazz and Rational Team Concert capabilities in the Eclipse client

- Repository search
  - Search an entire repository
  - Limit a search to these elements:
    - Attachments
    - Plan items
    - Wiki pages
    - Work items

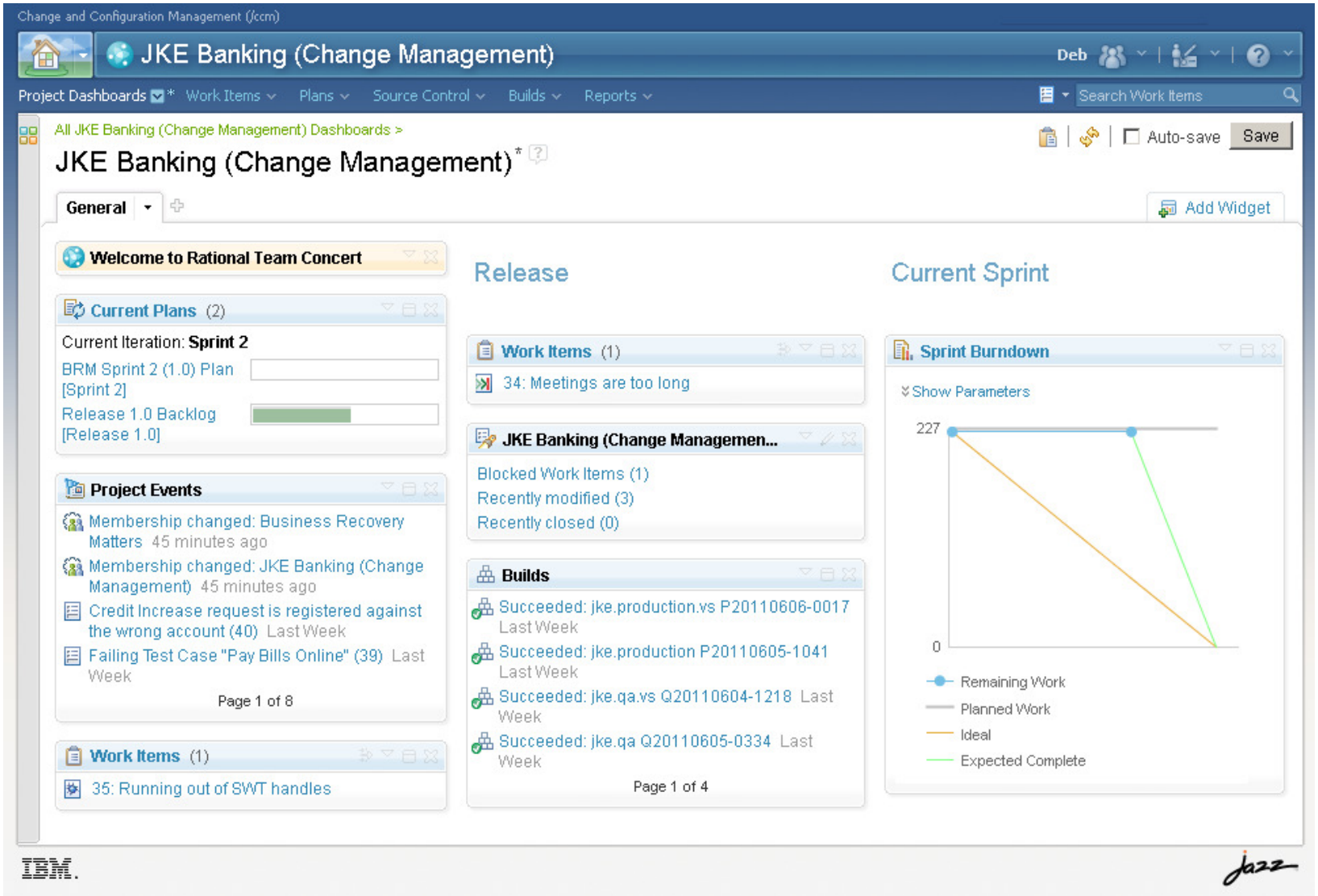


Rational Team Concert web client

- Ideal for project managers, stakeholders, and other contributors who are not using a supported integrated development environment
- Provides access to these capabilities:
  - Dashboards
  - Work-item management
  - Project planning
  - Source control
  - Build management
  - Reporting

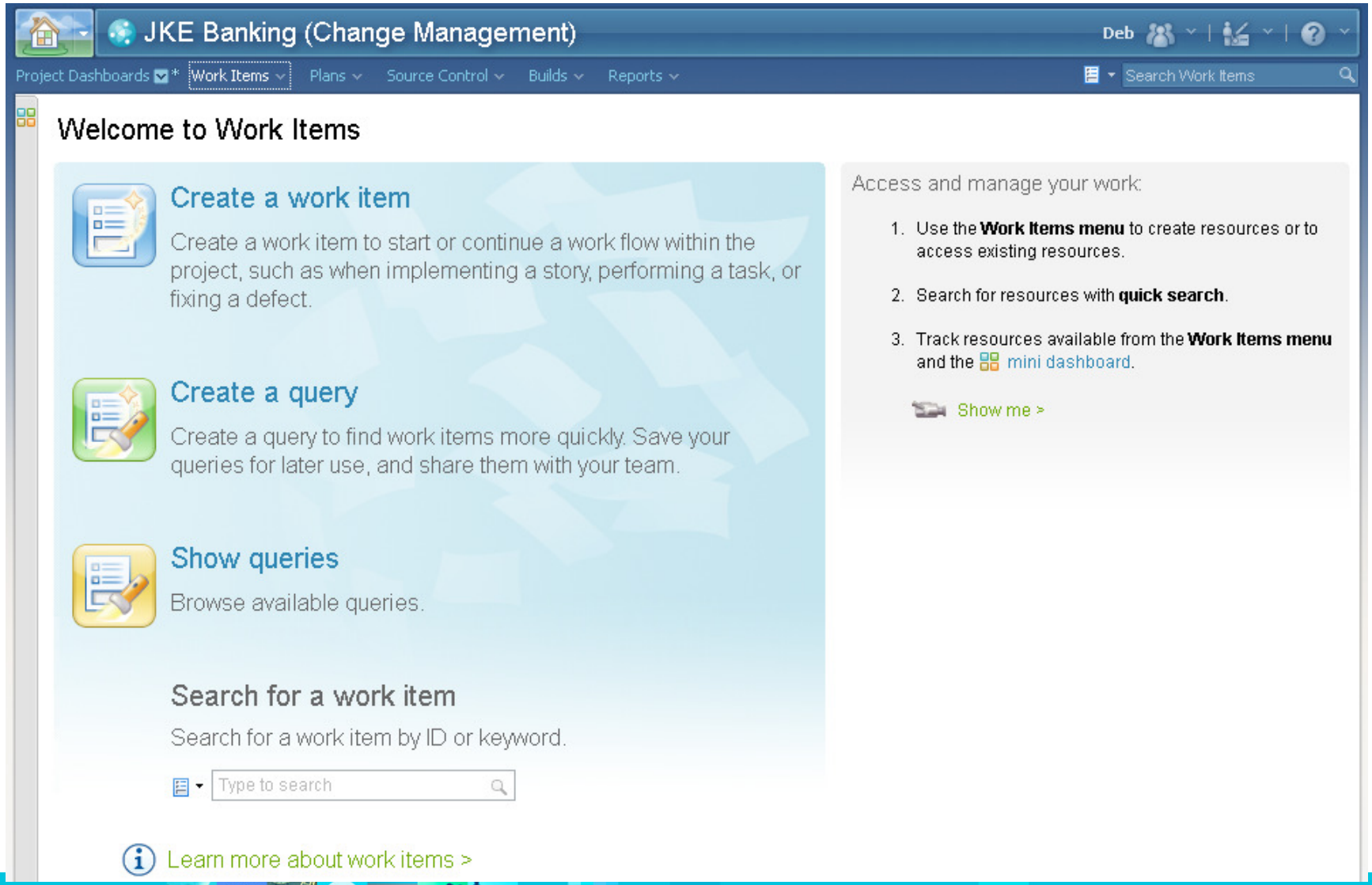
Jazz and Rational Team Concert capabilities in the web client

- Dashboards



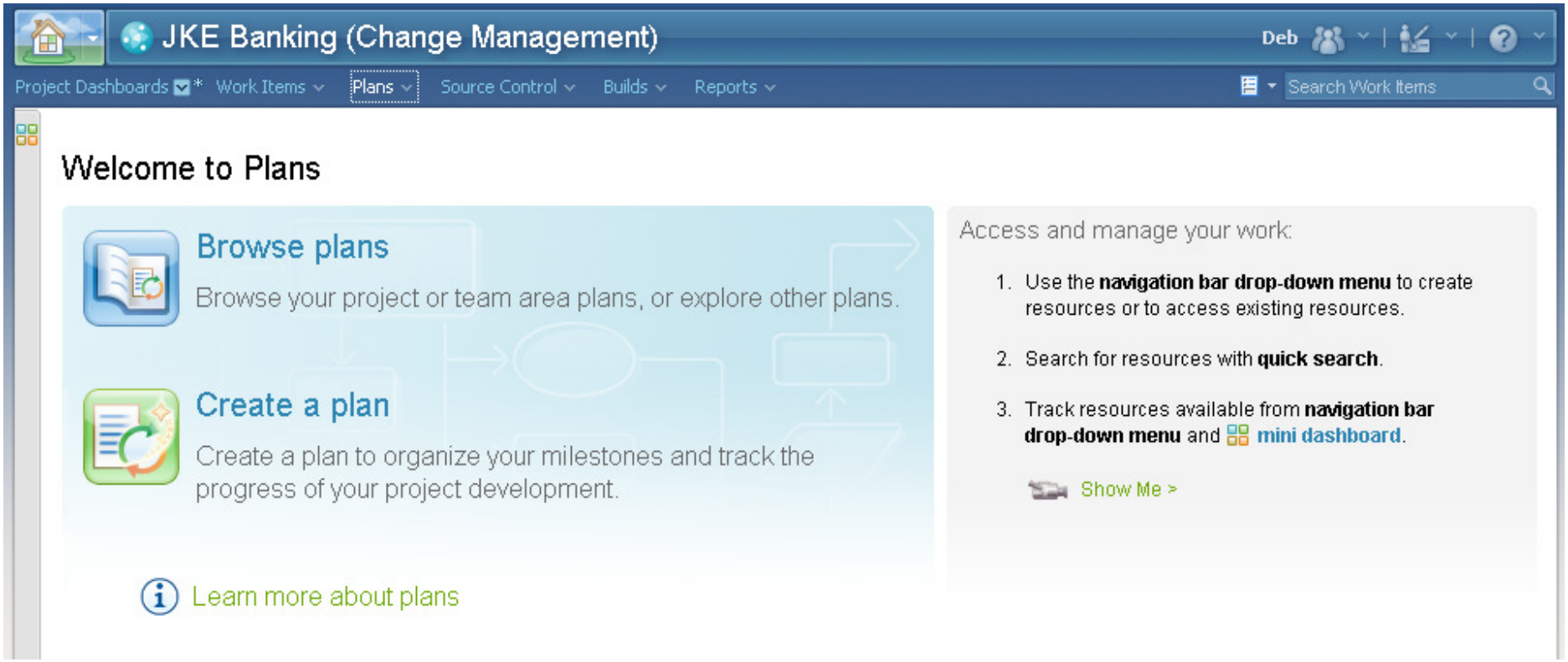
Jazz and Rational Team Concert capabilities in the web client

- Work-item management



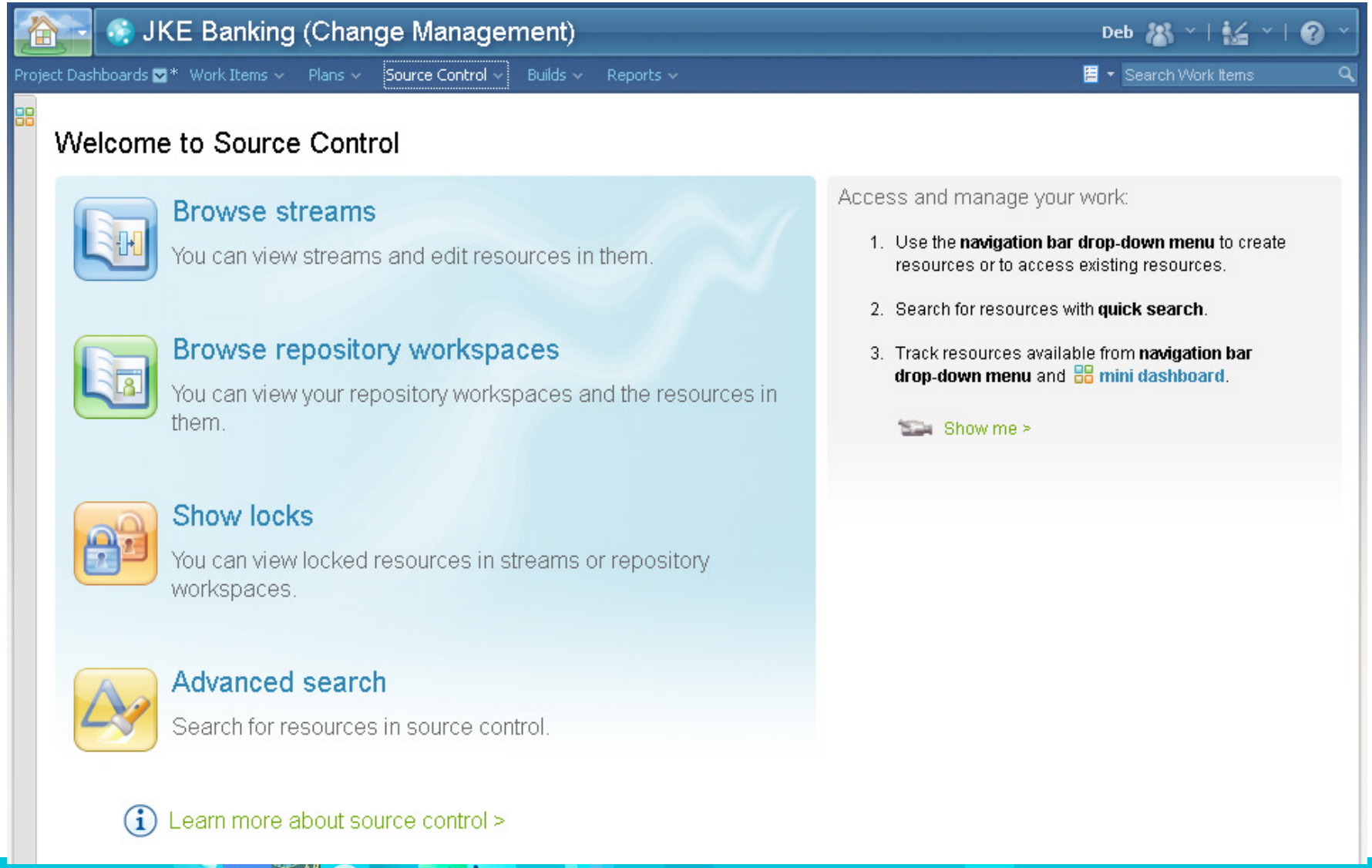
Jazz and Rational Team Concert capabilities in the web client

