Tracking Issues with JIRA

ONAP Bugs, Epics, Stories, and Tasks are tracked in the JIRA system at https://jira.onap.org/.

Note that you do not need to log into JIRA to view issues; however, your view will be read-only and not all fields described below will be visible.

Get a Linux Foundation Account

If you intend to file or update information in JIRA, you will need a Linux Foundation identity to access the ONAP JIRA account. Read this page for instructions.

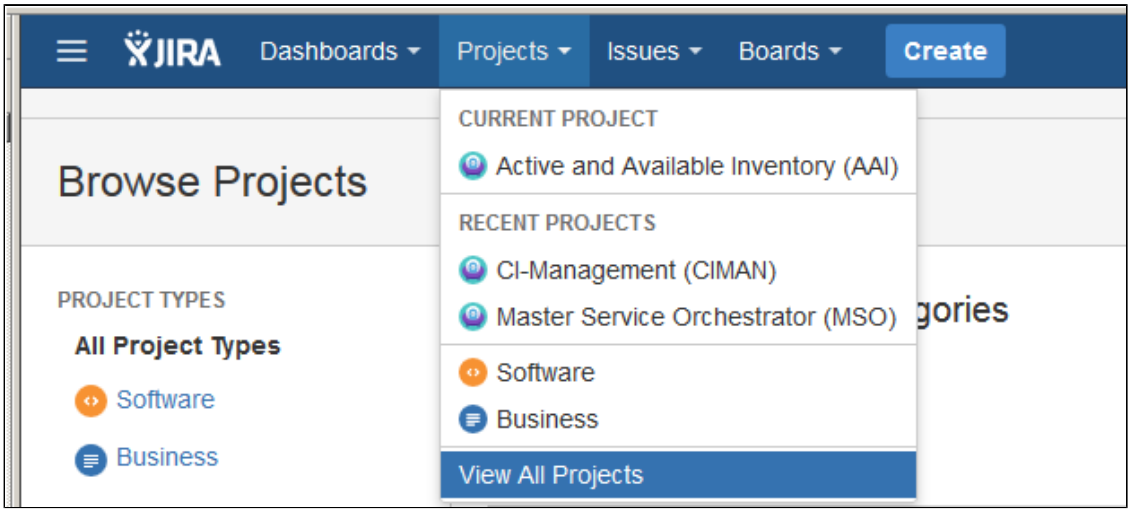
JIRA User's Guide

If you are not familiar with JIRA, you can refer to JIRA User's Guide provided by Atlassian: https://confluence.atlassian.com/JIRA064/JIRA-user-s-guide-720416011.html

JIRA Setup for ONAP

ONAP defines a number of different projects for the various subsystems.

To view the current set of JIRA projects, go to the Projects menu, select View All Projects from the pull down menu to see what is currently defined.



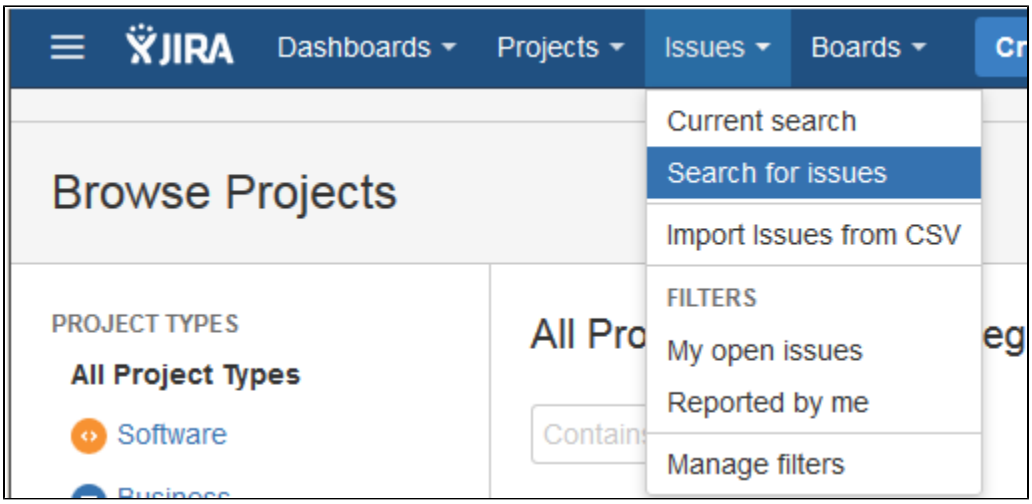
Below is the current set of Projects; however, note that additional JIRA projects may be added as needs dictate, so this list is not static

	Project	Key
	Active and Available Inventory	AAI
	Application Controller	APPC
	Certification of Platform and VNF Implemenations	CERT
	CI-Management	CIMAN
	Common Platform and CI/CD Capabilities	COMMON
	Data Collection, Analytics, and Events	DCAE
	Documentation	DOC
	Master Service Orchestrator	MSO
	Network Controller	SDNC
	Policy Framework	POLICY
	Portal	PORTAL
	Service Design and Creation	SDC
	Test	TEST
	Use Case Analysis	UCA
	Virtual Infrastructure Deployment	VID

