JIRA Workflow Proposals

JIRA contains the entire project history, but as the project has matured our use of JIRA has not kept up. This page proposes some changes to the structure of our JIRA project, to capture more information, simplify the data entry and nudge people towards more complete and accurate data entry. This will better allow us to measure release quality over time and identify when Cassandra is ready for (or due a) release.

Summary

- Removal of issues types: Wish and Test
- Fields
    - Removed: Reviewer, Environment, Reproduced In, Docs Text, Due Date, Epic Link, External Issue Id, External Issue URL, Flags, Sprint, Time Tracking
    - Modified: Priority, Component
    - Added: Complexity, Feature, Impacts, Test and Documentation Plan, Platform
    - Added (Bug only): Severity, Bug Category, Discovered By
    - Added (Improvement and Feature only): Change Category
- Workflow
    - New States: Triage, Review in Progress, Change Requested
    - Transitions with required fields
    - Order of field display changed
- Issue Permissions
    - Anybody may file a Triage ticket
    - Contributor role will be removed, and in all places replaced with jira-users.
    - Only jira-users will be permitted to transition a ticket in the workflow
- Schema Permissions
    - Only committers will be permitted to introduce new options to the schema for the fields: Component, Feature, Platform. Removals can be negotiated with the person who introduced them, or litigated on list.
    - All other fields should not have their schema-defined options modified without endorsement from the mailing list.

Removals

To simplify the maintenance of JIRA, it would help to remove any unnecessary concepts. Mostly these are concepts we do not use in practice, except very spottily (so as to make them useless).

Remove Issue Types

Firstly, we propose the removal of the Wish and Test ticket types.

Issue Type	Reason
Wish	is a feature/improvement, and can be better communicated via priority/complexity details we will introduce below
Test	is logically a component, and a non-specific one at that

Remove Fields

Field	Reason	Migration
Reviewer	Replaced by Reviewers	Populate empty Reviewers fields with contents of Reviewer
Environment	Use is very patchy, noisy, and seemingly of little value over a comment if the extra content is useful	Propose new multi-select 'Platform' field with curated option list, in detail below. We can insert a comment with the environment text for any tickets containing it presently.
Reproduced In	Use is very patchy; seems to offer little practical value above 'Since Version' or a comment.	
Docs Text	Unused	
Due Date	Unused	
Epic Link	Unused	
External Issue ID	Unused	
External Issue URL	Unused	
Flags	Unused	

Sprint	Unused	
Time Tracking	Unused	

Migrate Labels

All labels that provide utility to the project and can be represented in the new schema should be migrated to the new schema, but the original labels will be left intact.

remap cassandra jira labels.csv

Modifying/Expanding Existing Fields

Priority, Complexity, Severity and Category

Presently, Priority encodes some ill-defined combination of the first three of these independent properties, making it hard to draw strong conclusions about any of them. We propose introducing two new fields, and clarifying the Priority terminology to only suggest urgency, as well as reducing the number of priorities, since we don't effectively use them.

Priority

New Priority	Logically Replaces	Migrate From	Details
Low	Low-Minor	Low-Minor	
Normal	Minor-Major	Major	It wasn't clear what Minor or Major meant, but priority is a relative concept - a 'normal' is our logical baseline, and the default for a new issue
High	Rest of Major		For tickets that community members plan to prioritise over other outstanding work, but has no immediate urgency
Urgent	Critical-Blocker	Critical-Blocker	There's limited logical distinction between Critical and Blocker; it seems to make more sense to have a single 'do ASAP' tag. This should be limited to issues that should both block (until completed) and accelerate (once completed) the next release.

For the Bug type, this field will be auto-populated (if possible).

Complexity

This field will be required, discussed further in the Workflow section below.

Complexity	Description	Initial Value
Low Hanging Fruit	Trivial, No Dependencies, Localised, Accessible to New Contributors	Old Priority = Trivial or label=lhf
Normal	Unremarkable; default	
Challenging	Localised, but requiring sophisticated analysis to understand	
Byzantine	Affecting many components, requiring sophisticated analysis to understand	
Impossible	Suspect this is not even possible (may be paired with Wish)	

Bug Only Fields

Severity

This field will be required, discussed further in the Workflow section below. It will be available only for the Bug issue type.

Severity	Old Priorities	Description
Low	Trivial-Low	Limited usability or low visibility impact with no impact on correctness; no urgency to resolve
Normal	Minor-Major	This issue is either unremarkable or extraordinarily rare
Critical	Critical-Urgent	This bug may have significant impact on correctness, availability, stability or some other critical production behaviour

Discovered By

This field will be required for the Bug issue type. It will be used for analysis of our efforts to establish project quality.