- Project planning



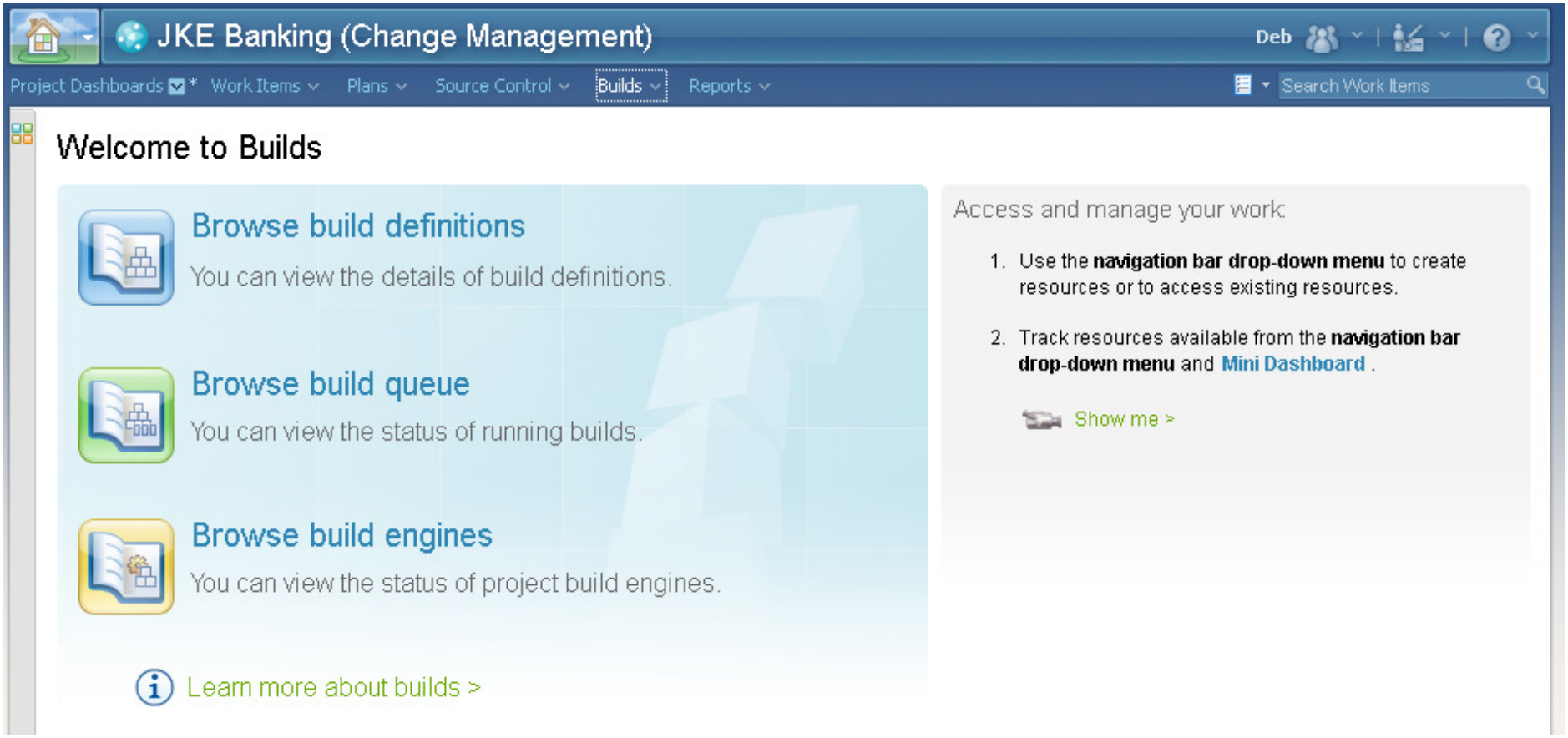
Jazz and Rational Team Concert capabilities in the web client

- Source control



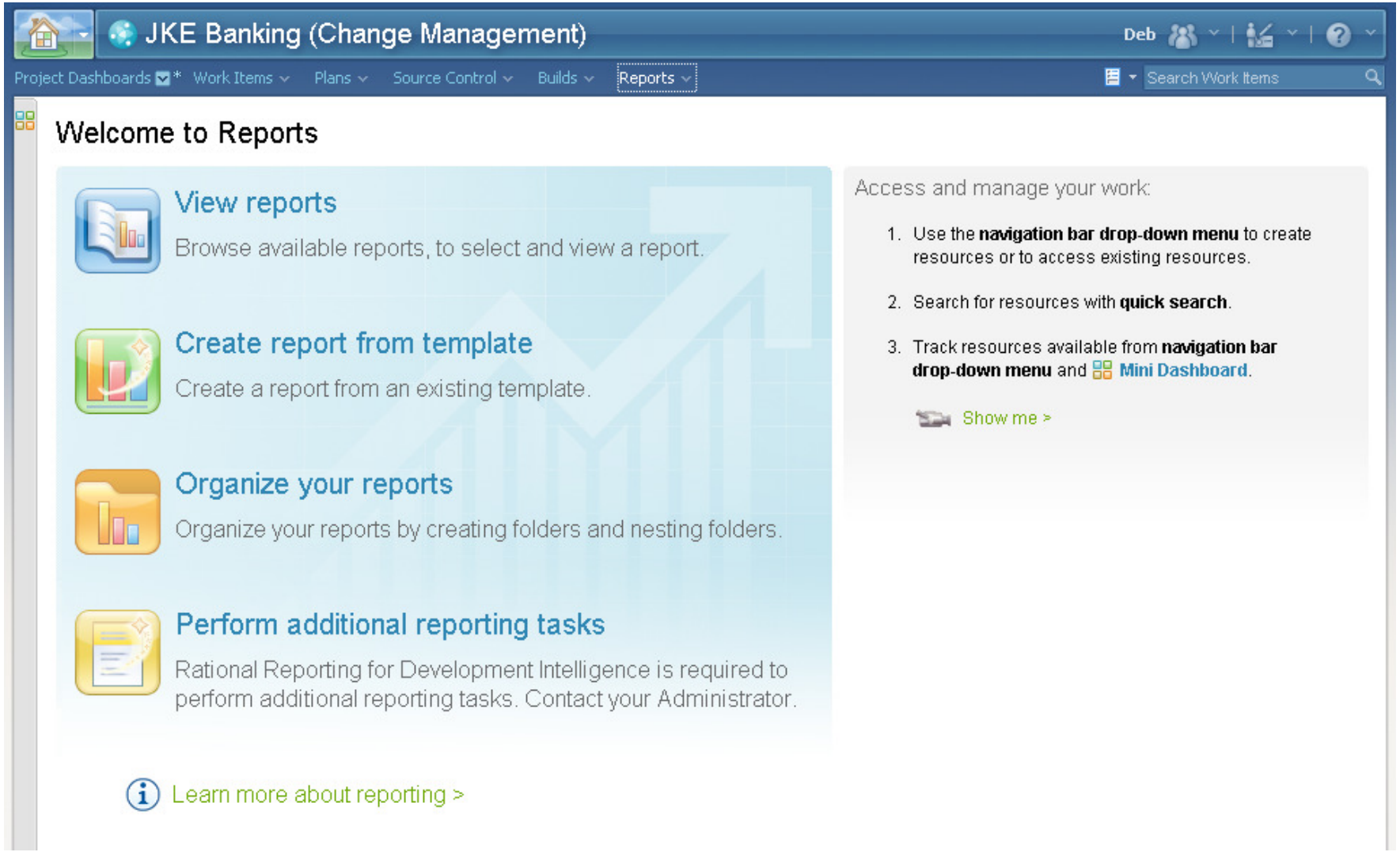
Jazz and Rational Team Concert capabilities in the web client

- Build management



Jazz and Rational Team Concert capabilities in the web client

- Reporting



Work items

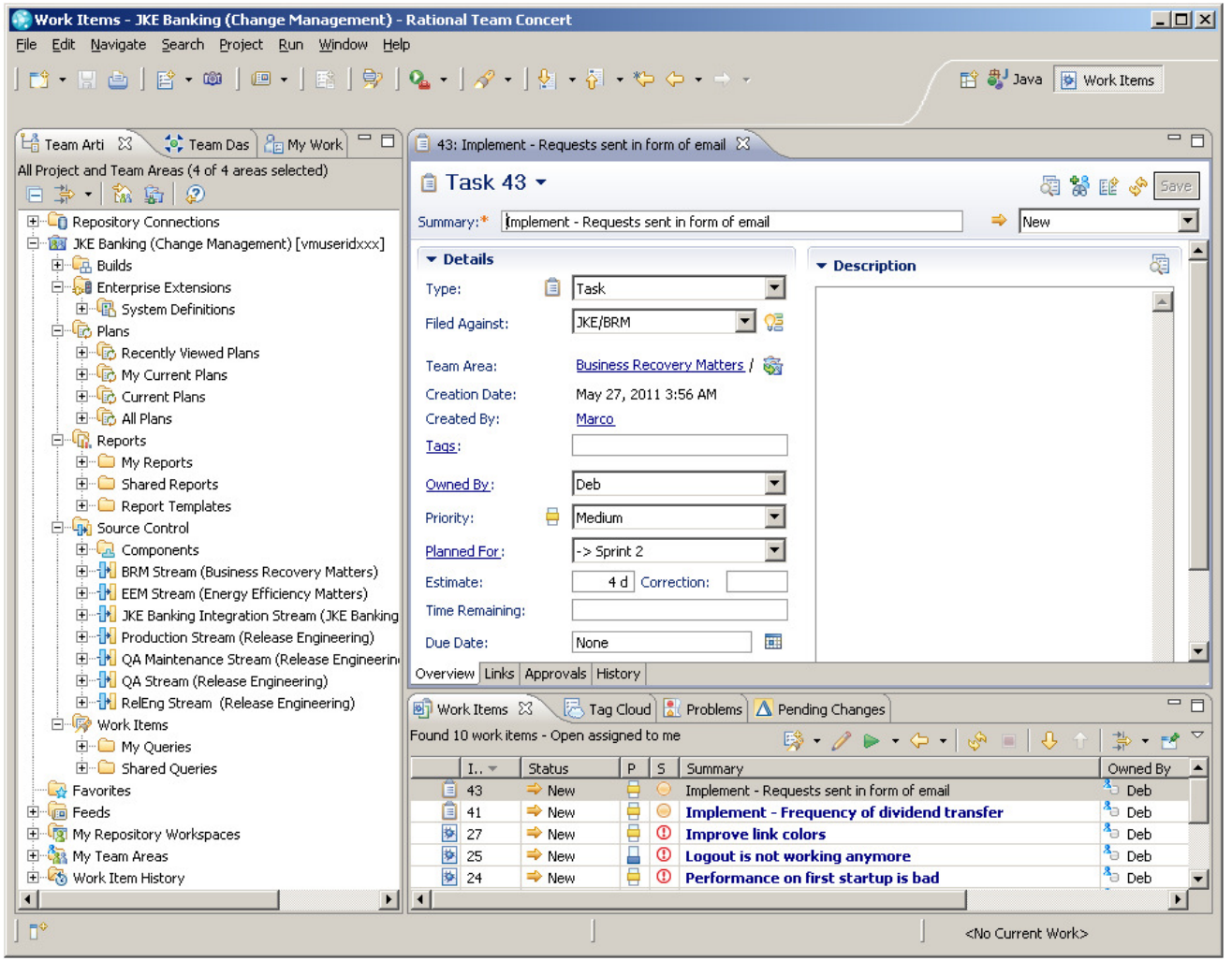
- Work items capture planned work for a project.
- Different types of work items capture different kinds of work:
  - **Defect:** Identifies a bug
  - **Enhancement:** Describes a requested new feature
  - **Task:** Describes a specific piece of work
  - **Plan Item:** Provides a high-level description of work that is targeted for a specific iteration
  - **Story:** Describes part of a use case
- Typical work-item attributes:
  - Workflows and status
  - Properties and values
  - Approvals
  - Attachments

Additional work item capabilities

- Dynamically create relationships to other work items
  - Establish parent-child relationships
  - Specify dependencies
  - Find duplicates
  - Create references
- Use work items to help you complete these tasks:
  - Associate work items with change sets.
  - Allocate work items to plans and iterations.
  - Estimate the effort that is required to resolve the work item.
  - Provide metrics regarding project health.

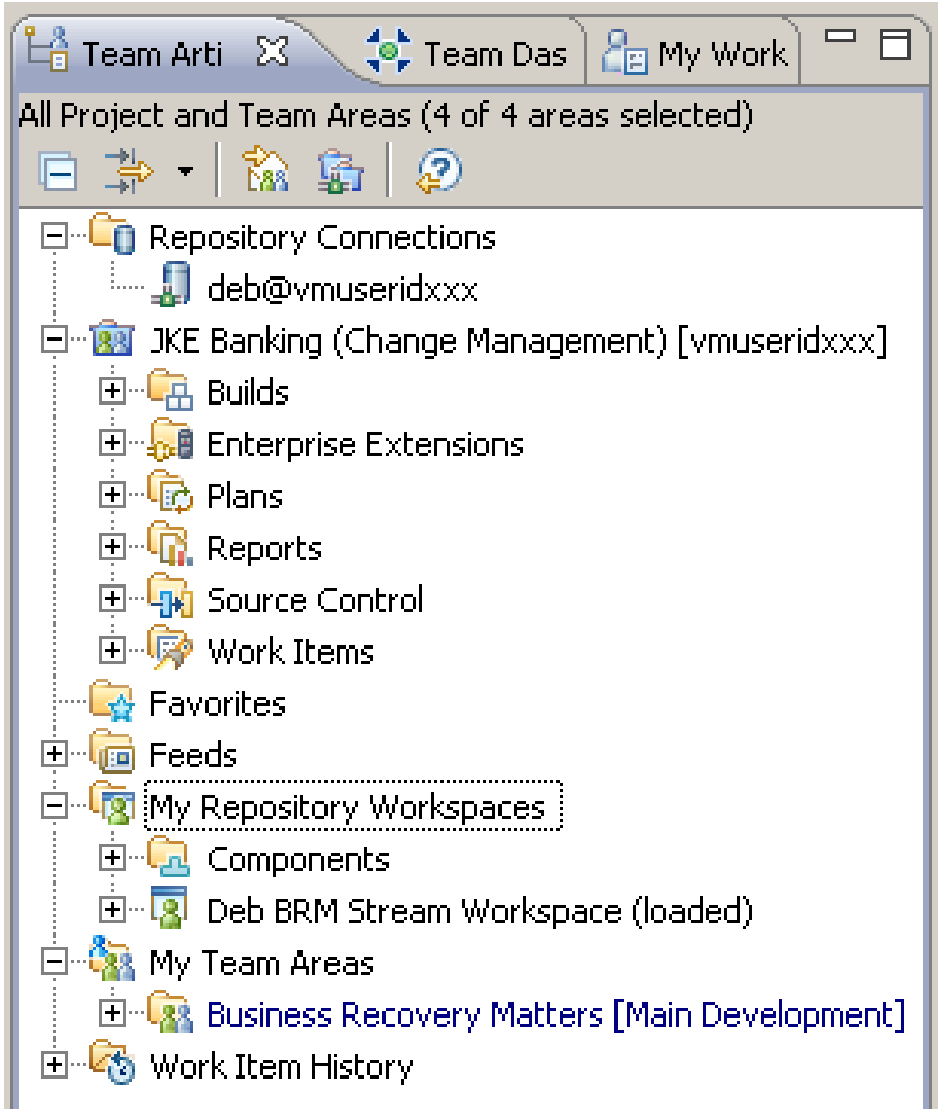
The Work Items

- In the Work Items perspective, you can access and manage your work items. The perspective has these views:
  - Team Artifacts
  - My Work
  - Team Dashboard
  - Work Items