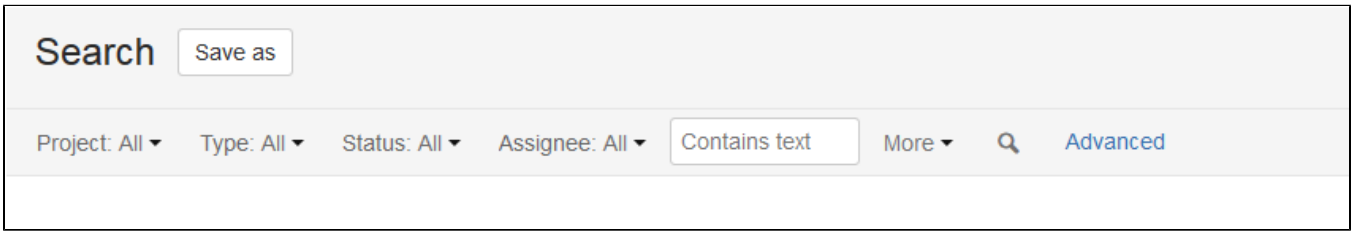
Viewing Issues in JIRA

There are many ways to view issues in JIRA. The description here is one way.

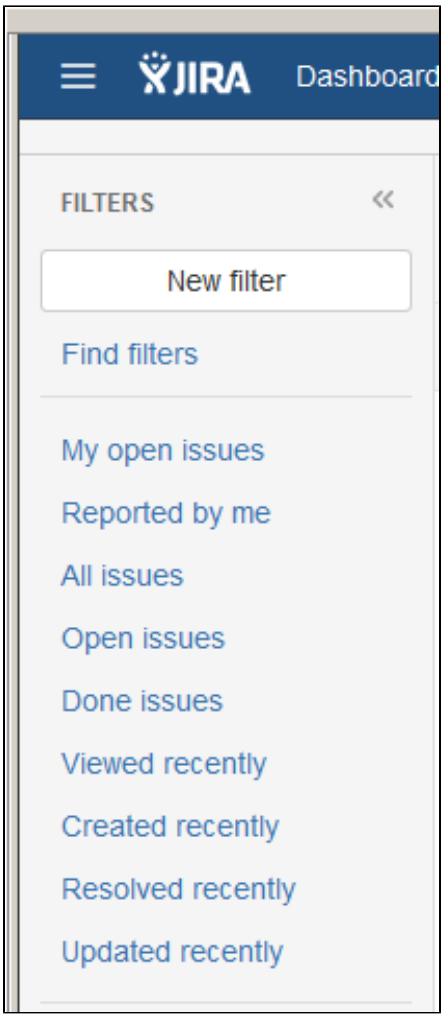
Go to Issues menu, select *Search for issues*



This will bring up the following screen. You will be able to select which Project(s) you want to search, what issue Types (Bug, Epic, Story, Task), what Status and so on. The "More" menu allows you to select additional fields to search on to further narrow down your search.

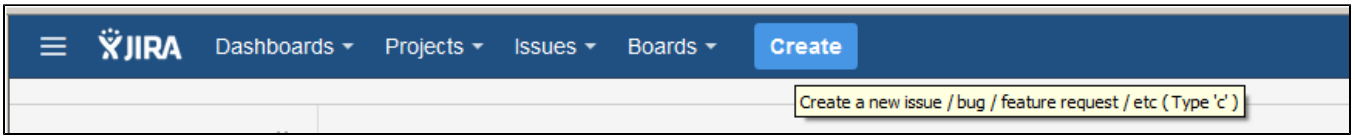


An alternative to searching, you can use predefined filters provided by JIRA as shown below:



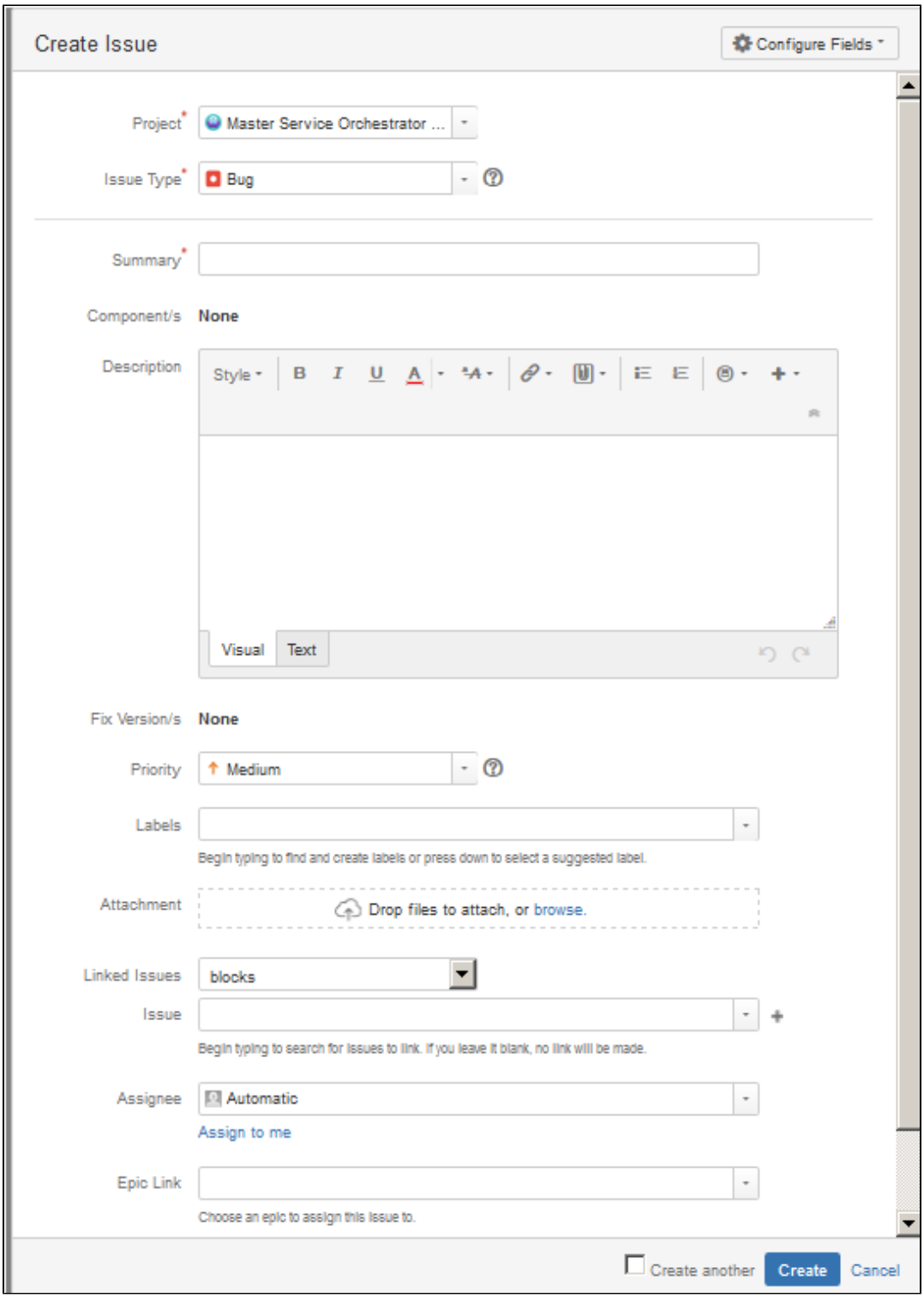
Reporting a Bug

To report a bug against ONAP, select Create (please note that screen display may vary slightly depending from where in JIRA you create the bug)



The following Create Issue screen will be displayed.

- Select the Project against which you want to open the Bug
- Ensure that you select the Issue Type "Bug"
- Provide a concise description of the error
- Select Component if the project provides greater granularity on component breakdown
- Provide a detailed description of the issue
- Attach any supporting information (logs, stack trace, screen shots, etc…)
- Assign the bug to yourself if you plan to fix the issue; otherwise, leave Assignee blank
- Select Create at bottom of screen