- User Report
- Code Inspection
- New unit/dtest
- Performance Regression Testing
- Fuzz Testing
- Workload Replay Testing (e.g. FQL)
- Shadow Traffic Cluster
- Adhoc Testing

Bug Category

Required. Only provided as an option for the Bug issue type. Uses a Cascading Select List.

Category	Subcategory	Description
Correctness	Persistent Corruption / Loss	Corruption that persists, and may propagate across the cluster
	Response Corruption / Loss	Corruption that does not propagate or persist, only results in a client receiving erroneous responses
	Semantic Failure	The logical behaviour is either not to spec, or the spec is faulty/ambiguous
	Consistency Failure	Apparently successful action, but with lower consistency than required
	Test Failure	A test is broken - if this turns out to be a legitimate bug, it should transition to the bug's category once diagnosed
Availability	Response Crash	An operation does not succeed/respond because of a crash while servicing it, without affecting process stability
	Process Crash	An isolated exceptional state occurs that brings down the affected node
	Cluster Crash	A correlated exceptional state occurs across the cluster, bringing down a multiplicity of nodes
	Unavailable	Apparently unavailable, when should be available
Degradation	Resource Management	Either a resource leak or overcommit
	Slow Use Case	A specific use case with suboptimal characteristics that have not yet been accommodated
	Performance Bug /Regression	Unintended performance behaviour, including e.g. exceptions stalling compactions
	Other Exception	An exception is being thrown, that is not coinciding with another category of degradation
Security	Information Leakage	
	Privilege Escalation	
	Denial of Service	
	Remote code execution	

/Bug Only Fields

Change Category

Required. This is the only field unique to the Feature and Improvement issue types.

Category	Description
Performance	
Change Semantics	Introduce new, or clarify/modify existing database semantic behaviours
Improve Operability	Reduce the burden of operating a cluster (i.e. handle uncommon states better, with less operator involvement)
Quality Assurance	Work to improve the guarantees we can make about the stability and correctness of Cassandra

Component

Presently, the meaning of each component is unclear, even to long-serving project members. As such, it is very inconsistently used and probably of limited value for analysis. We propose a more granular definition of components that more closely matches the every day project vernacular.

The biggest difficulty here will be migration. It might be that a "Legacy" component is the best option, with the old schema replicated exactly. We can then manually migrate tickets as the value presents itself, or organise such a transition.

Multi-select List

```
Consistency/Coordination
Consistency/Hints
Consistency/Repair
Consistency/Streaming
Consistency/Bootstrap and Decommission
Consistency/Batch Log
Cluster/Membership
Cluster/Gossip
Cluster/Schema
Local/Commit Log
Local/Memtable
Local/SSTable
Local/Caching
Local/Compaction
Local/Compaction/DTCS
Local/Compaction/TWCS
Local/Compaction/LCS
Local/Compaction/STCS
Local/Config
Local/Startup
Local/Shutdown
Local/Scripts
Messaging/Internode
Messaging/Native v4
Messaging/Native v5
Messaging/Thrift
CQL/Syntax
CQL/Interpreter
Observability/JMX
Observability/Metrics
Observability/Tracing
Observability/Logging
Tools/fql
Tools/cqlsh
Tools/nodetool
Tools/sstable
Tools/bulk load
Tools/stress
Tests/dtest
Tests/unit
Tests/fuzz
Tests/benchmark
Docs/Javadoc
Docs/Website
Docs/Blog
Packaging
Dependencies
Build
```

Feature

Features tend to cut across many components of the database. So an orthogonal field to track these, instead of ill-defined labels is probably of utility. It is any way useful to track things like bugs per component/feature combination.
This field will not be required, since many tickets will not touch on features explicitly.

