Tips for implementing Test Versioning

When talking about versioning, different things may come to the surface that may somehow be related.

Is it requirements versioning? Test versioning? Are you talking about tracking changes to your artifacts or having explicit versions each time you change an artifact?

Let's see first how Jira handles changes and versions in general and how that approach is feasible also for managing versions of Tests, at least up to some level.

Then we'll see how Xray Enterprise enables proper Test Case Versioning, allowing users to manage multiple versions at the same time for each one of your Tests.

How Jira handles versioning in general

When we talk about versions in Jira, we're usually referring to project releases. Jira issues are assigned to a given project version by specifying the "Fix Versions" field. Additionally, "the "Affects version" can be used to identify that a given issue impacts the mentioned project version/release.

At each issue level, Jira ensures that changes are tracked under the History section available in the corresponding issue screen, so you know who changed what and when. Even though changes are tracked for history and auditing purposes, there is no version management at issue level, where users would be able manage multiple versions of the issue content, including all its fields.

So, how do teams deal with this need, whenever they want to have multiple versions of the same issue?

Usually, these versions are tied up to a new iteration, a new enhancement. As an example, if you have a requirement (e.g., a story) for version v1.0 and you want to change it somehow in a future version (e.g., v2.0), normally, you will have to create another requirement issue to specify the new behavior. This will lead to the creation of additional issues.

However, if you are changing the specification during the development lifecycle of that requirement, you can simply update its description and use Jira workflows to guarantee that is being handled properly.

If you make changes in some of the attributes of an issue, it gets tracked in the History changes; however, the version is not incremented because it is used for another purpose: to tell for which version you're delivering it.

Test Versioning with Xray Standard

Xray can handle the versioning of Tests in the same way as Jira handles the versioning of other issue types. Since Xray's Tests are Jira issues, Xray Standard allows the same approach so your users work the same way they already do.

However, even though you can see all the changes of a Test case using the Jira History tab, it is not easy to revert or view a previous Test version easily unless you create a new Test issue for each version. This is also not ideal, as you end up with multiple issues for the same entity that complicates their management.

Using the Jira way for addressing test versioning, even though feasible and sufficient for most teams, has many limitations if you aim to have a thorough test versioning system in place.

Tips

- Avoid cloning Tests; implementing test versions by cloning Tests and changing their content will lead to many issues that will be hard to manage

Limitations

- No way to create multiple versions, as such, for each Test (i.e., each one holding a specific copy of the test specification details)
    - No way to revert to a previous "version" of a Test, as there is no notion of version
    - No way to create different variants of a Test (e.g., a version with the manual test steps and another version with the Gherkin specification used by automation)
- No way to manage the lifecycle of Tests independently of the project lifecycle
    - Jira versions are created at the project level. Hence, all issues within the project are affected by the same version numbers. However, tes t cases can have a different lifecycle than other issues in the same project, such as requirements or bugs. For instance, we can have multiple versions of a Test case for the same project version (e.g., the test has evolved with the changes in the requirement; they have different types).

Although your tests may live across many versions of your project, they're made in the context of some version, most probably for the version of the requirement they aim to validate. Thus, assigning them to a Fix Version serves the purpose of tracking the specification+implementation of the test case, the first time they have been used. This way, you can also track it in the Release as one artifact that needs to be "done" (i.e., specified/reviewed/approved).

In order to overcome these limitations, Xray Enterprise introduced an explicit way of versioning Tests.

Test Versioning with Xray Enterprise

With Test Case Versioning, available only for Xray Enterprise, it is now possible to create new versions within the same Test issue. You can revert back to a specific version and view all versions of a Test case easily.

Learn more

Please check the Test Case Versioning user documentation page to see how to use this feature.

Test Versions

A Test Version is like a different branch or “variant” of a Test, that has its own specification, test type, and history.

A Test Version is not like an incremental revision or “snapshot of the test at a given moment”, even though users can create a new version of Test using the latest contents of another version.