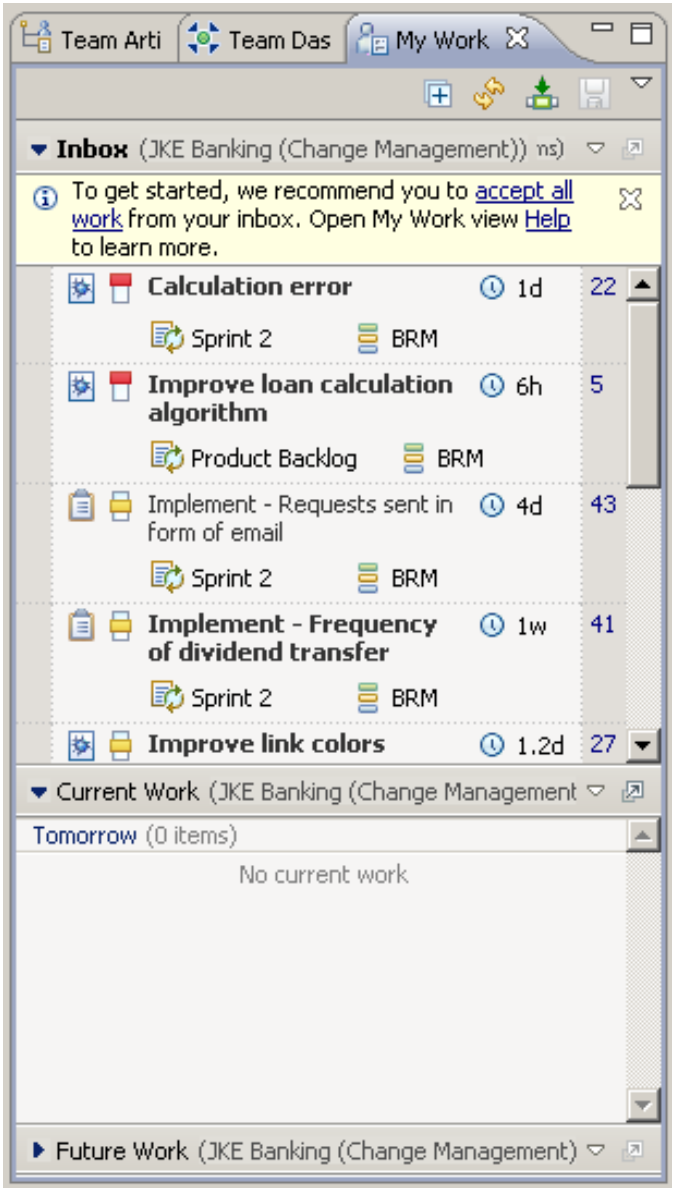
Team Artifacts view

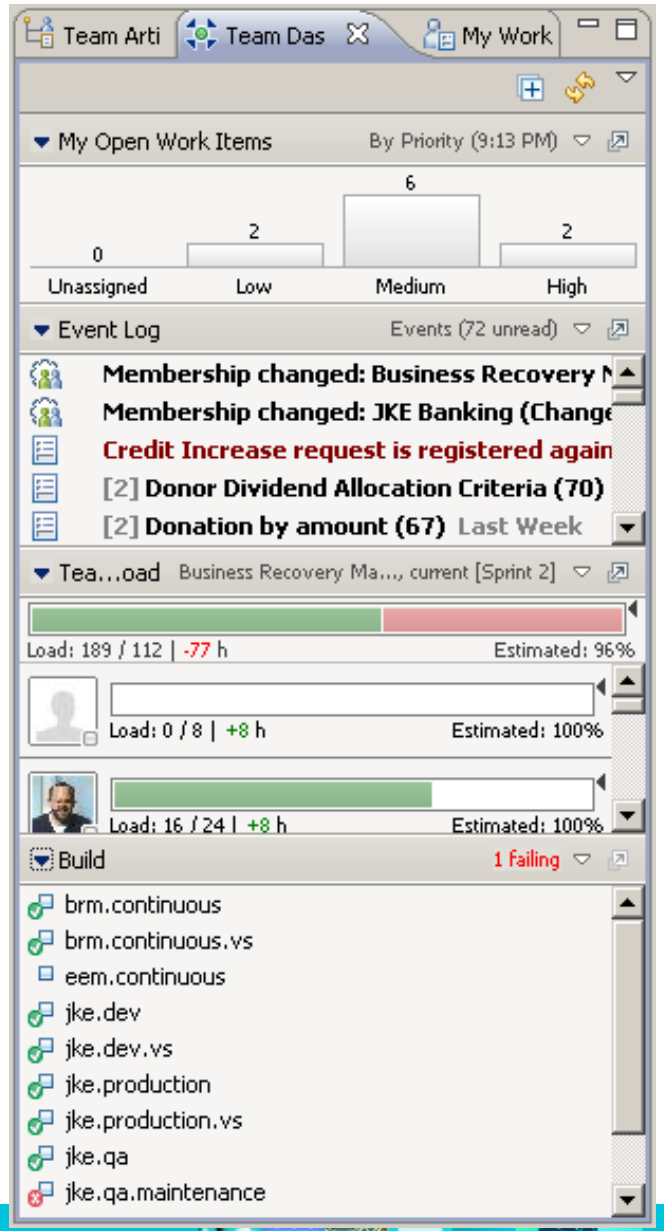
- This view is the central access point for project data.
- The view is organized by these folders:
  - Your repository connections
  - Project and folder hierarchies
  - Your personal folders
    - Feeds
    - Repository workspaces
    - Teams areas that you are assigned to
    - Work Item history



My Work view

- Use this view to manage your work items.
- This view is organized by sections:
  - **Inbox** lists newly assigned work.
  - **Current Work** lists work items for the current iteration.
  - **Future Work** lists work for future iterations.
- Changes that you make in the My Work view can directly impact work items and your teams plans.
- You can customize this view by clicking actions in the **view** menu: Click the **View Menu** icon.



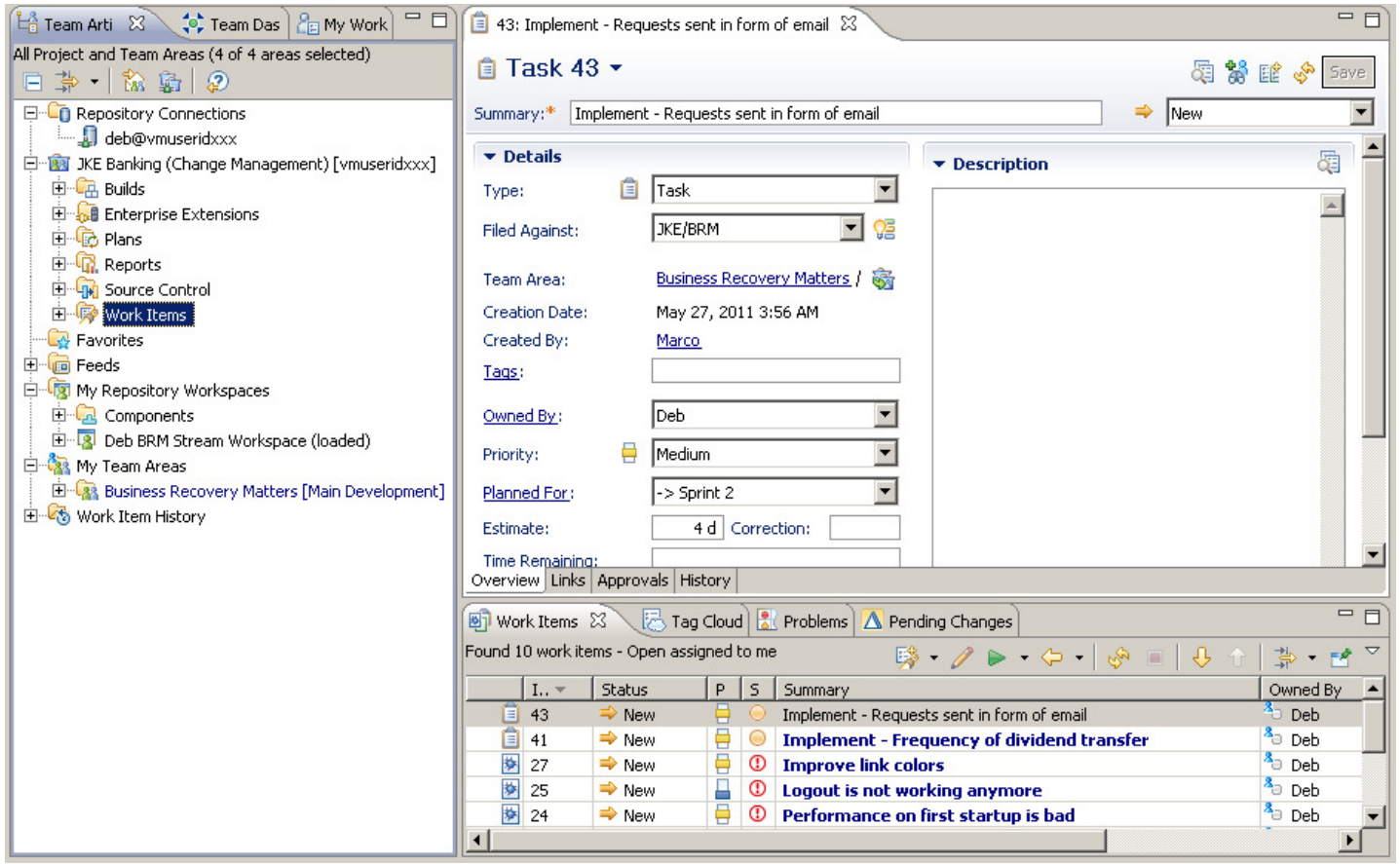


Team Dashboard view

- This view is a central access point for team communications and news.
- The view is continuously updated.
- The view includes these default sections:
  - **My Open Work Items** lists open work items by Priority
  - **Event Log** displays information from news feeds.
  - **Team Load** shows workloads by member.
  - **Build** lists events for builds.
- You can customize this view by using **view** menu.

Work Items view

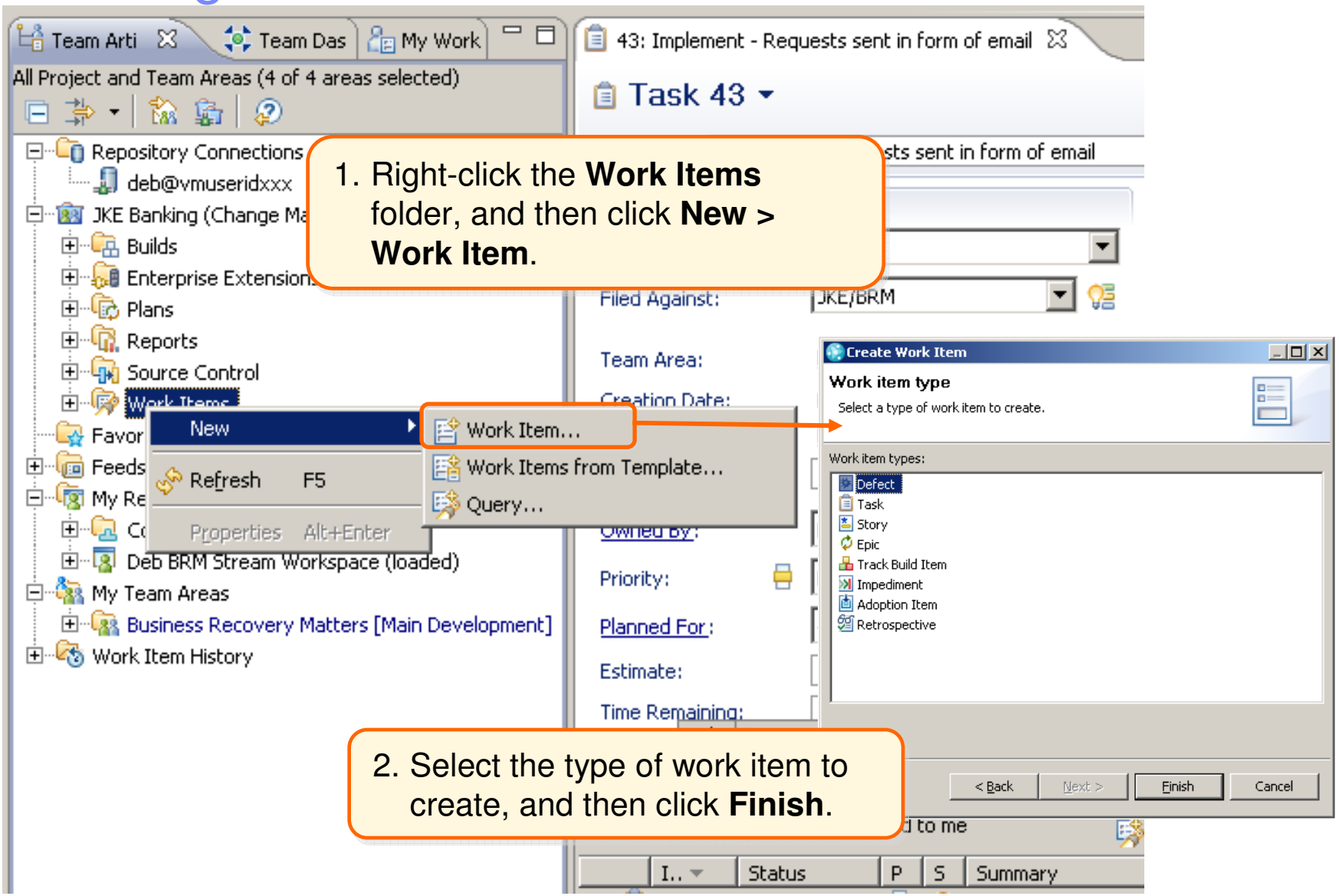
- This view displays work- item query results in a table.
- Double-click a work item to open it in the Work Item editor.