- Lightweight Transactions
- Counters
- Change Data Capture
- Transient Replication
- 2i Index
- SASI
- Materialized Views
- Virtual Nodes
- Virtual Tables
- Authorization
- Encryption
- Compression
- KMS/Vault
- Super Columns
- UDF
- UDT
- UDA

Other New Fields

Platform

To replace the existing Environment field that is of limited value.
A curated list, that can only be modified by project members. Initially seeded with:

- Java {7,8,9,10,11}
- OpenJDK, Oracle Java, Azul, ...
- Linux (major kernel versions), Windows, OpenBSD, ...
- x86, ... (added as necessary)
- NVMe, SSD, Magnetic HDDs
- AWS, GCE, Azure

Impacts

To replace certain labels, and help external maintainers track features of relevance to them.
A curated list, that can only be modified by project members. Initially seeded with:

- Clients
- Docs
- Security
- JDBC
- Spark
- Hadoop

Test and Documentation Plan

A new required field containing free-form text field, required when transitioning to 'In Progress'.
The intended purpose is to encourage explicit upfront consideration of the work needed on these areas
either before or following commit. This may entail filing follow-up tickets that need to be resolved before release,
or a brief statement on the tests that will be written, or simply 'n/a'.
This also provides a promise to hold implementors to before release, and a point of discussion before a ticket lands.

Workflow

To encourage high quality data entry and better observability, we propose a few changes to the project workflow. We propose:

1. Introducing some new issue states to better track the current ticket status, and handover between responsible parties;
2. Making certain fields required during certain state transitions, to ensure we have the minimally necessary ticket information complete at each stage'
3. Reordering the fields that we display on transition, to highlight the most important

New 'Triage' State

Currently it is easy for the project to miss a ticket, and for that ticket to fall through the cracks indefinitely.
At the same time, user reports cannot be expected to fill out all of the required fields accurately.
It's proposed that we introduce a new initial state named 'Triage' that has no required fields, and that anybody may file.
To transition to the Open state, you must be a contributor in JIRA (equivalent to able to assign tickets), and must ensure the required fields have been correctly filled out before doing so.

New 'Review in Progress' and 'Change Requested' States

Presently there is no way to indicate that a patch is under active review, so it is hard for assignees to monitor the progress of their patch to completion. Similarly, there is no useful way for the reviewer to indicate that their comments are ready to be addressed by the assignee. With the introduction of these two states, there is a clear handover at each stage of the process. This makes it clear who is responsible for taking the patch forward to the next step, as well as transparency over progress on any steps you are waiting on.

Removal of Reopened State

The Reopened state is of dubious value - arguably, it is in any scenario more helpful to file a new ticket, link the two via a relation, and leave a comment on the original for interested parties to migrate to the new discussion. In any case, its use is frowned upon and rare. It will of course remain possible to move a ticket from the 'Resolved' state to e.g. the Open state, but this will not be officially sanctioned except when correcting filing/procedural errors.

New Workflow

State	Description	Expected Transitions (To)
Triage	This ticket has been filed, perhaps by a member of the community, but has not been considered by a contributor competent to assess its impact, severity, etc. Before transitioning to Open, the contributor should consider updating the title and summary to best reflect the report in a way the project will understand.	Awaiting Feedback, Open, Resolved
Awaiting Feedback	Most beneficial as a cyclical state between Triage and itself, as dialogue takes place to establish any facts needed to understand, categorise and prioritise the report.	Triage, Open, Resolved

Open	The ticket is prioritised and well summarised, but work is not yet underway.	In Progress
In Progress	The assignee is 'actively' working on this ticket	Patch Available, Open
Patch Available	The assignee has a patch that is ready for a reviewer. The assignee should endeavour to solicit from the community a reviewer competent in the subsystem(s) from, if none is already assigned.	Review in Progress
Review in Progress	The assigned reviewer is 'actively' reviewing the available patch	Change Requested, Ready to Commit
Change Requested	The reviewer has provided feedback for the assignee to consider and incorporate into their patch. Once they are ready to address these points, they should transition the ticket back to 'In Progress'	In Progress
Ready to Commit	The reviewer(s) consider this patch to be ready to commit	Resolved
Resolved	The ticket has been closed (either successfully or unsuccessfully)	

The column on the right represents the states we will provide buttons for performing a simple transition between. It does not include all acceptable transitions.

Required fields on transition to 'Open'

- Component
- Feature
- Priority
- Complexity
- Bug/Change Category
- Severity (if bug)
- Discovered By (if bug)

Required fields on transition to 'Patch Available'

- Impacts
- Platform
- Test and Documentation Plan

Required fields on transition to 'Review in Progress'

- Reviewers

Required fields on transition from 'Ready to Commit' to 'Resolved'

- Since Version (if bug)
- Fix Versions

Field Display Order

- Project
- Issue Type
- Summary
- Bug/Change Category
- Discovered By
- Component
- Priority
- Complexity
- Impacts
- Description
- Since Version
- Assignee
- Reviewers
- Test and Documentation Plan
- Tester
- Reporter