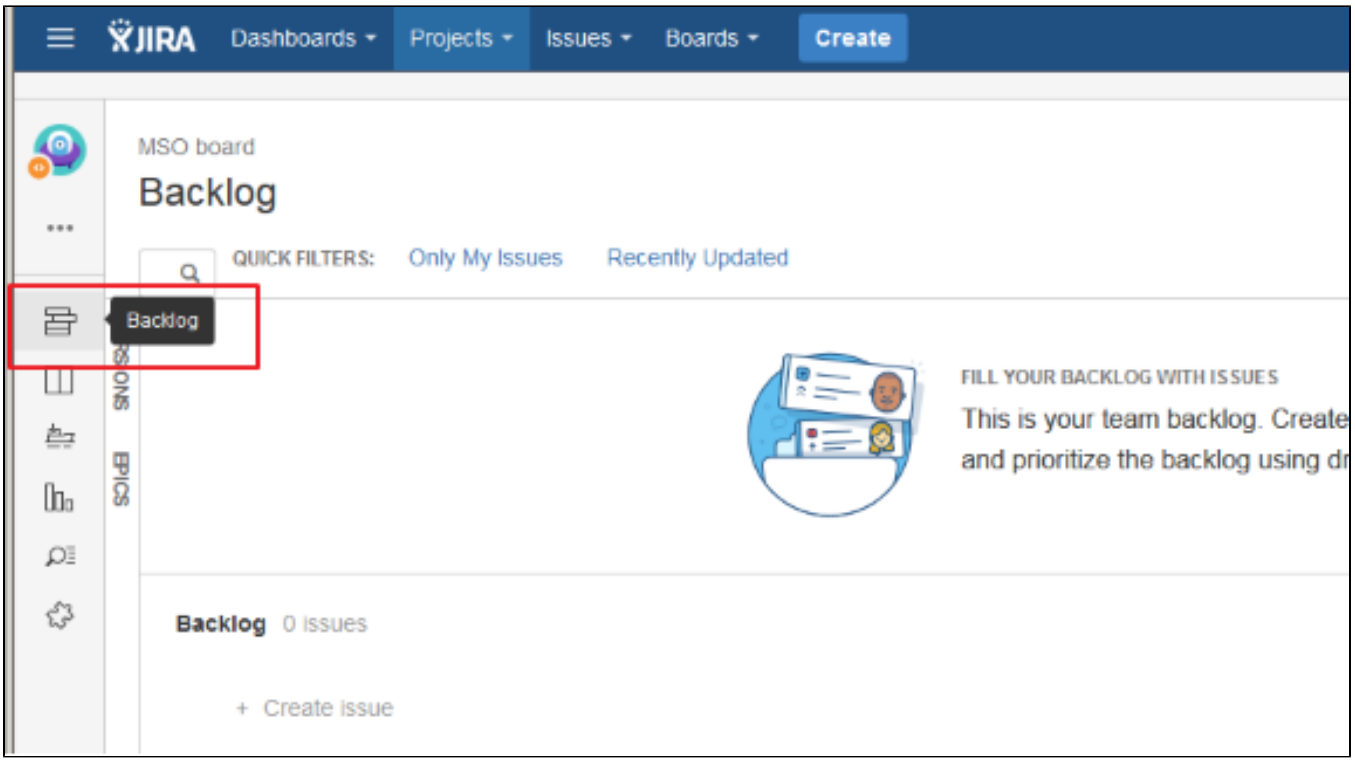


Proposing a New Feature

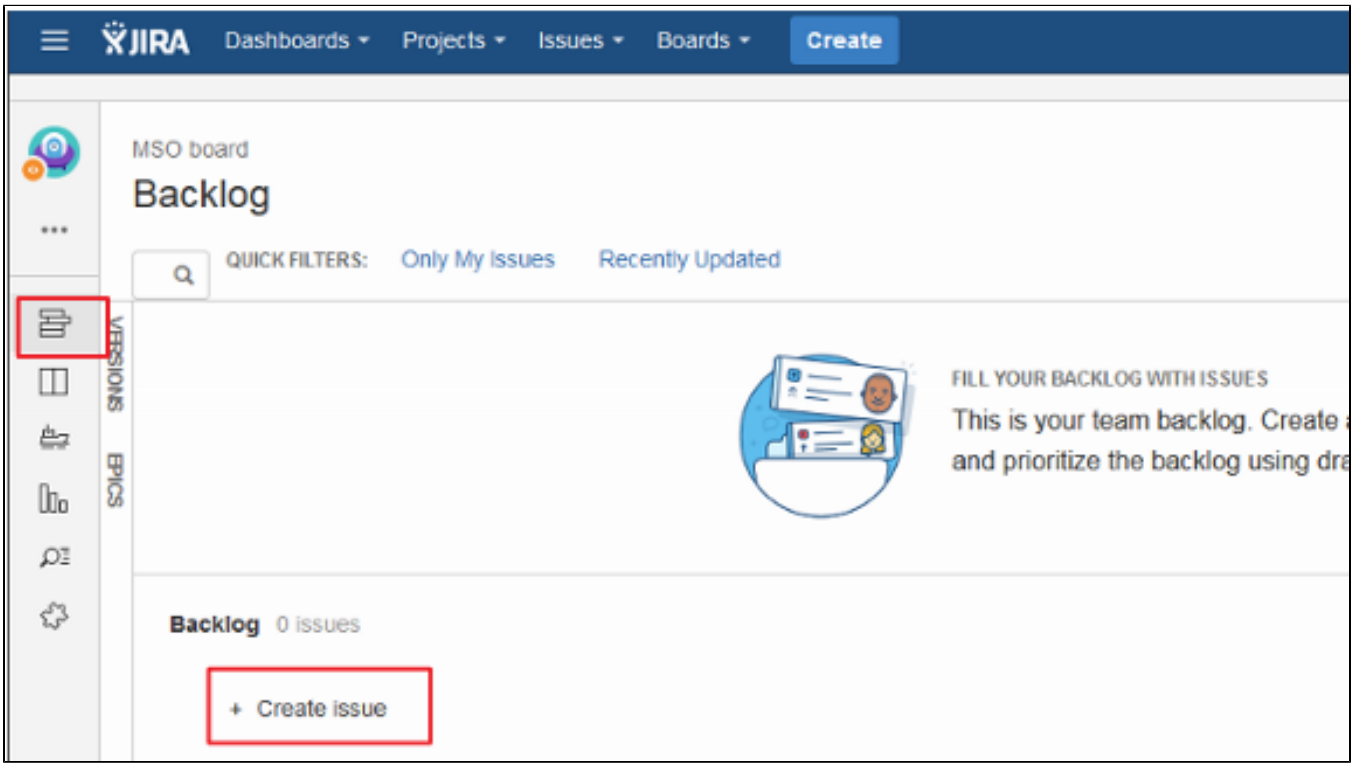
New feature or enhancement proposals are submitted via JIRA using the Story issue type.