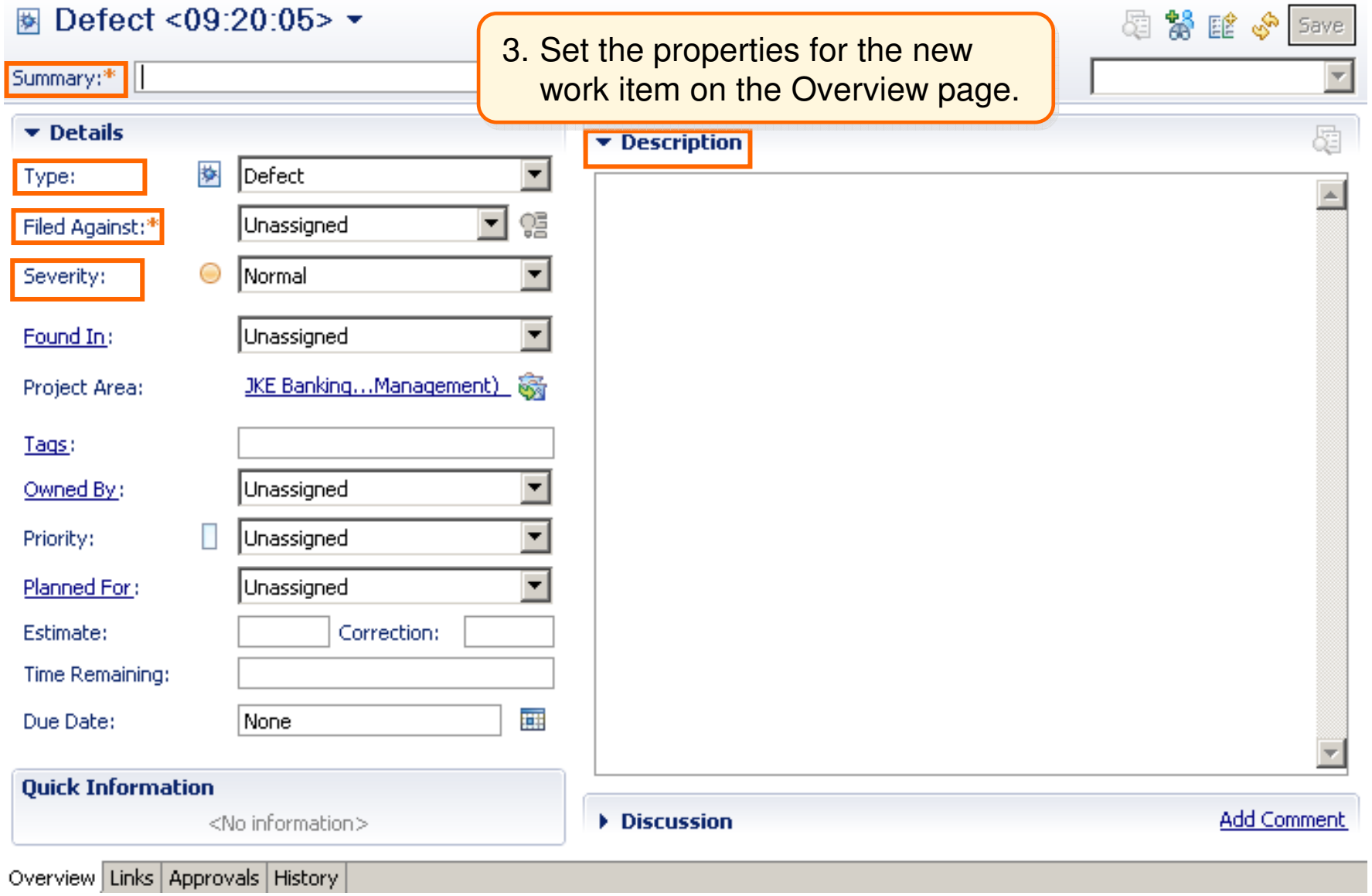
Ways to create a work item

- You create work items to submit a defect, task, enhancement, or other type of work request for a project.
- There are several ways to a create work item:
  - Click **Work Items > New > Work Item**.
  - Duplicate an existing work item.
  - Create a work item whose **Summary** field contains text that you select from text in the **Summary**, **Description**, or **Discussion** fields in the Work Item editor
  - Take a screen capture, and attach it to a new work item.
  - Create a work item in the Build editor.

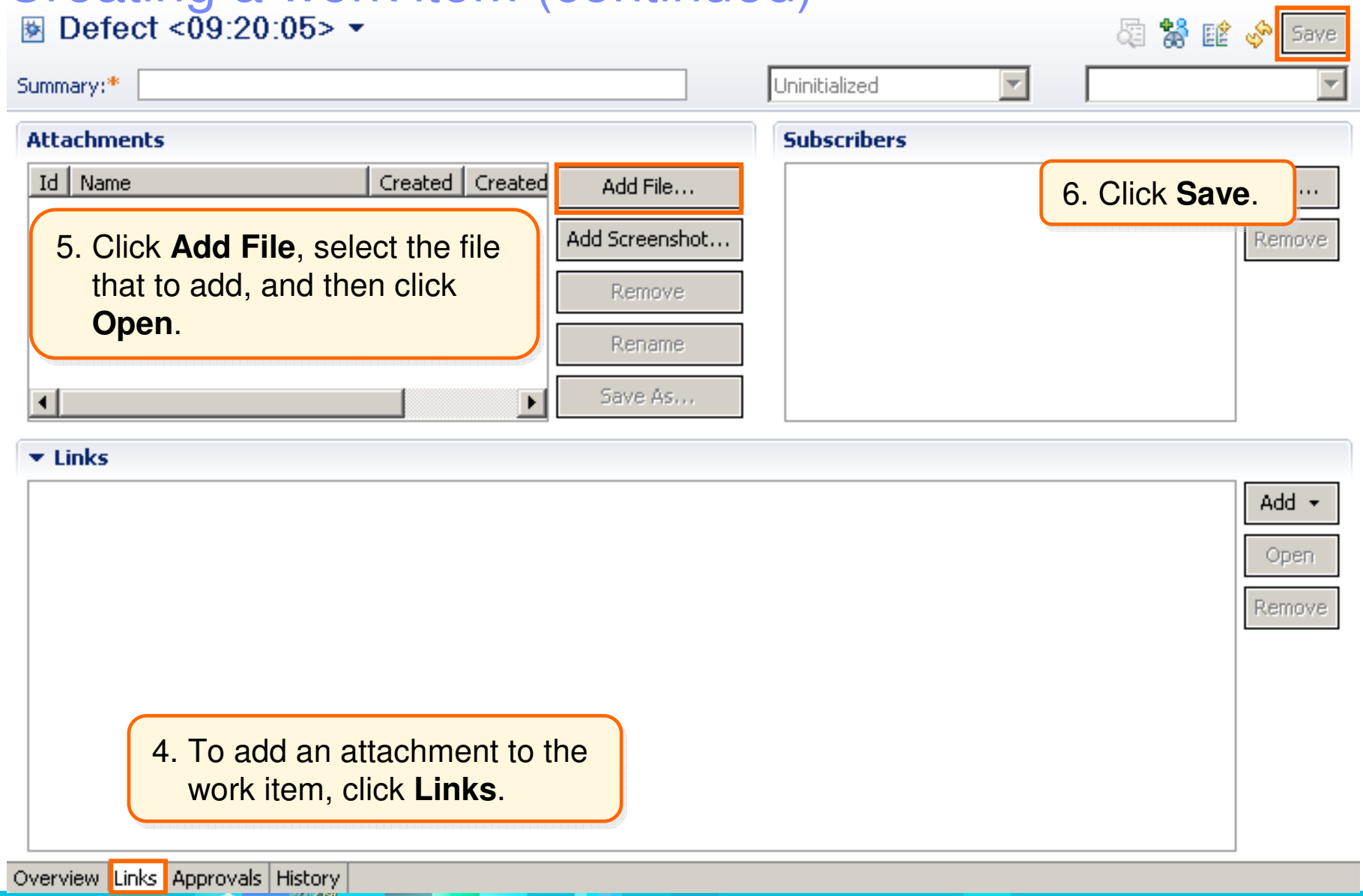
Creating a work item



Creating a work item (continued)

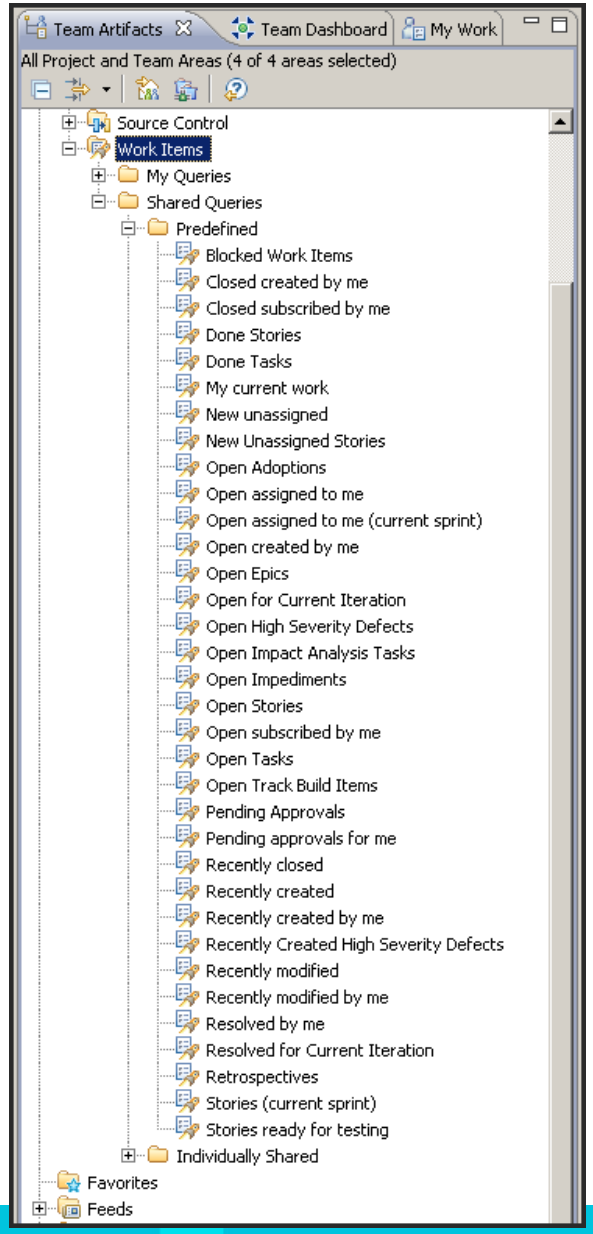


Creating a work item (continued)

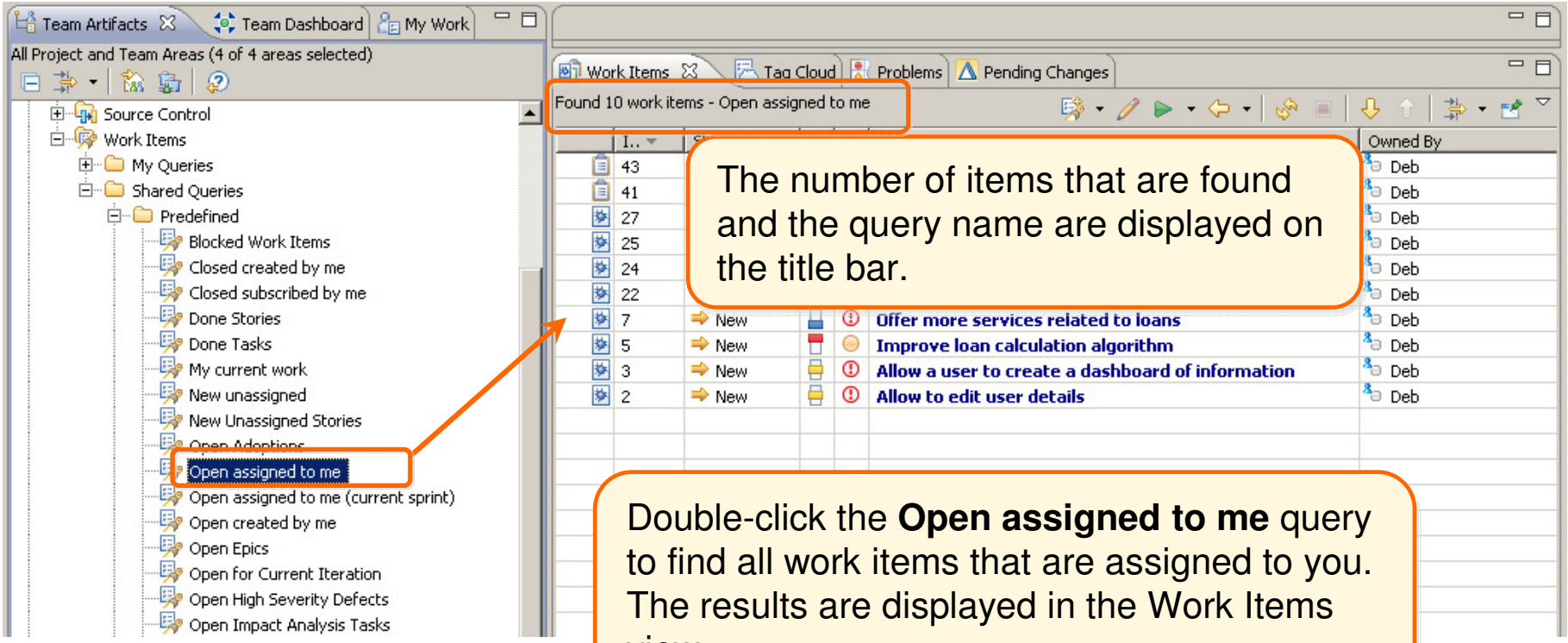


Use queries to find work items

- Queries are the primary mechanism for finding work items.
- Queries are stored in the `Work Items` folder of the Team Artifacts view:
  - The `My Queries` folder stores your private queries.
  - The `Shared Queries` folder stores predefined and individually shared queries.



Finding work items that are assigned to you

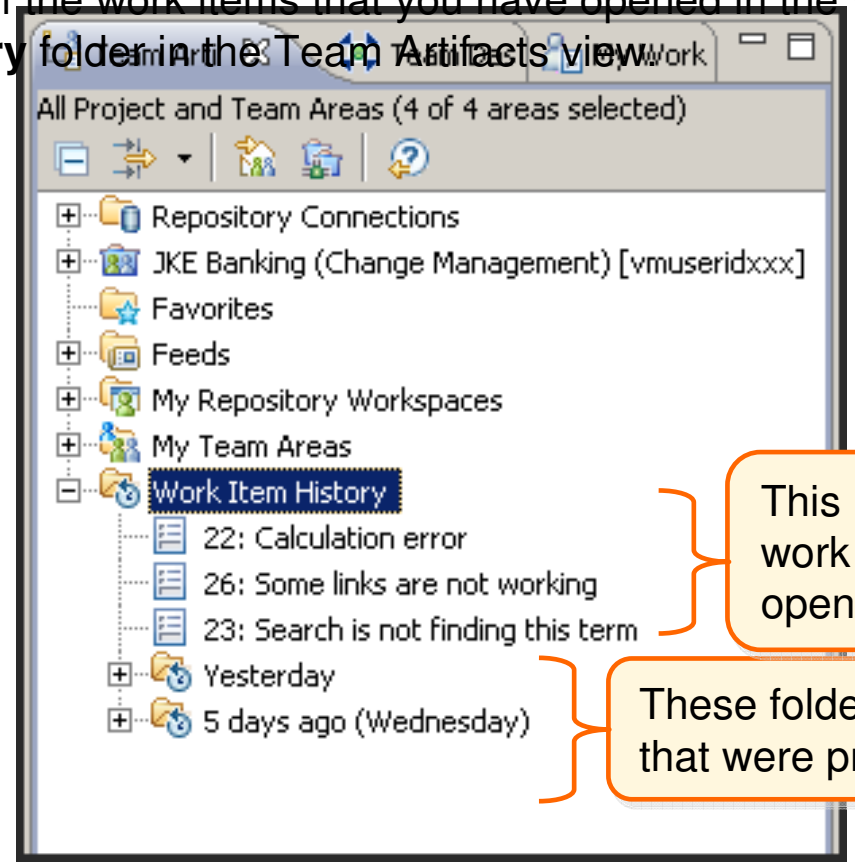


The number of items that are found and the query name are displayed on the title bar.

Double-click the **Open assigned to me** query to find all work items that are assigned to you. The results are displayed in the Work Items view.

