Integrating Rational Team Concert with GitHub

Rational Team Concert and Git

Starting with release 5.0, Rational Team Concert (RTC) supports integration with the Git version control system. This integration is particularly useful if there are pockets of Git use in your teams and you want to use RTC for project and process management, permissions, defect and task management, and planning across all teams, regardless of whether you use the built-in RTC source control or Git.

Supported Git distributions

RTC supports different types of Git configurations. You can use your own Git repository, or a popular Git distribution such as GitLab or GitHub. If you use your own central Git repository, then you can use RTC to manage permissions and behavior for your Git operations. If you use a Git distribution hosted by a third party, then the extent of integration depends on the extensibility supported by the third-party distribution. Work item linking with Git commits are supported across all Git distributions that RTC integrates with. Process and permissions are supported in varying degrees.

Note: For process support, RTC and Git must share a common user base (for example, be served by a common directory server), because permissions and process are determined by user roles in RTC project areas.

Where does a Git repository get plugged in?

A Git repository gets registered to an RTC project area as an alternative source control system for teams. After the repository is registered, the process and permissions for Git operations in the repository can be governed by the “owning” process area, which can be the project area or one of its child team areas. Optionally, you can map the branches of the repository to the child team areas. If the branches are mapped, then operations in those branches are governed by the process and permissions of that team area.

After a Git repository gets registered, RTC generates a unique key for it. The key needs to be added as a property for the Git repository and is used by RTC to identify the repository during work item linking or process checks.

Permissions and process control

RTC supports role-driven permissions around Git operations, such as clone, create branch, push commits, and so on.

You can also control the Git push operation via process rules that are set up in RTC preconditions and follow-up actions. For example, you can mandate that a commit message must always contain a valid work item number for linking.

For information about setting up process checks for your Git repository, see [Defining Git-related permissions and preconditions](https://jazz.net/help-dev/clm/index.jsp?topic=%2Fcom.ibm.team.connector.cq.doc%2Ftopics%2Ft_git_install.html).

Note: As of 6.0.6, the RTC Integration only supports process and permissions for those Git repositories where you have permissions to deploy hooks. There is no process support for publicly hosted GitHub or for publicly-hosted GitLab.

Linking work items

You can link Git commits to RTC work items by specifying the work item number in the commit message. After the commit is pushed to the Git repository, the RTC Git integration links the Git commit to the specified work item.

You can use commit messages to perform actions, such as creating a new work item, or resolving an existing one. For example, the following commit message creates a task in the JKE Banking Project Area in RTC, with priority High, owner ADMIN, tag “checking”, summary “Add support for checking accounts”, and links the Git commit to the newly created work item.

git commit –m “#create \*Task ^\”JKE Banking \” @:ADMIN $High #checking Add support for checking accounts”

For information about updating existing work items or creating new ones when you push Git commits, see [Updating work items with actionable Git comments](https://jazz.net/help-dev/clm/index.jsp?topic=%2Fcom.ibm.team.connector.cq.doc%2Ftopics%2Ft_git_actionable_comments.html) and [Creating work items with Git commit messages](https://jazz.net/help-dev/clm/index.jsp?topic=%2Fcom.ibm.team.connector.cq.doc%2Ftopics%2Fr_git_wi_commit.html).

Git and Jenkins

[Team Concert Git Plugin](https://wiki.jenkins-ci.org/display/JENKINS/Team+Concert+Git+Plugin) for Jenkins integrates Jenkins with RTC builds, which use Git as source control. This plugin creates traceability links from a Jenkins build to Rational Team Concert [Work Items](https://jazz.net/products/rational-team-concert/features/wi) and [build](https://jazz.net/products/rational-team-concert/features/build) results.

More information