Before you submit your idea, first check to see if something similar already exists in the backlog. If not, go ahead and create a Story.

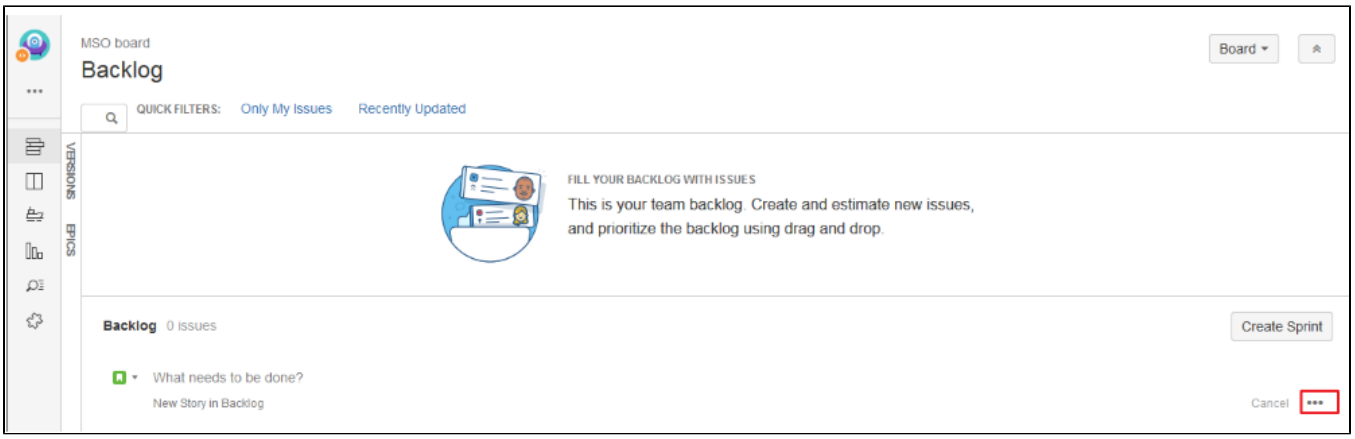
Searching for a Story in the backlog works the same way as searching for a Bug (Viewing Issues in JIRA), just select Issue Type Story instead of Bug along with the Project(s) of interest in the Search menu. Alternative, you can go to the Backlog board for a particular project, such as shown below for MSO.



To submit your feature or enhancement proposal, go to the Backlog board of the applicable Project and select Create Issue. In our example, we are using MSO.