Your work item History

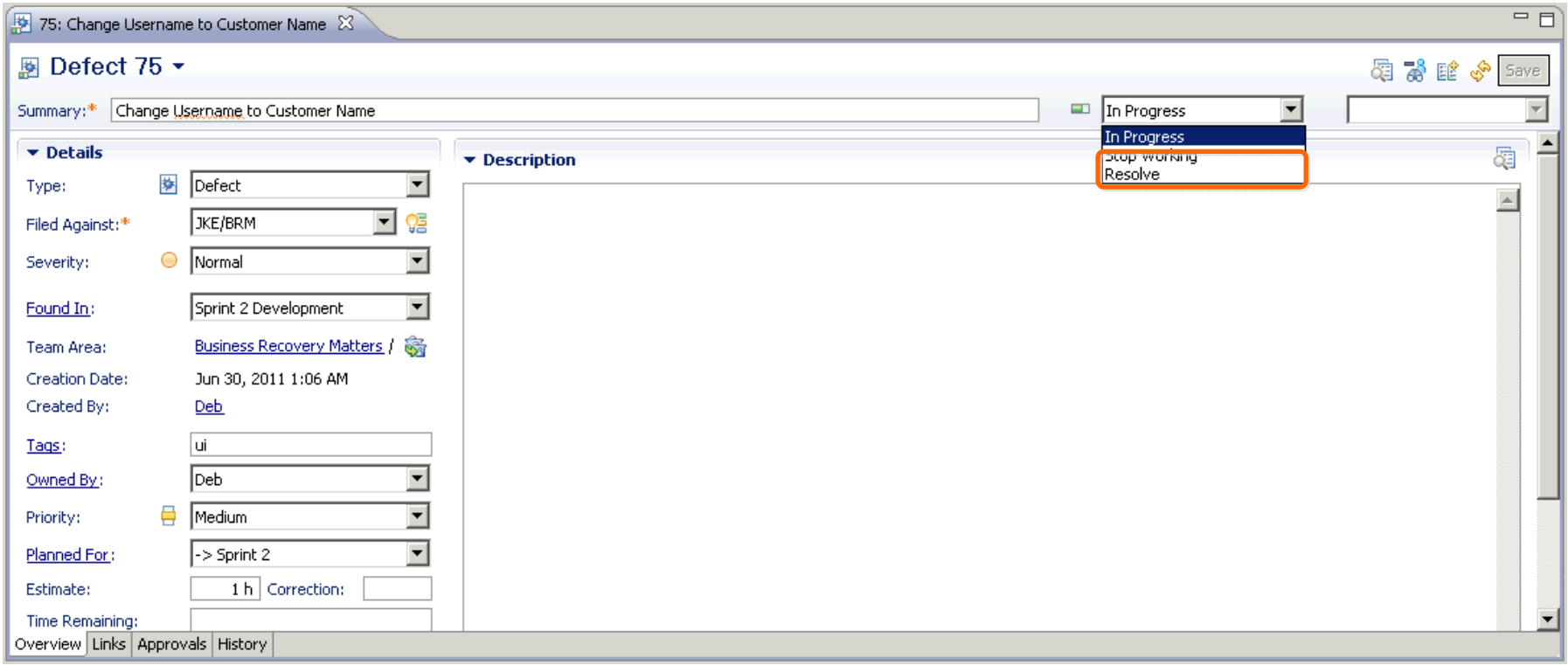
- To see a history of the work items that you have opened in the Work Item editor, expand the **Work Item History** folder in the Team Artifacts view.



Work items must reflect real-time status

- State the overall status of the work item.
- Describe your work and progress.
- Typical status changes include the following information in these areas:
  - Overview tab
    - State
    - Estimated time to resolve the work item
    - Discussions
  - Links tab
    - Add an attachment or subscriber
    - Add relationships to other work items
  - Approvals tab: Identify who can approve the work item
  - History tab: View the change history of the work item

Resolving a work item



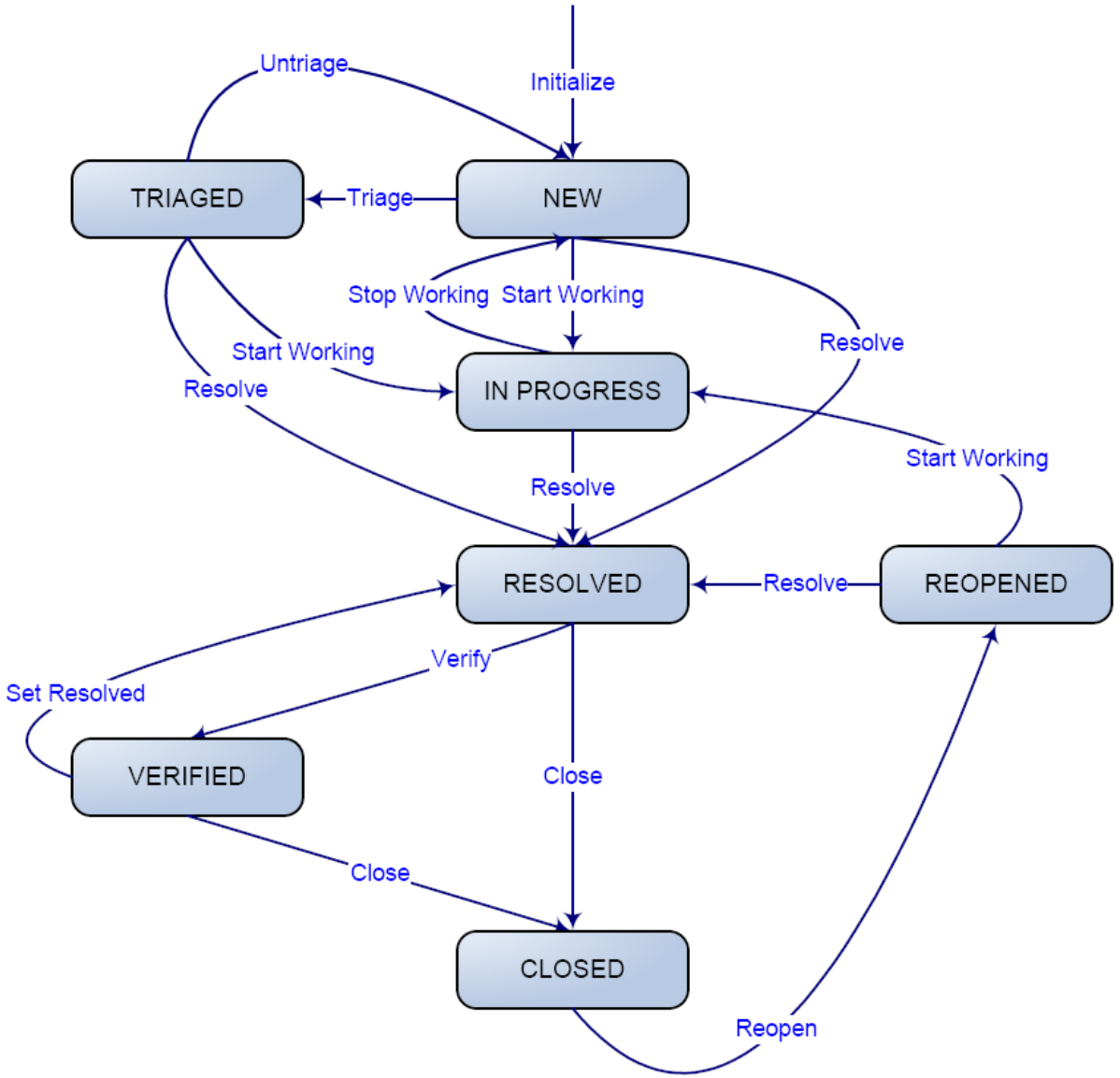
Work item states

Work items evolve from an initial state (New) to a final state (Closed).

Actions move the work items between the states.

This is a typical **workflow** for task-level work items.

Workflows are customizable.

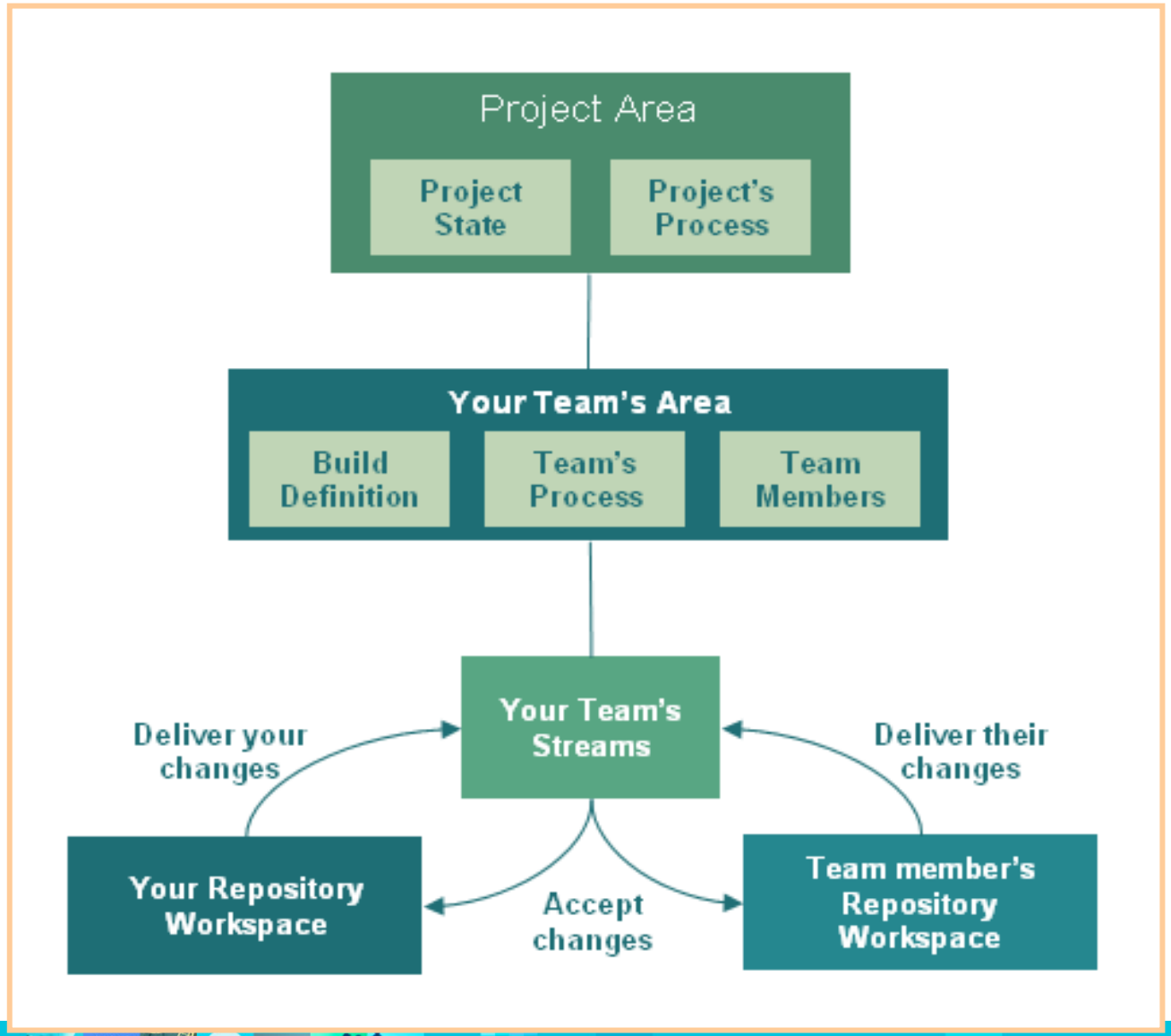


Jazz source control

- The Jazz Team Server
  - Uses a relational database in a central location for all artifacts that are based on Jazz:
    - Stores source code, documents, binary files, images, and so on
    - Stores all Rational Team Concert artifacts
    - Supports geographically distributed teams
  - Provides strong support for parallel development
  - Supports process-centric automation, both agile and traditional processes
  - Provides the infrastructure to link work items, plans, source-control artifacts, and builds

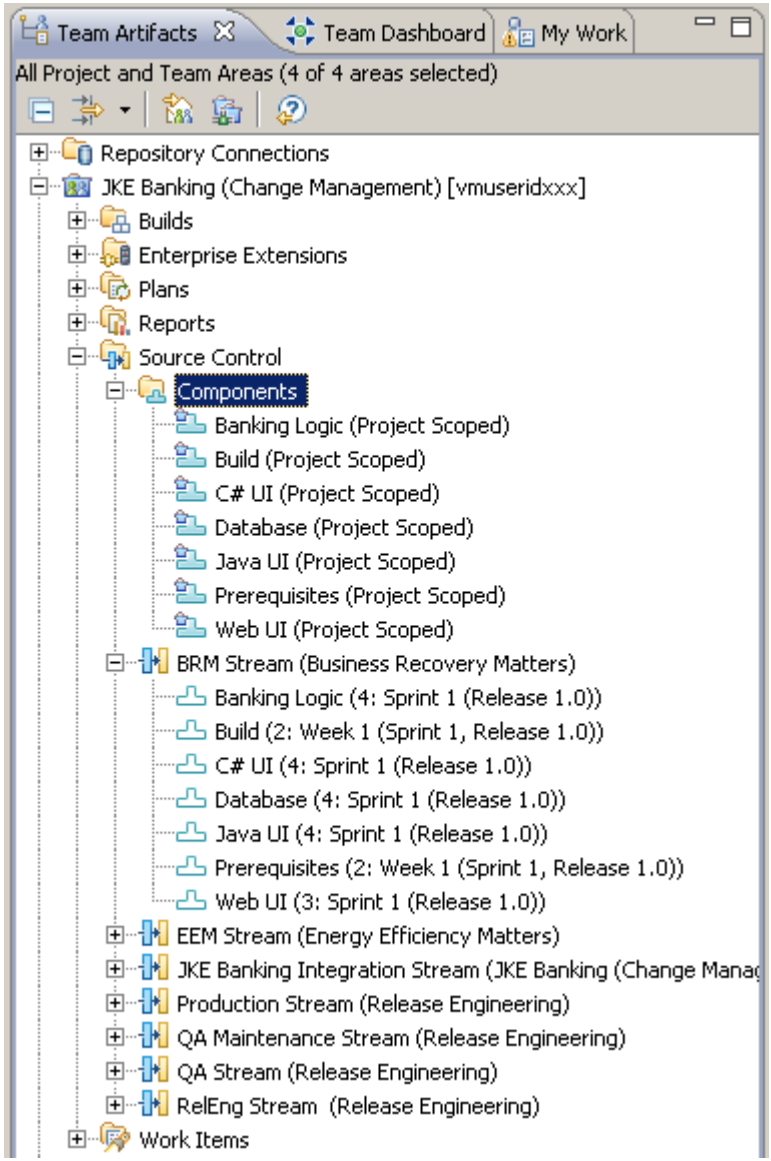
- The Jazz source-control repository
  - Is entirely change-set based
    - Requires that all changes to artifacts are related to a change set
    - Ensures atomic changes to sets of files are together
  - Provides secure mechanisms for creating, retrieving, updating, and deleting artifacts
  - Maintains a complete audit trail of all changes, including these events:
    - A record of past states of the item and past field values
    - The user who saved the item
    - The time of the change

