Begin planning with JIRA
- EPICs, Stories, Subtasks

Jan-Simon Möller

training.linuxfoundation.org
Topics

- Basic Rules
- What are EPICs, (User-)Stories, Sub-Tasks
- Guidelines, Best Practices
- How to create EPICs, Stories, Sub-Tasks
- How to enter the data for the upcoming release
Basic rules
Basic rules

• Use JIRA as the only place to plan the project work.
  → It will make it transparent to all participants
  → In your project, but also to related projects
  → Progress and backlog allows coordination
Basic rules

• Use same JIRA issue types across all projects;

• OPNFV uses:
  – Epic
  – Story
  – Subtask
  – Bug
Basic rules

• Use Stories for planning the work items
  → Coordination with fellow projects
  → Define relationships (depends/blocks)
Basic rules

- Define clear Acceptance Criteria / Definition of Done
  - e.g. list of activities that add verifiable/demonstrable value to the product
    - written code
    - code comments
    - unit tests, integration tests
    - release notes
    - design documents
    - etc.
What are EPICs, (User-)Stories, Sub-Tasks

(OPNFV style)
EPICs, Story, Sub-Tasks

- Agile Development / Scrum
- Issue Types
- Different abstraction levels
- Used to compose the Backlog of the development cycle
EPICs

• >> An epic captures a large body of work. It is essentially a large user story that can be broken down into a number of smaller stories. It may take several sprints to complete an epic. <<

• big, coarse-grained

• large themes, overall goals
User Story

• >> A story or user story is a software system requirement that is expressed in a few short sentences, ideally using non-technical language. <<

• Non-technical language, expresses user needs

• „As a <user> = who
  I want to <be able to do XYZ> = what
  so that <ABC can be done>“ = why
Subtask

- A sub-task issue is typically one of many tasks that make up a story.
- Contains description of individual work item (detailed, technical)
- Breaks-up stories into digestable workloads
- Can be shared, assigned, relate to, depends/depended on by, blocks/blocked by
Guidelines, Best Practices (0)

- Check for existing issues first
  - check the project - the issue might exists already
  - avoid duplicates!
Guidelines, Best Practices (1)

- Write good
  - subject
  - description
  - definition of done (acceptance criteria)

→ Makes it easier to understand work items
→ Makes it easier to align/share work items
Guidelines, Best Practices (2)

• Label stories in order to query backlogs easily.
  – OPNFV uses the label **R2** for the Brahmaputra Release
  – Use this label!

→ Makes it easier to query the backlog
→ Keeps track of the workload per release
Guidelines, Best Practices (3)

• Properly progress work items
  – Open → In Progress → Resolved → Closed

  → Gives fair view of what is happening
  → Allows estimate of remaining work / time
Guidelines, Best Practices (4)

• Keep the work item specific conversations on JIRA.
  – Do not copy to Wiki or Etherpad
  – use links to JIRA

→ Fellow projects can join the conversation
→ Single place to track
→ Decisions, Problems, Solutions documented
How to create EPICs, Stories, Subtasks
How to create an EPIC

• Log in to JIRA

• Create issue

• The „Create Issue“ form appears
How to create an EPIC

• Select the Project
  – e.g. SANDBOX

• Select the Issue type
  – EPIC

• Short Name
  – e.g. „OPNFV Tutorials“

• Summary
  – e.g. „OPNFV Tutorial series about developer tools“
How to create an EPIC

- Add Description state:
  - Development goal
  - Requirements
  - Definition of Done

A series of tutorial videos shall be developed.
The tutorial shall:
- cover the development tools
- come with slides
- come with examples
Definition of Done:
- Videos uploaded and slides posted.
How to create an EPIC

• Add Attachments
• Add Labels
  – e.g. **R2** (for Brahmaputra)
• Click on „Create“
How to create a Story

• Log in to JIRA

• Create issue

• The „Create Issue“ form appears
How to create a Story

- Select the Project
  - e.g. SANDBOX

- Select the Issue type
  - Story

- Summary
  - e.g. „Tutorial about EPICs/Stories/Subtasks in JIRA“
How to create a Story

• Add Description state:
  – Task goal
  – Definition of Done

As a user ...
I want to ...
So that ...