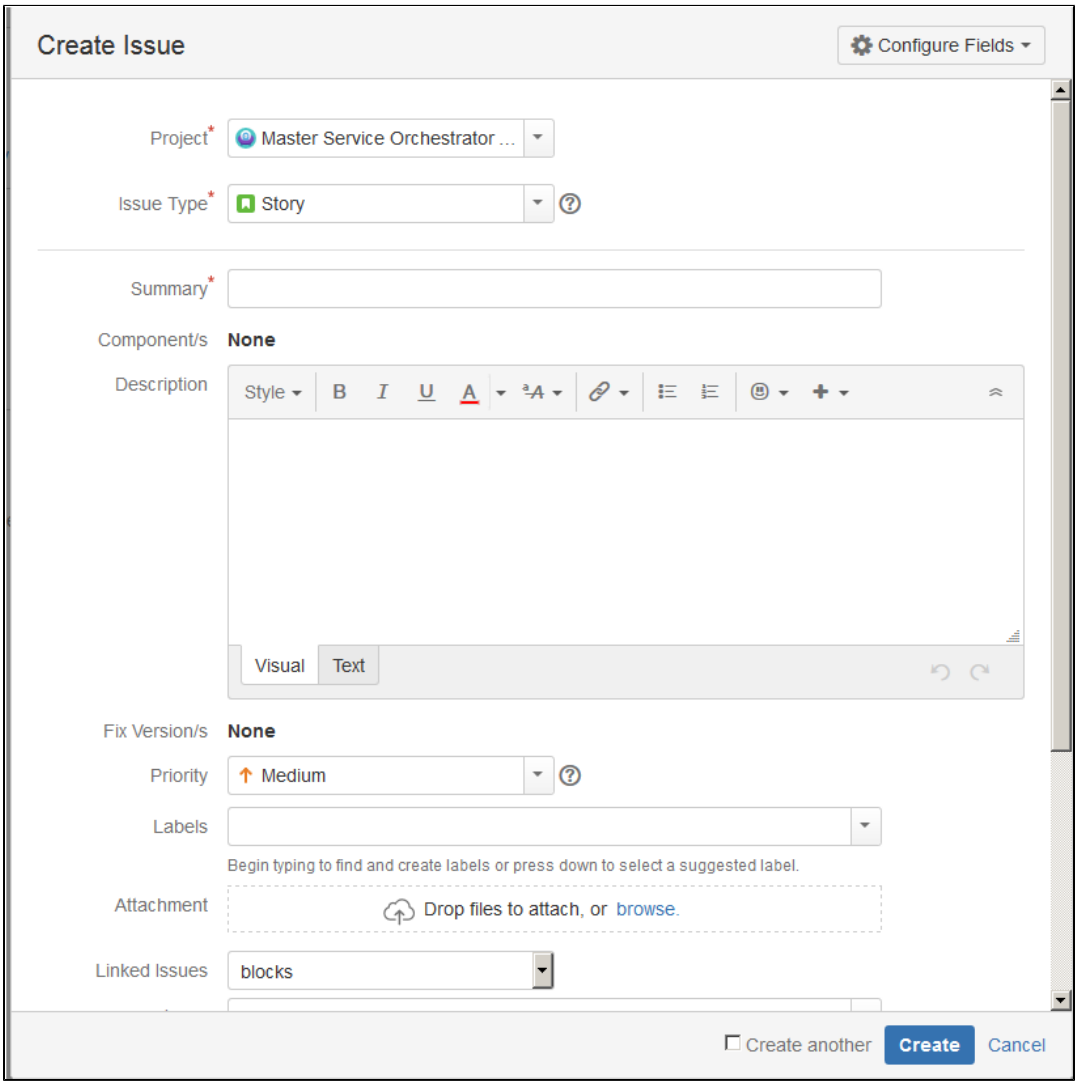


This will present the following menu, go to the far right and click on the three dots.



This will bring up the Create Issue screen as show below.

- Provide a clear and concise Summary of the new feature
- Provide detailed Description of the new feature
- Select Create