Project area hierarchy



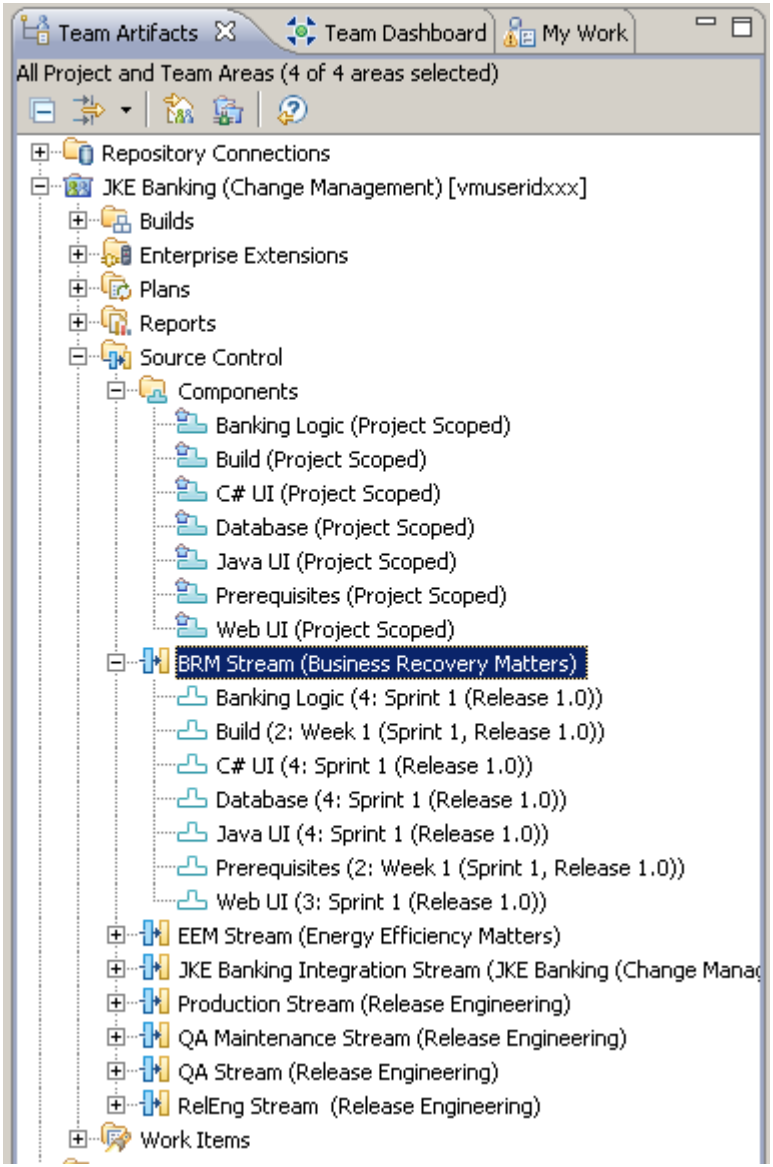
Components

- Fundamental organizational unit for artifacts under source control
- A collection of related artifacts
  - For example, an Eclipse plug-in or a group of documents
  - Any group of files and folders that share a common root can be a component
- A **Baseline** represents the “version” of a component



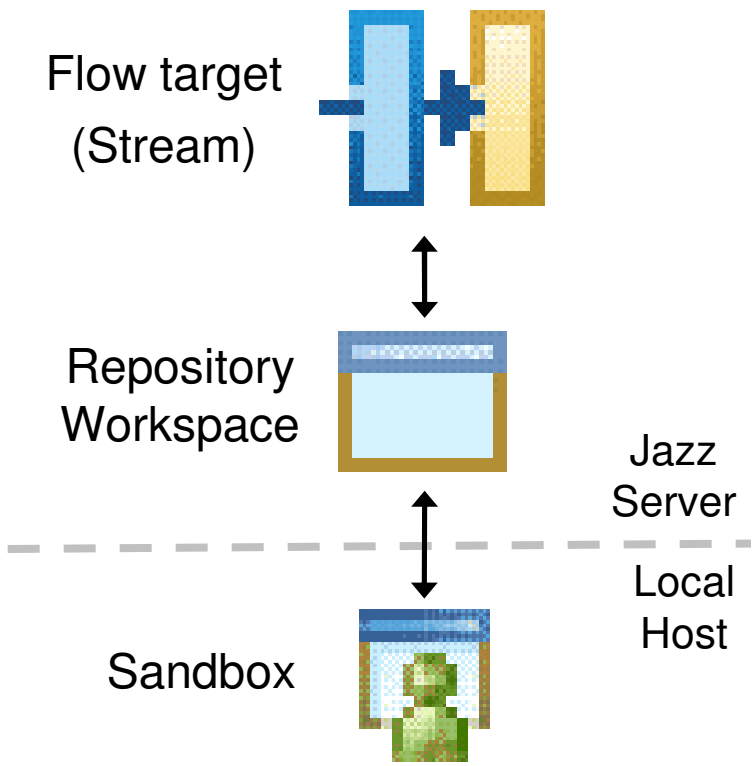
Streams

- A collection of one or more components
- Typical uses for streams:
  - Provide a team integration area
    - Controlled sharing of changes sets between teams and team members
  - Recreate important configurations
    - Team integration builds
    - Previous releases
    - Stable versions of third-party or open source software packages
  - Provide a mechanism for a phased promotion model
    - Development stream or streams
    - Integration stream
    - Production stream



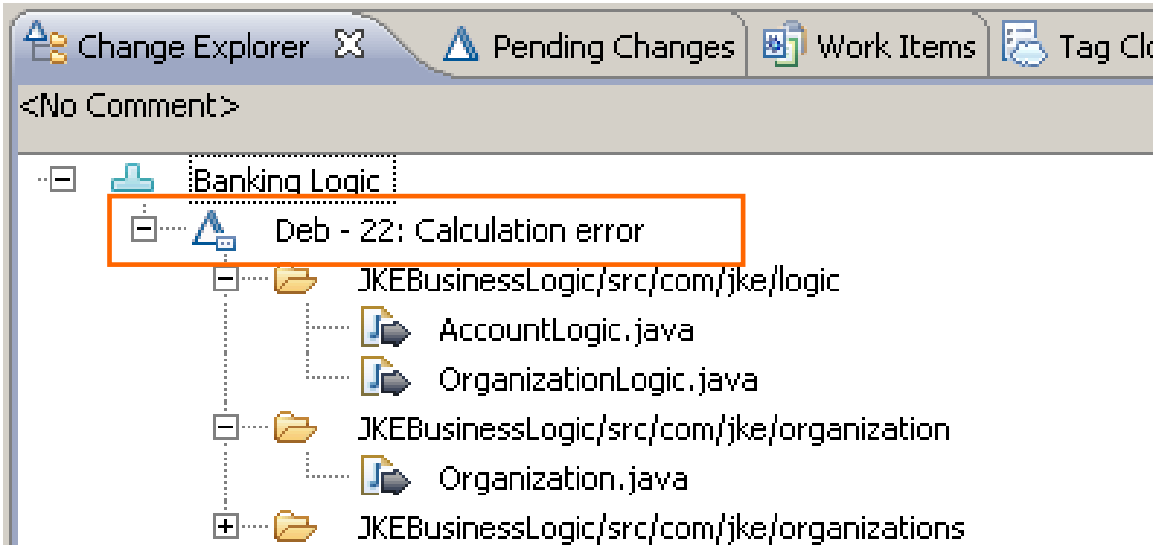
Repository workspaces

- A server-side storage area that a developer uses for current work
  - **Flow target**
    - Specifies the initial configuration of components
    - Identifies the location where change sets are shared
  - **Sandbox**
    - provides access to files and folders for desktop integrated development environments (IDEs), such as Eclipse and Visual Studio, and other development tools



Change sets

- A **change set** is a repository object that collects a related group of changes in a component:
  - The system tracks the changes that you make in the context of your current work as a change set.
  - Change sets are applied to a flow target in a single atomic operation



 indicates change set

Pending changes

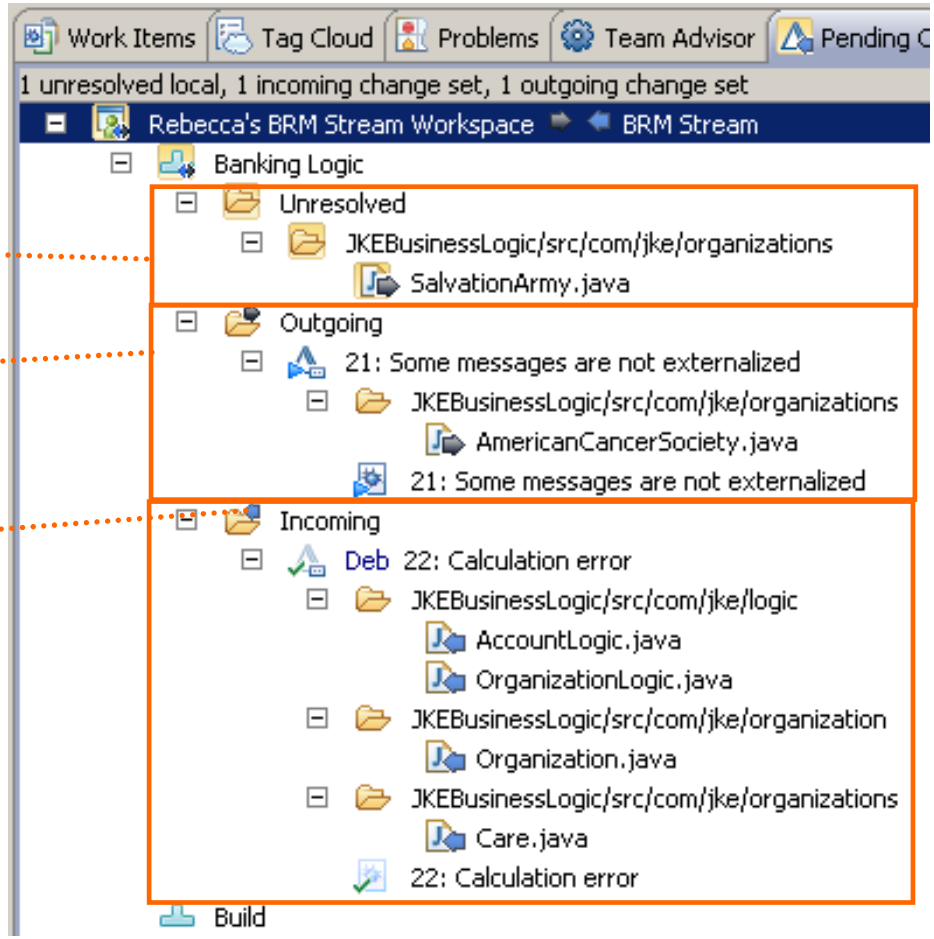
- Use the **Pending Changes view** to manage the flow of change sets into and out of your repository workspace
  - **Unresolved**

    Files in this section have been modified in the local sandbox, but have not been checked in to the repository workspace.
  - **Outgoing**

    Checked-in files are organized into change sets.

    Outgoing change sets have not been delivered from the repository workspace to the flow target.
  - **Incoming**

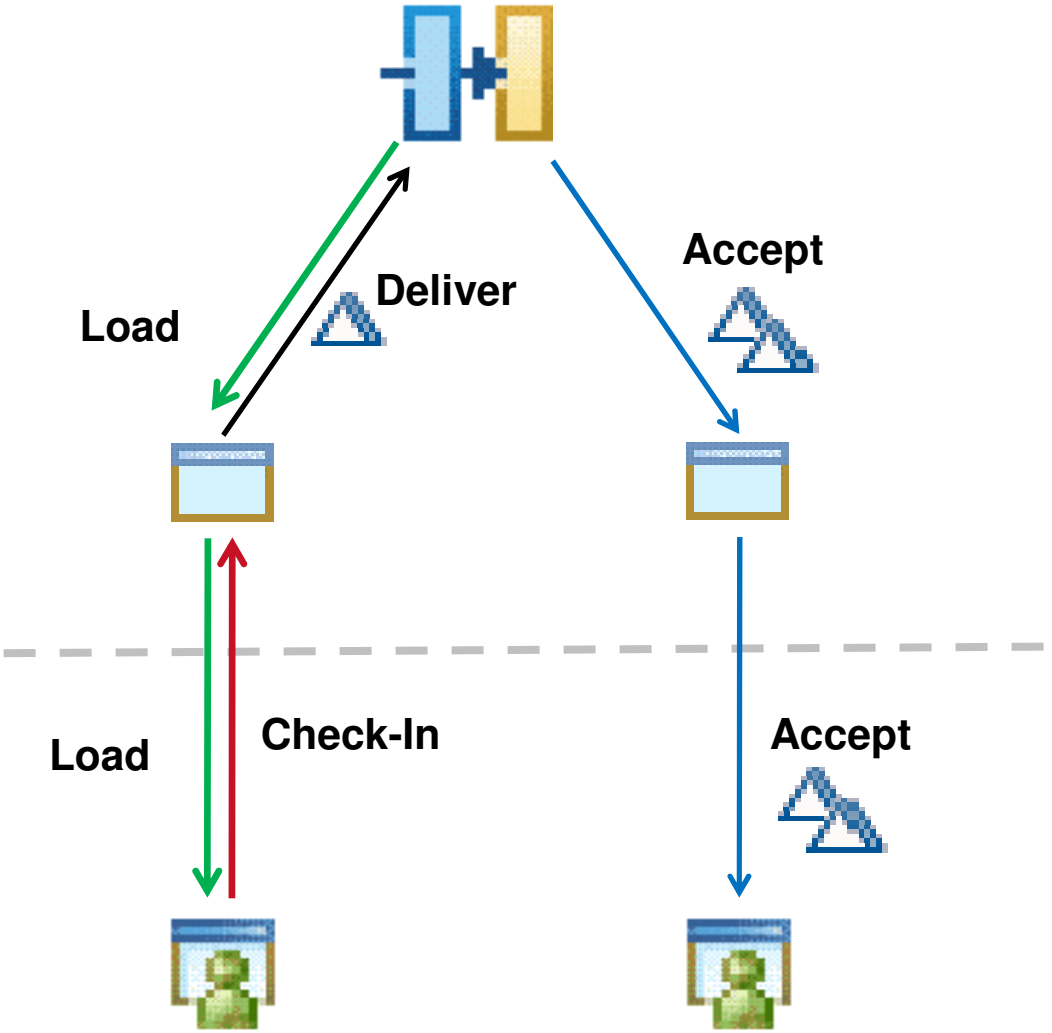
    Incoming change sets represent changes that have been applied to the flow target, but have not been accepted into this repository workspace.



Using workspaces in a team environment

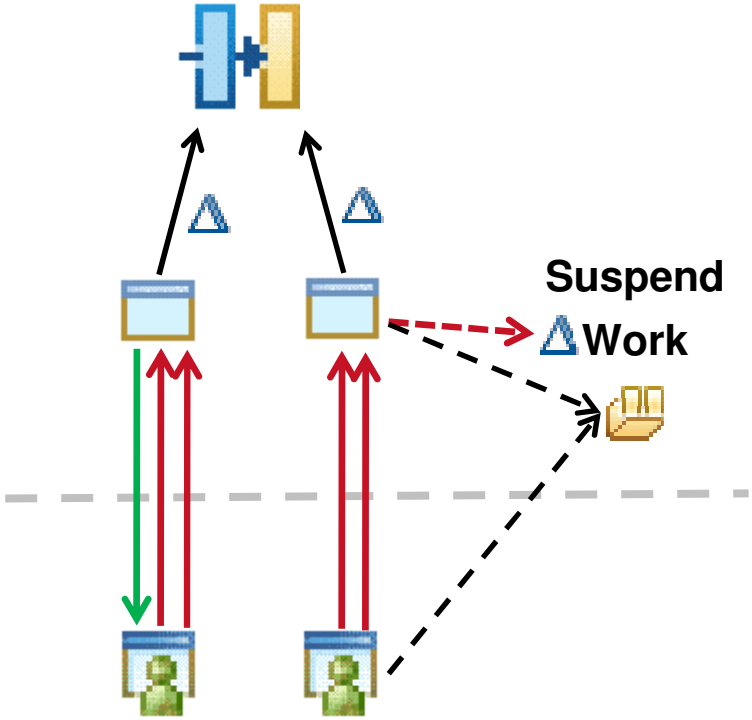
- Load
  - Copies a specified configuration of files from the stream to your repository workspace and local sandbox
- Check-In
  - Copies changed files from the local sandbox to the repository workspace
- Deliver
  - Copies the change set or sets from the repository workspace to the flow target
- Accept
  - Copies a change set or sets in the flow target into the repository workspace and the local sandbox

- Deliver and Accept operations will identify potential conflicts if the same file is modified in more than one change set.