Main characteristics

- A Test always has one default version; the default version is the one that is considered whenever adding a Test to a Test Execution
- A Test Version has:
    - A “name” attribute, that acts as a description and is the common name for referring to the Test Version itself
    - An internal attribute “version” (incremental); this is like a revision, useful if we wish to mention a specific change made within the Test Version
    - Dataset, if any is associated with it
- Coverage information, i.e., the list of covered requirements, is not part of the version; in other words, the link between the Test and requirements is independent of the version; it’s like saying that “all versions cover” exactly the same requirements
- A Test Version can be archived/unarchived. An archived test version:
    - Cannot be added to a new Test Execution
    - Can still be executed if it’s already part of a test execution

Possible use cases for adopting Test Case Versioning

Simultaneously manage multiple variants of the same test

- A requirement has a manual scripted test and we want to implement an automated variant of it, and we wish to have both at the same time and clearly track results independently

Simultaneously manage multiple versions of the same test, with different content

- A requirement can have some slight differences, for example localization related; a different test version can be created to have the specifics of those (e.g., on a different language)

- A requirement may be evolving while it's being developed and the Test needs to evolve also; different versions may be used as "milestones" allowing the team to easily rollback or go to a specific "milestone" of that requirement and test

Test Versions and other entities

Preconditions

- Preconditions don't have versions
- A Test in a given version can require some precondition(s); another version of that Test may require different preconditions, if any

Test Sets

- The Test Sets only contain a list of Tests, and not a specific version of each Test

Test Plans

- The Test Plans only contain a list of Tests, and not a specific version of each Test
- Versions are assigned to the Tests within the Test Execution whenever it is created from the Test Plan

Test Executions

- A Test Execution contains a specific version of each Test; the version is assigned at the moment each Test is added to the Test Execution; the default one is picked unless otherwise specified
- To add a specific version of a given Test to a Test Execution, the user needs to go to the Test issue screen and add it to an existing Test Execution; it's not very straightforward and cannot be done directly from the Test Execution issue

Tips

- Avoid cloning Tests; assess if Test Versions can be used to fulfil your needs
- Avoid having many versions of a Test, as it can complicate their management
- Don't use Test Versions as a way to deal with different requirement versions; a different requirement version usually in Jira is handled as a separate requirement issue, with its own specifics, with its own development lifecycle, and its own tests
- Test Versions provide the ability to manage multiple versions for each one of your Tests. This may be quite useful but also adds some complexity, therefore we recommend discussing the usage of this feature with your teams

Limitations

Test Case Versioning with Xray Enterprise provides real management capabilities for test versions. However, there are some limitations (some by design and other that can be overcome in the future).

- A specific Test Version cannot be linked to a "requirement"; the coverage is always between the Test and the requirement issues, and there are no additional variables taken into account
- It's not possible to select the versions to use for each Test, whenever adding Tests to Test Plans or Test Executions
    - It's also not possible to select the versions to use for each Test, whenever creating Test Executions from within the Test Repository or from the Test Plan Board; it's also not possible to do so from the requirement issue screen
- Each Test can have a set of Test Versions, with user-defined names; there is no convention across different Tests, therefore a Test may name a version "v1" and another Test may name a version "myinitial-version-1"
- Even though changes are tracked individually, comparing versions overall (i.e., see the differences between different test versions) is not yet possible
- Restoring to a specific "version in time" (i.e., like a snapshot) or to a specific change is not possible because the chosen approach for test versions follows a different concept
- It's not possible to say that version X of the system/project uses a specific version of test cases, as each test version is part of each Test

Test Runs and Test versions

No matter the approach you're choosing for managing test versions (the Jira/Xray Standard way or the Xray Enterprise way), it is still important to note that Xray makes a copy of the Test specification whenever you schedule a Test for execution in a Test Execution.

- A Test Run effectively contains the Test specification of the Test version chosen when the Test Run was created (at that moment).
- You can go to an already recorded Test Run and reset the specification to the current specification on the Test version which originated the Test Run. Or, you can merge it, which means that only the changed steps will be updated, and the other step results are not changed. More info can be found on the Execute Tests page.

By persisting the Test specification on the Test Runs, Xray ensures data consistency. The version of the Test executed in a Test Run is persisted in the Test Run itself. That means that if you change a Test specification today, that won't affect your already recorded runs for that Test (unless you want to).

Learn more

Please have a look at Test Runs for more information on the fields that are persisted at the Test Run level.