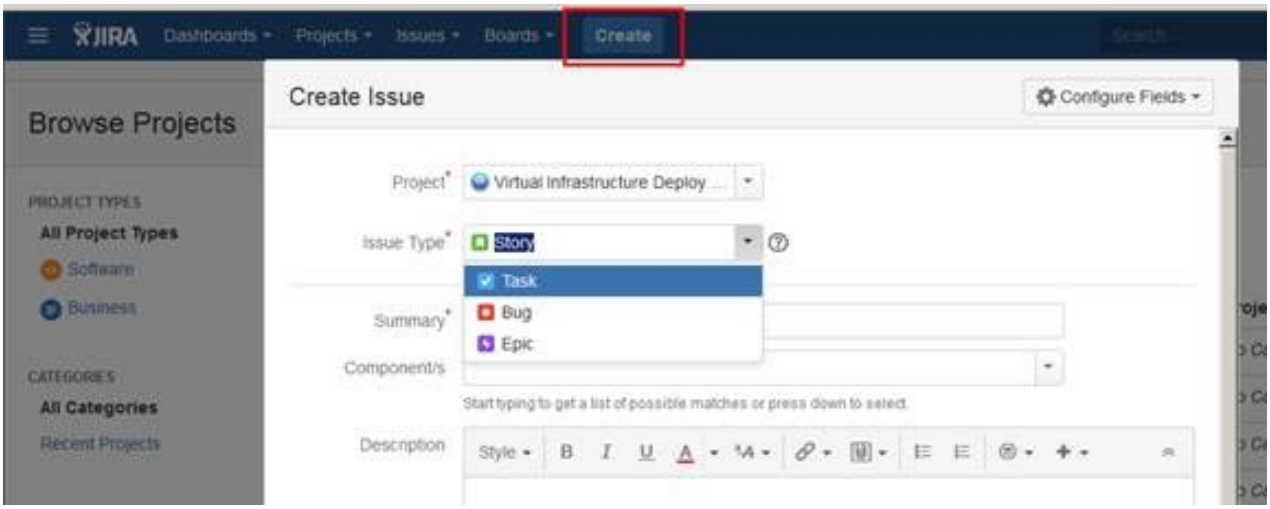


JIRA Issue Types

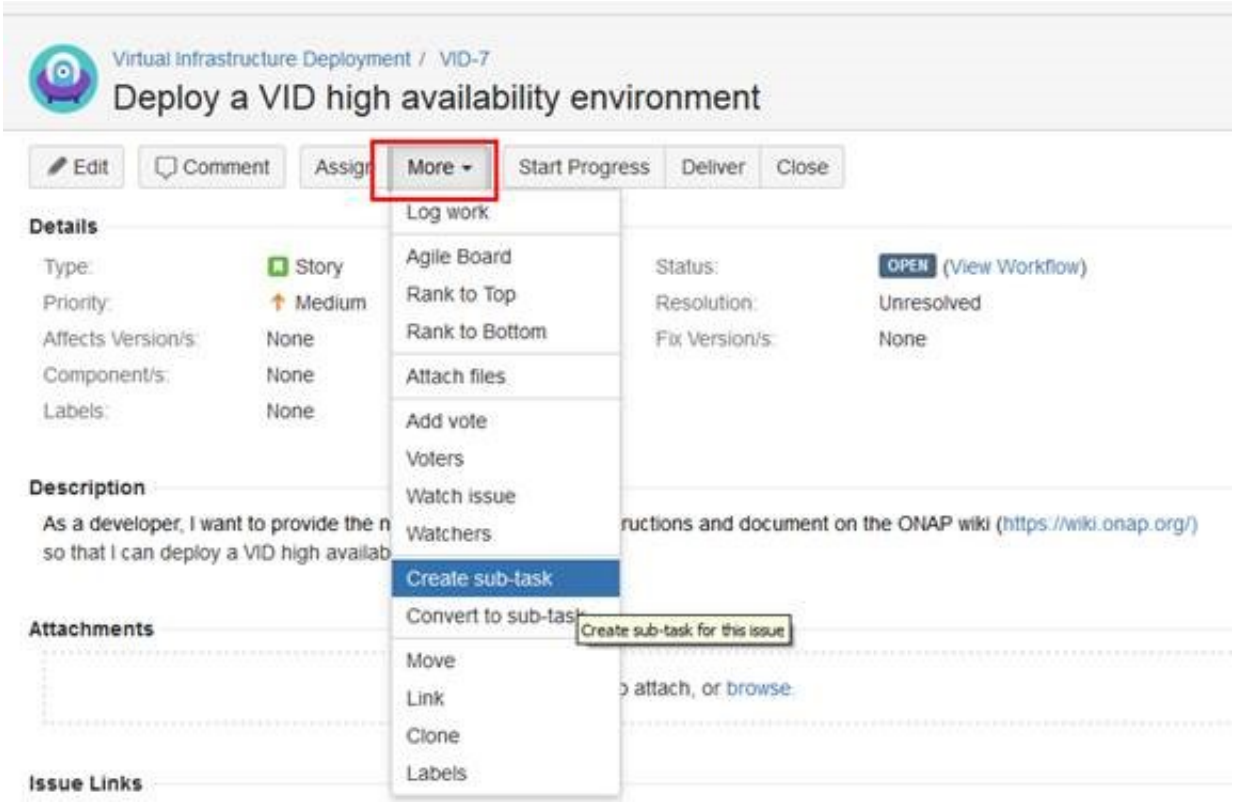
There are generally 5 Jira issue types used within ONAP; the 5th one, Sub-task, can only created from within a Story.

1. Epic
    - An Epic is used when a feature is large in scale, may take several sprints to complete, and/or may spans multiple components.
    - The Epic itself is an umbrella issue. An Epic is comprised of one or more stories.
    - Usage: Anyone can create an Epic and put it in the backlog for a project, but is usually created by PTLs
2. Story
    - A Story or user story is a piece of work that can be completed within a sprint. It is usually part of an Epic.
    - Usage: Anyone can create a Story and put it the backlog for a project.
3. Bug
    - A Bug is a defect, an error, or a flaw that requires a fix, it can be a code change, a documentation update, configuration change, etc…
    - Usage: Anyone can create a bug and assign to himself or someone else or leave it unassigned if they wish someone else to pick it up and work on it
4. Task
    - Task can be used to breakdown a story into smaller work units that can be completed in a day or two; however, for deconstruction of a story, it is recommended that a Sub-task be used. Using a Sub-task will save the user Jira steps and will be automatically linked to the story since it's created from within the story. See Sub-task below.
    - Tasks can be used to track discrete activities that must be done and may not necessarily be related to a story (for example, fix a Jenkins job); however, if a user chooses to use Task issue type to track work associated with a story, be sure to link the Task to the Story via the Linked Issues and Issue fields as illustrated below.
    - Usage: Anyone can create a task and assign to himself or someone else

Example of screen shot for creating a JIRA issue via the Create menu option.

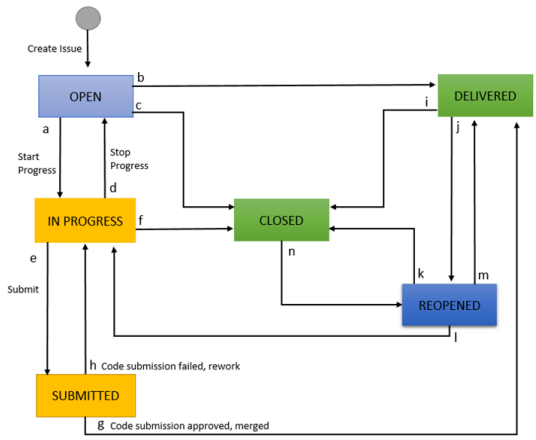


5. Sub-task
    - Sub-task is used to breakdown a story into smaller work units that can be completed in a day or two. Breaking a story down into discrete work units that can be burned down daily will communicate progress on the story.
    - Sub-tasks are created from within the story via the More menu option by selecting *Create sub-task*



JIRA Workflow

The ONAP projects have all aligned on the following workflow, which is a slightly modified Atlassian Classic workflow.