As a OPNFV developer I want to know more about EPICs, Stories and Subtasks so that I can start planning for the Brahmputra cycle using JIRA.
How to create a Story

• Add Attachments
• Add Labels
  – e.g. **R2** (for Brahmaputra)
• Link to EPIC
  – „OPNFV Tutorials“
  – Use Drop-Down
  – Or just start typing
    • OPNFV Tutorial
    or
    • SANDBOX-3
• Click on „Create“
How to create a Sub-Task

• Log in to JIRA

• To create a Sub-Task for a story first open the story within JIRA.

As an OPNFV developer, I want to know more about EPICs, Stories, and Sub-Tasks so that I can start planning the Brahmaputra cycle using JIRA.
How to create a Sub-Task

• To create a Subtask for a story first open the story within JIRA.
• Click on „More“
• Select „Create Sub-Task“

• A new form will appear
How to create a Sub-Task

- Write the Summary

Create Subtask: SANDBOX-2

- Issue Type: Sub-task
- Summary: Create Demo Projects for OPNFV tutorial in JIRA
- Priority: Major
- Component/s: None
- Affects Version/s: None
- Assignee: Automatic
How to create a Sub-Task

• Add the Description

Two projects need to be created in JIRA:
- EXAMPLE
- SANDBOX

Assign ownership to XYZ, ABC

Definition of Done:
Project created
Owners assigned
How to create a Sub-Task

- Add Attachments
- Add Labels: e.g. „R2“
- Click on „Create“
How to enter the data for the upcoming release
Typical workflow for planning

• Create project high level plan using EPICs
  – As a project/team, start writing high level Epics in order to plan your work/deliverables
  – Epics help you to group related stories under them to improve the visibility and traceability.
  – Epics can also be used to give a high level roadmap regarding your project.
Typical workflow for planning

• Write your user stories and plan your work
  – Write good subject, description and definition of done/acceptance criteria
  – Write user stories from user perspective
  – „As a user, I want to ... , so that ... “
Typical workflow for planning

- Label the stories that are in scope of the release
  - Use **R2** for Brahmaputra
Typical workflow for planning

- Highlight the dependencies between Epics and/or Stories
  - show what dependencies you have within your project
  - Use JIRA **Link** feature to create associations
Typical workflow for planning

• Highlight the dependencies between your Epics and/or Stories and other projects.
  – Important – shows external dependencies
  – Use JIRA Link feature to create associations
Typical workflow for planning

• Prioritize the EPICs/Stories/Sub-Tasks
  – Project priorities are visible to other Projects which might depend on it
  – Available levels:
    • Blocker - e.g. blocks development, test, production
    • Critical - e.g. crash, loss of data, memory leak
    • Major - e.g. Major loss of function
    • Minor - e.g. Minor loss of function
    • Trivial - e.g. Cosmetic problem or spelling
Typical workflow for planning

• Done
• Backlog ready and you can start to develop
• Regardless which workflow your project uses (this is up to the projects),

track progress in JIRA!
Task

• Create an EPIC in the SANDBOX project
• Create a Story in the SANDBOX project as part of your just created EPIC.
• Create two Sub-Tasks to your Story. Make one Sub-Task depend on the other.
References

- http://www.romanpichler.com/blog/10-tips-writing-good-user-stories/
- https://www.scrumalliance.org/community/articles/2008/september/what-is-definition-of-done-(dod)
- https://confluence.atlassian.com/display/AGILE/Product+Backlog
- https://confluence.atlassian.com/display/AGILE/Creating+an+Epic
- https://confluence.atlassian.com/display/AGILE/Story