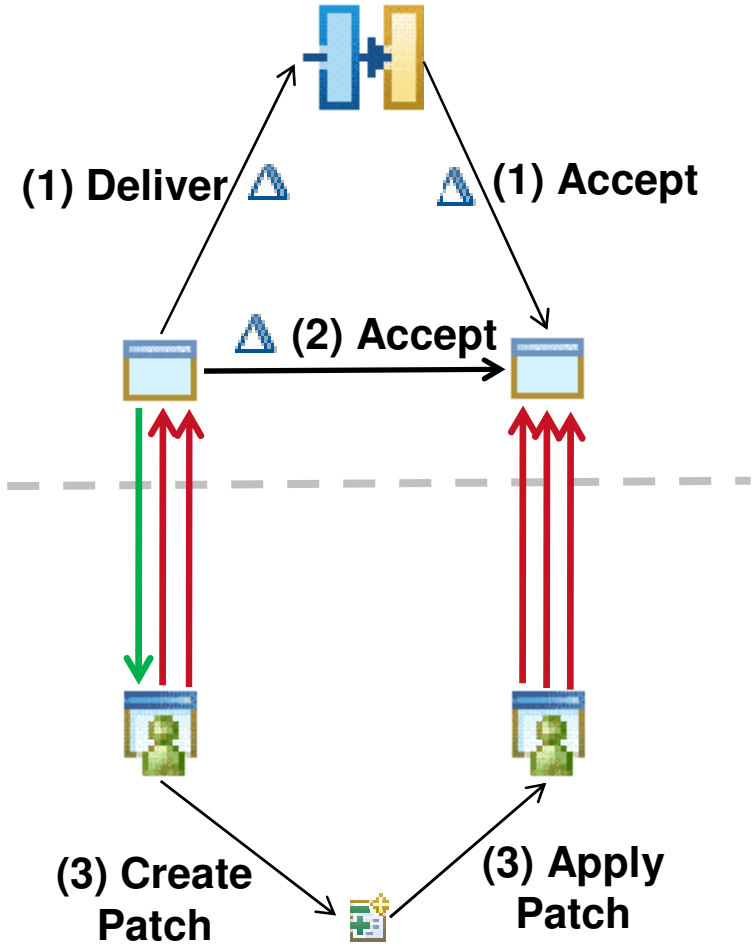
Creating and managing change sets

- **Best practices:**
  - **Set your "Current Work Item."**
  - **Work on one task at a time.**
- If you have to multitask:
  - Use multiple repository workspaces if you need to work on multiple tasks at the same time.
  - **Suspend** work on a change set, which puts that work off to the side temporarily. You can resume the work later.
- Worst case scenario:
  - Adjust change set contents in your repository workspace before you deliver.



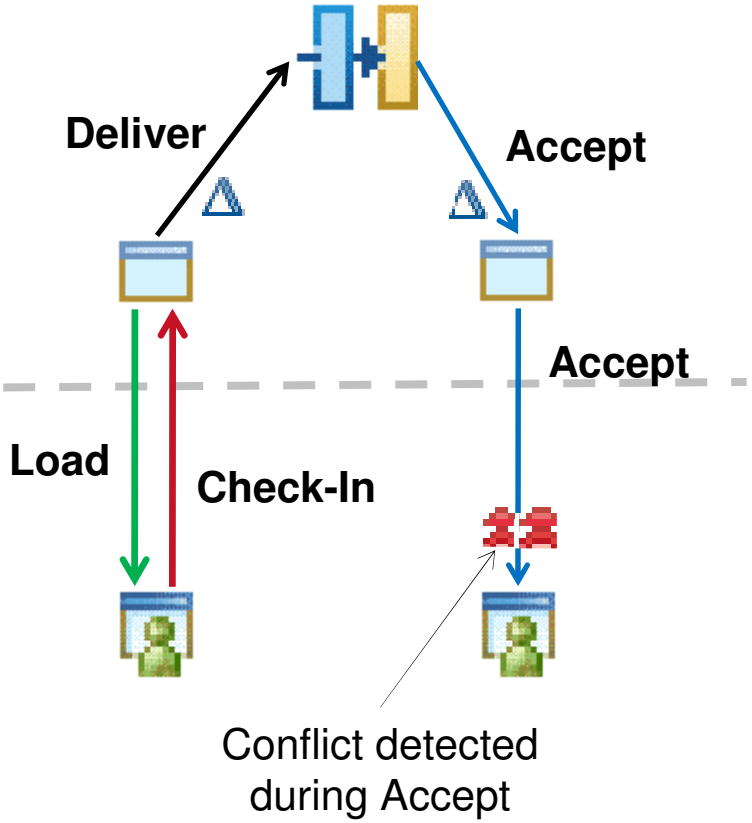
Sharing work between workspaces

- Multiple developers who collaborate on a task must be able to share their work:
  1. Use the normal **deliver-accept** model to propagate a change set from one workspace to the other through a parent stream or repository workspace.
  2. Change your flow target to point to the other workspace and **accept** changes directly from that workspace.
  3. Create a patch file, which other users can apply to their workspaces.



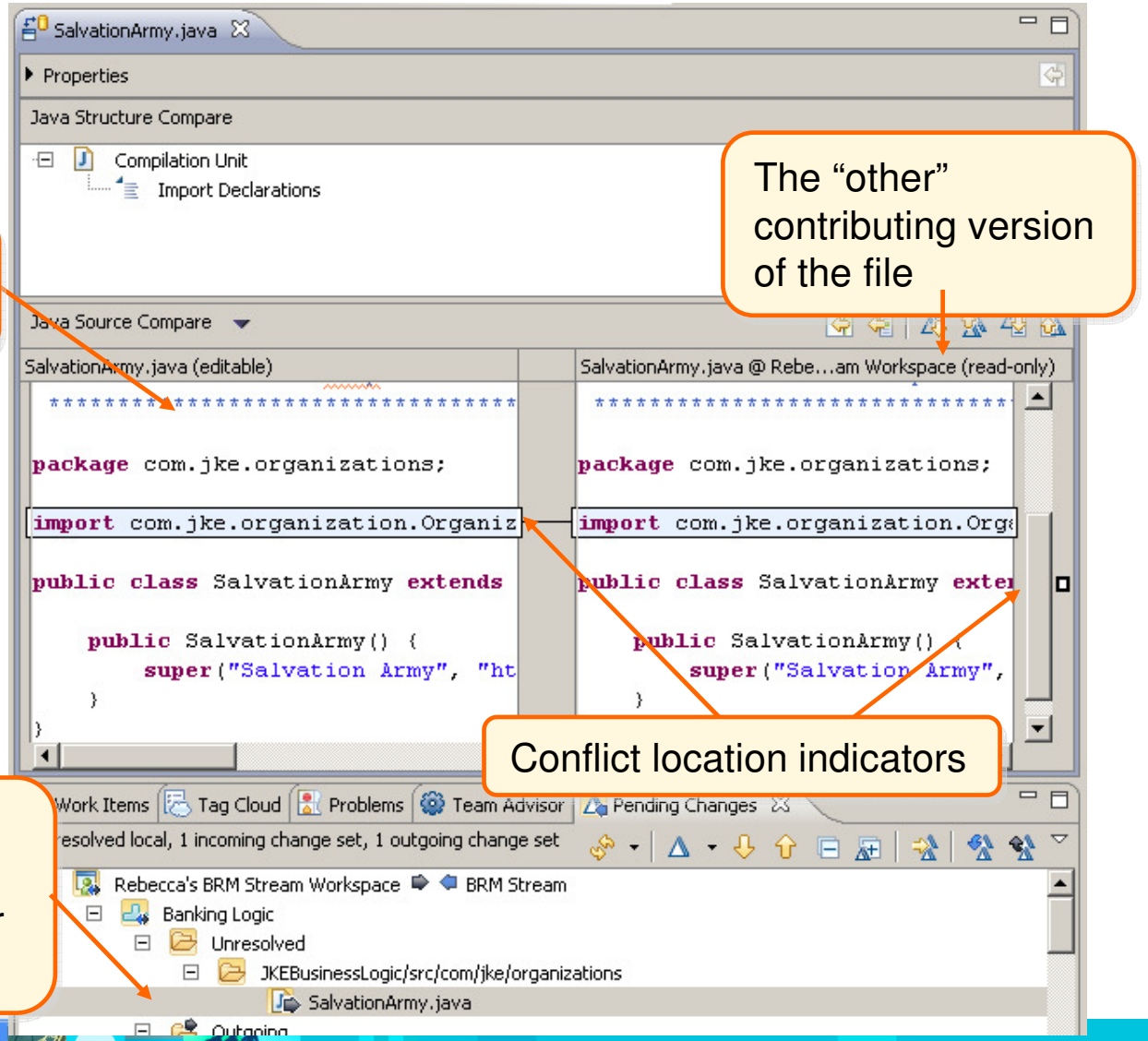
Finding conflicts

- Conflicts are detected at the file level, so any changes to the same file by two different change sets can potentially result in a conflict
  - Conflicts can be detected during either the **delivery** or the **accept** process
  - Rational Team Concert will offer to “auto-resolve” conflicts without user intervention
  - If the changes cannot be resolved automatically (for instance, the same line of code was modified in both workspaces), Rational Team Concert will prompt the user to perform a manual merge



Resolving conflicts

- Use the **Compare editor** to manually resolve conflicts.



Your editable version of the file

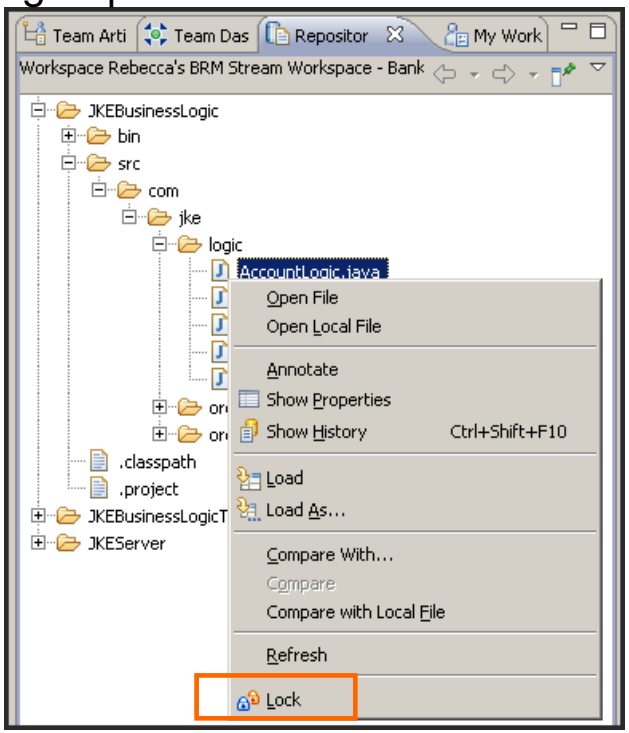
The "other" contributing version of the file

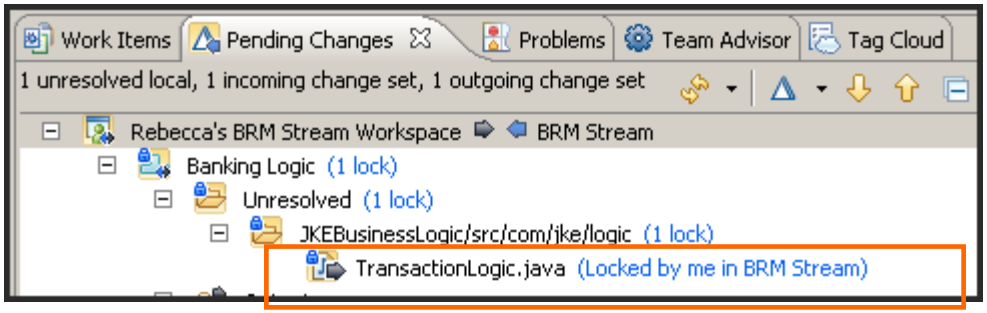
Conflict location indicators

The conflict is called out in the Change Explorer view

Avoiding conflicts

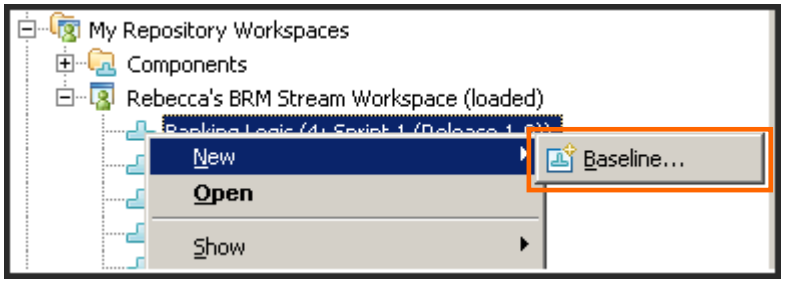
- File locking
  - You can lock a file on a "per-stream" basis.
  - While you have a file locked, nobody else can deliver changes to that file in that stream.
  - A file can be unlocked by the user who owns the lock or by a member of the JazzAdmins group



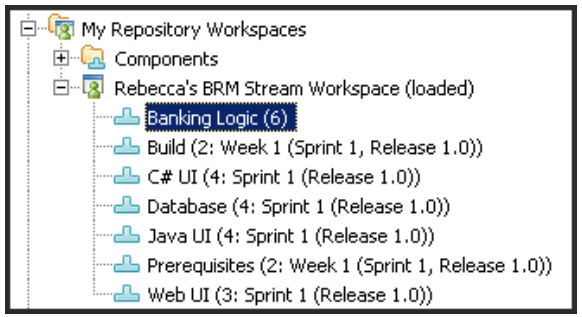


Baselines

- A **baseline** is a version of a component
  - Components configurations can be saved at any point in time through creation of a **baseline**.
  - Baselines are created in a repository workspace and delivered to a stream as a special type of change set.