JIRA Statuses and supported transitions

	From Status	Transition		To Status
a	Open	Start Progress		In Progress
b	Open	Deliver		Delivered
c	Open	Close		Closed
d	In Progress	Stop Progress		Open
e	In Progress	Submit		Submitted
f	In Progress	Close		Closed
g	Submitted	Deliver		Delivered
h	Submitted	In-Progress		In Progress
i	Delivered	Close		Closed
j	Delivered	Reopen		Reopened
k	Reopened	Close		Closed
l	Reopened	Start Progress		In-Progress
m	Reopened	Deliver		Delivered

n	Closed	Reopen		Reopened

JIRA Resolution Code

The Resolution field is very important because combined with Status field, it conveys critical information to the community about the general disposition of the Jira issue.

The below list is the set of Resolution options provided as defaults by Jira. Some projects have added to this list, such as Not A Bug, PTLs can request LF to add more Resolution options to their project; however, recommendation is that one set be defined that is used across all projects - this is a work in progress still.

- **Unresolved** (by default)
- **Done**: Issue is fixed, closed
- **Cannot Reproduce** (self explicit)
- **Duplicate**: Provide duplicate issue number
- **Won't Do**: Explain reasons why the issue will never be fixed

JIRA Version

- Affects Version/s: Project version(s) for which the issue is (or was) manifesting - this field is really used for Bugs, think of it as the "Found In" version of the software.
- Fix Version/s: Project version(s) for which the issue is (or was) fixed or a story is delivered.

JIRA Priority Definition for Bugs

Use this guideline to setup the mandatory "JIRA Priority" field. Note that the PTL has jurisdiction over this criteria and may thus requalifies the priority.

1. **Highest**: Catastrophic defect that causes total failure of the software or unrecoverable data loss with no available workaround. Complete shut-down of the process, nothing can proceed further. Blocks testing of the product / feature.
    a. Must be fixed in the next build.
2. **High**: Severely impaired functionality. Erratic performance and reduced system availability. It may cause major components or features of the system to be unavailable although certain parts of the system remain functional. A workaround may exist but its use is unsatisfactory or at a high time cost (manual intervention required).
    a. Must be fixed in any of the upcoming builds and should be included in the current release.
3. **Medium**: Failure of non-critical aspects of the system but results in an inconvenient situation. It causes some undesirable behavior, but the system is still functional. There is a reasonably satisfactory workaround. It is still a valid defect that should be corrected
    a. May be fixed after the release or in the next release.
4. **Low**: It won't cause any major break-down of the system. It does not result in the termination of services, does not impact usability of the system and the desired results can be easily obtained by a workaround.
    a. Fix can be deferred until after more serious defect have been fixed.
5. **Lowest**: No impact to the functionality. More related to the enhancement of the system.
    a. May or may not be fixed at all

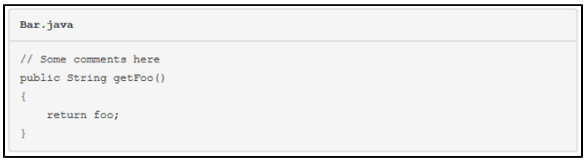
Recommendations for writing Proper JIRA Bug Issue

Do and Don't

- 1 record per Jira Issue. Do not report multiple bugs within one single JIRA Issue.

  Otherwise, it makes it very difficult:

    - to have a focused discussion
    - to track progress
- Even if you encountered several issues at once, you should open a separate JIra Issue for each and Link issues together.
- Keep ongoing communication in JIRA. Email do not scale. Email are lost. Use JIRA to provide all necessary Logs, code sniptets, screenshots.
- In case you need to get notified on a change: Use JIRA Watch capability.
- Assignee: By default, while creating a JIRA Issue, the PTL is assigned. Issue can be re-assigned by anyone. (we need to discuss best practice for assigning issues and come to consensus)
- Even if status is Closed, issue can be re-opened. Nothing is preventing to re-open a closed JIRA Issue.



- The JIRA Issue Summary should precisely describe the problem. "X is broken" is not a helpful description. "X is doing Y instead of Z" is much better.

Description Field

- Provide more details on the issue
- Describe the error you see, and describe the behavior expected
- Provide steps to reproduce the error
- If possible, provide actual commands to run
- Attach logs or screenshots
- Try to reduce the problem to the smallest possible use case to make it easier to reproduce the bug and to test the solution
- Embed code in the description and use formatting
- Provide info on: OS, Browser, Java version,…