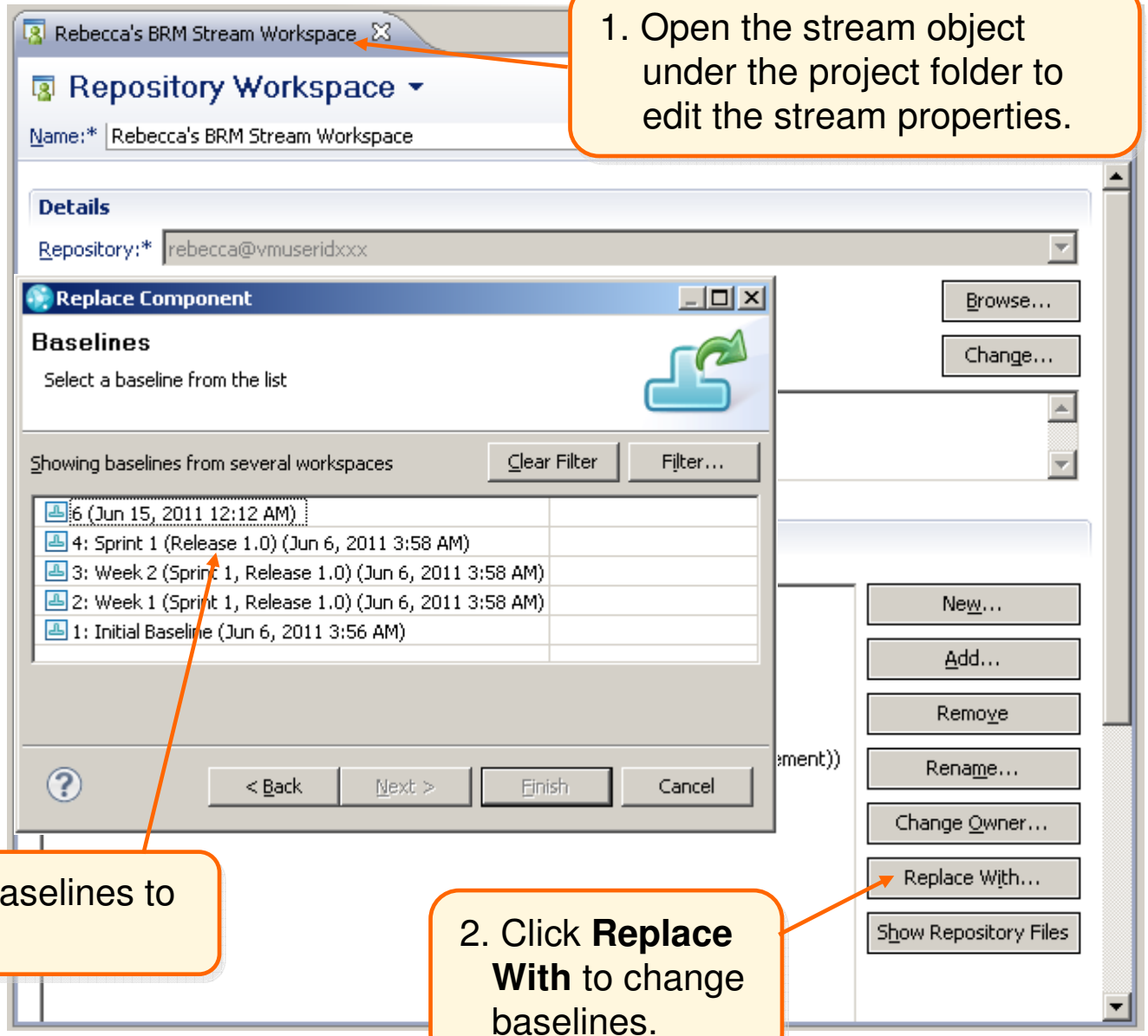


  - The Team Artifacts view indicates which component baselines are in a repository workspace or stream.



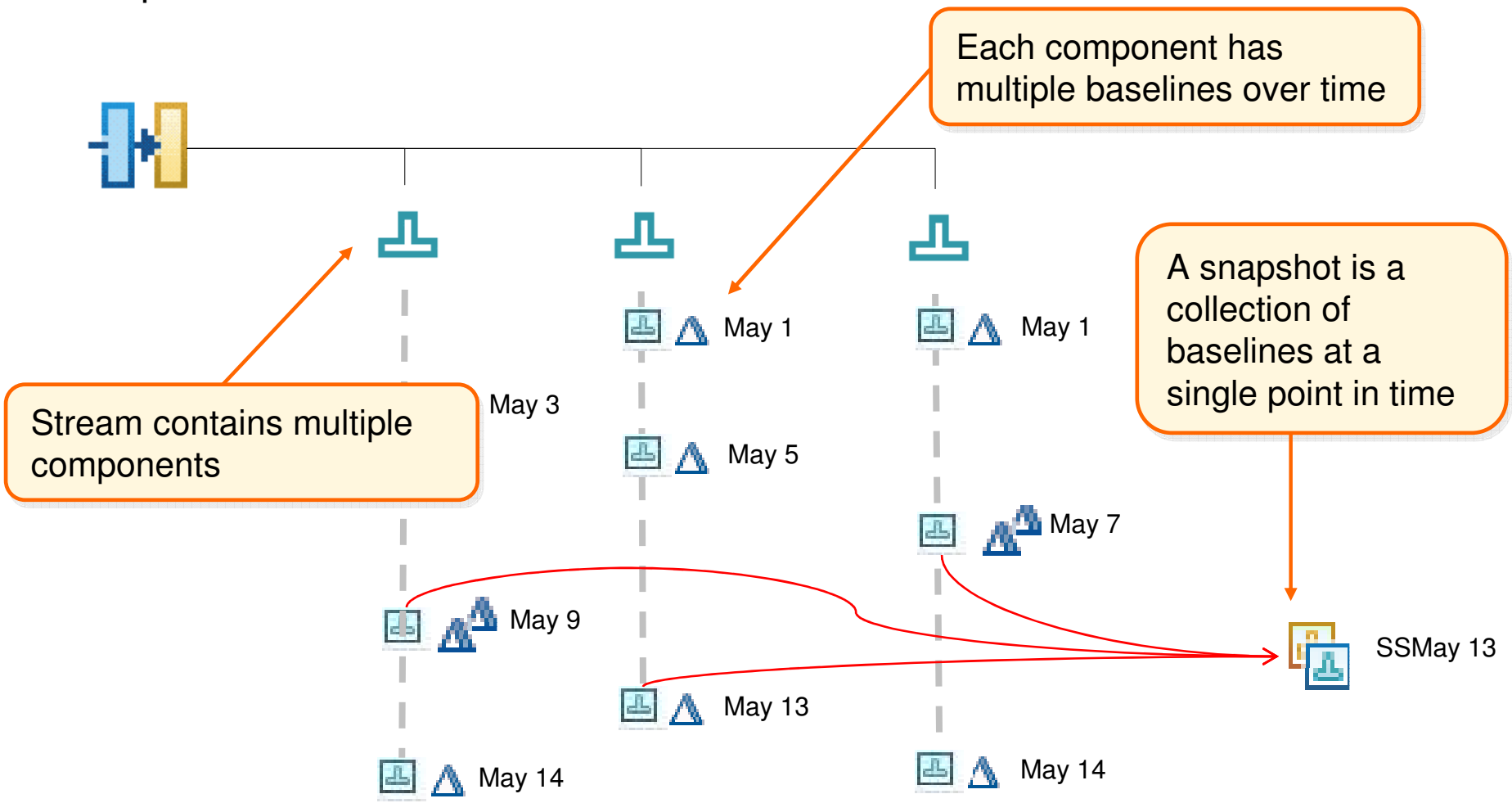
Restoring configurations

- Replace the current component configuration in a stream or repository workspace with a different baseline.

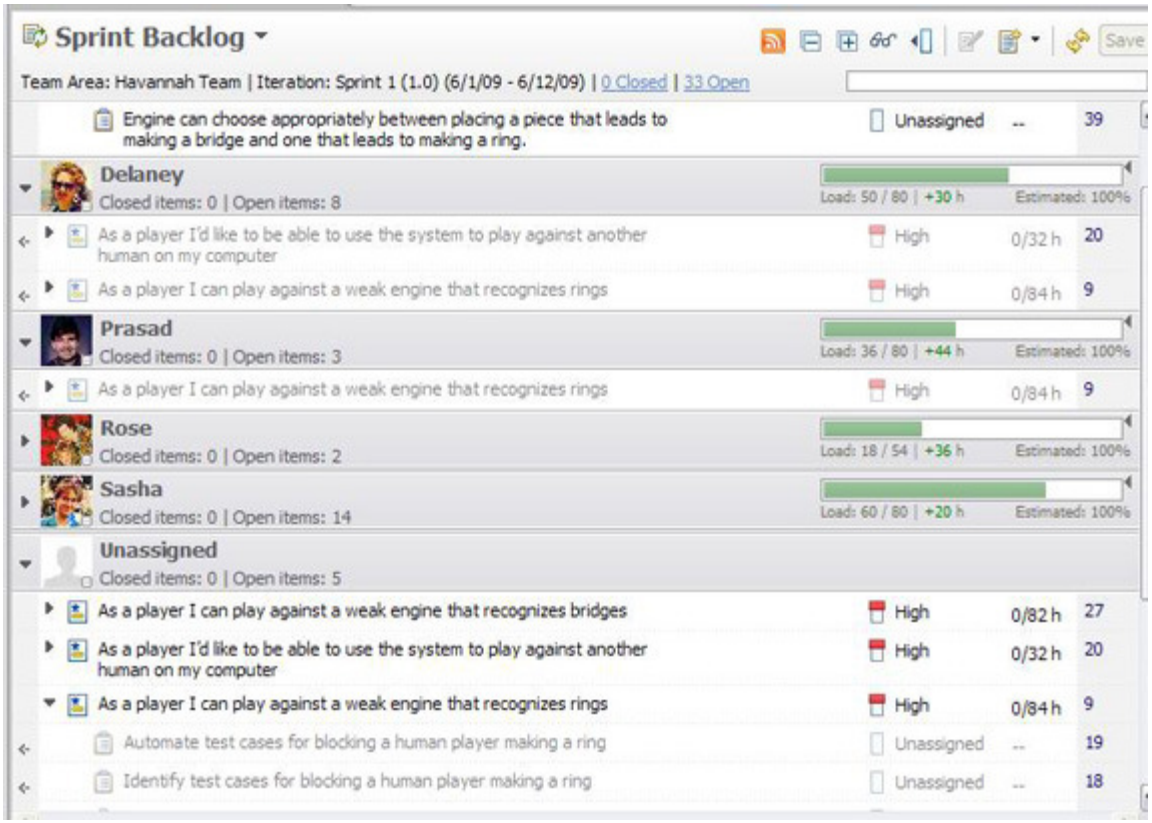


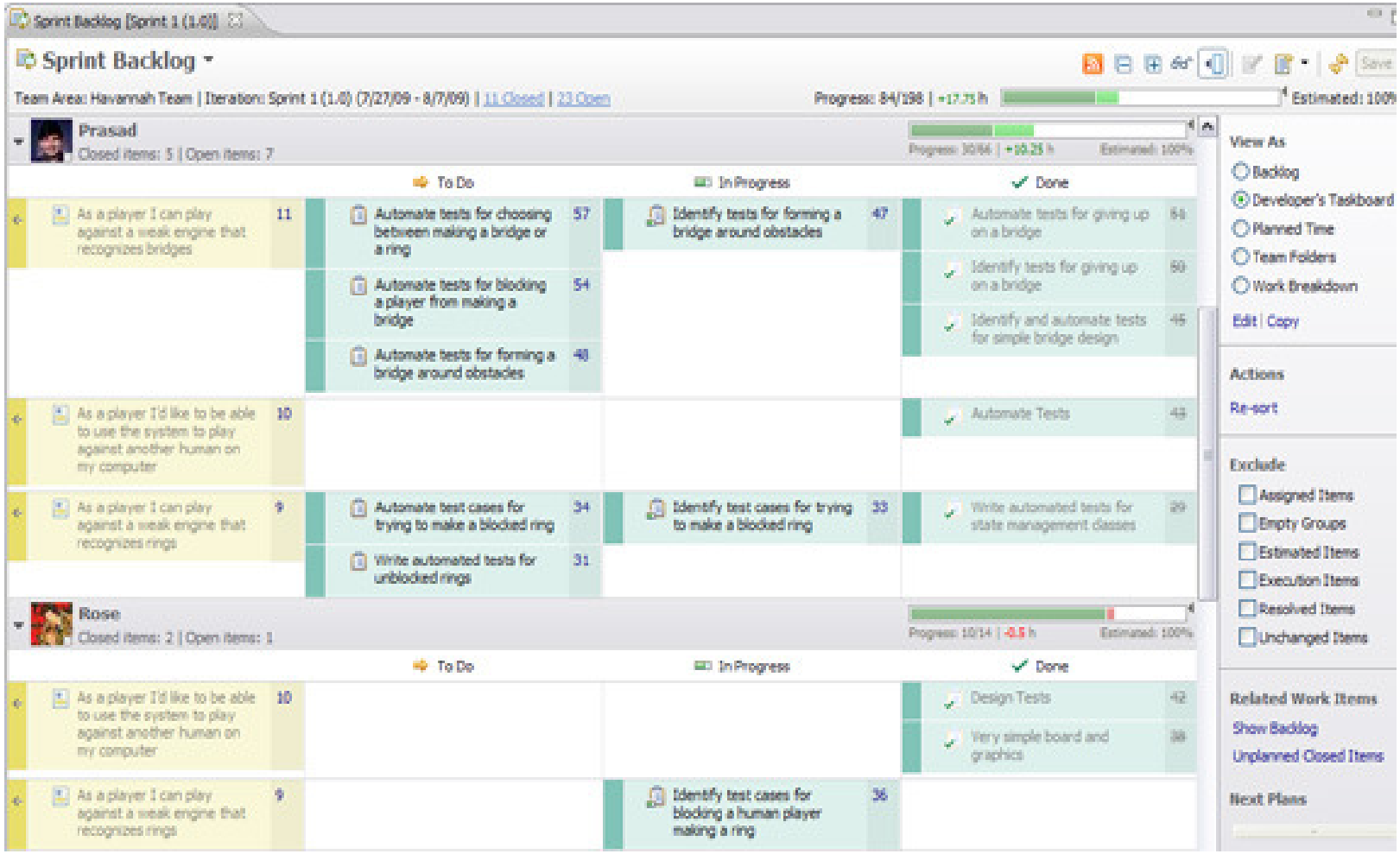
What is a snapshot?

- A **snapshot** is a collection of baselines across all of the components in a repository workspace or stream



Project Management - Dashboards




Sprint Backlog [Sprint 1 (1.0)]
Sprint Backlog
Team Area: Havannah Team | Iteration: Sprint 1 (1.0) (7/27/09 - 8/7/09) | 11 Closed | 23 Open
Progress: 84/198 | +17.75 h
Estimated: 100%
Prasad
Closed items: 5 | Open items: 7
To Do
In Progress
Done
As a player I can play against a weak engine that recognizes bridges 11
Automate tests for choosing between making a bridge or a ring 57
Automate tests for blocking a player from making a bridge 54
Automate tests for forming a bridge around obstacles 48
Identify tests for forming a bridge around obstacles 47
Automate tests for giving up on a bridge 64
Identify tests for giving up on a bridge 60
Identify and automate tests for simple bridge design 45
As a player I'd like to be able to use the system to play against another human on my computer 10
Automate Tests 43
As a player I can play against a weak engine that recognizes rings 9
Automate test cases for trying to make a blocked ring 34
Write automated tests for unblocked rings 31
Identify test cases for trying to make a blocked ring 33
Write automated tests for state management classes 29
Rose
Closed items: 2 | Open items: 1
Progress: 10/14 | -0.5 h
Estimated: 100%
To Do
In Progress
Done
As a player I'd like to be able to use the system to play against another human on my computer 10
Design Tests 42
Very simple board and graphics 38
As a player I can play against a weak engine that recognizes rings 9
Identify test cases for blocking a human player making a ring 36
View As
Backlog
Developer's Taskboard
Planned Time
Team Folders
Work Breakdown
Edit | Copy
Actions
Re-sort
Exclude
Assigned Items
Empty Groups
Estimated Items
Execution Items
Resolved Items
Unchanged Items
Related Work Items
Show Backlog
Unplanned Closed Items
Next Plans