

Perspectives

- [Software Agile Delivery Overview](#)
- [Research & Design: Discovery Plan](#)

Software Agile Delivery Overview

Backlog Refinement and Sprint Planning

Adding to the backlog

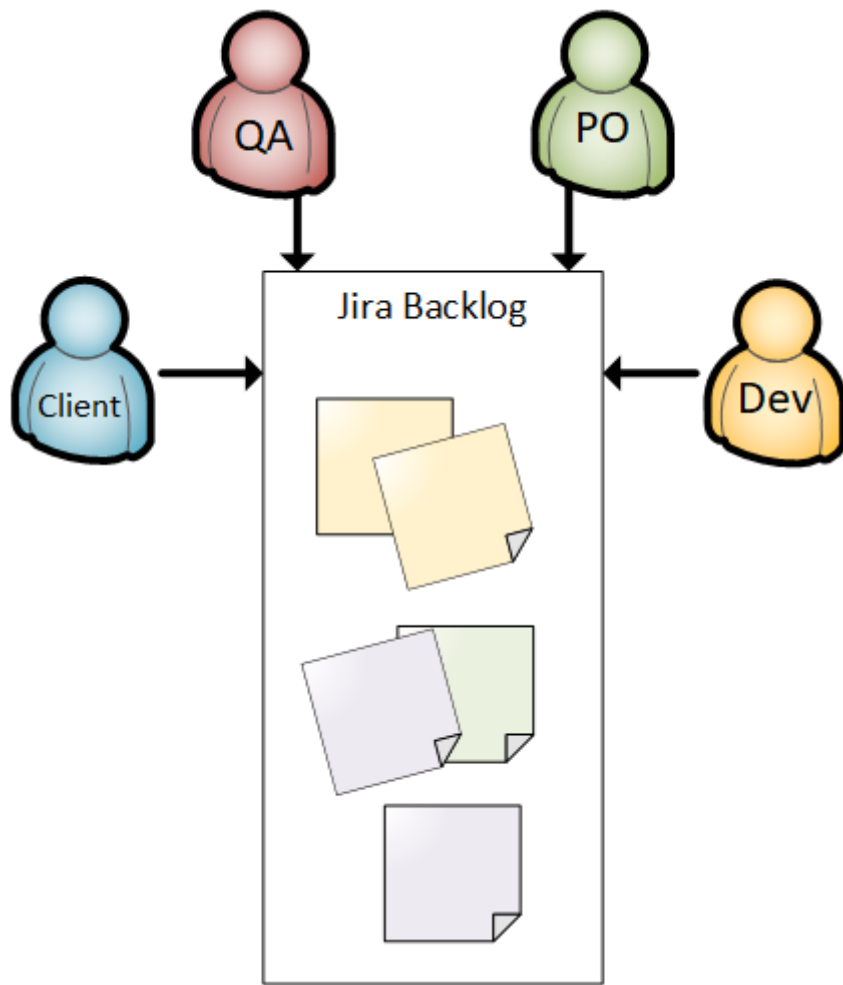
We run a flexible system where anyone in the project can add an issue to the backlog in Jira.

The backlog will contain a mix of technical tasks, bugs, user stories and spikes .

**see below*

Larger pieces of work, that might span several sprints, are entered as Epics. Other issues can be assigned to an epic to show that they will contribute to it.

Just because an issue is in the backlog, does not necessarily mean that it will be worked on.



Refining the backlog

The development team must have a complete understanding of each ticket before they can agree to work on it. Each ticket must be understood at a business level and at a technical level.

Where the definition of done is not well understood, or there are edge cases that aren't specified, the ticket will be referred to stakeholders for clarification.

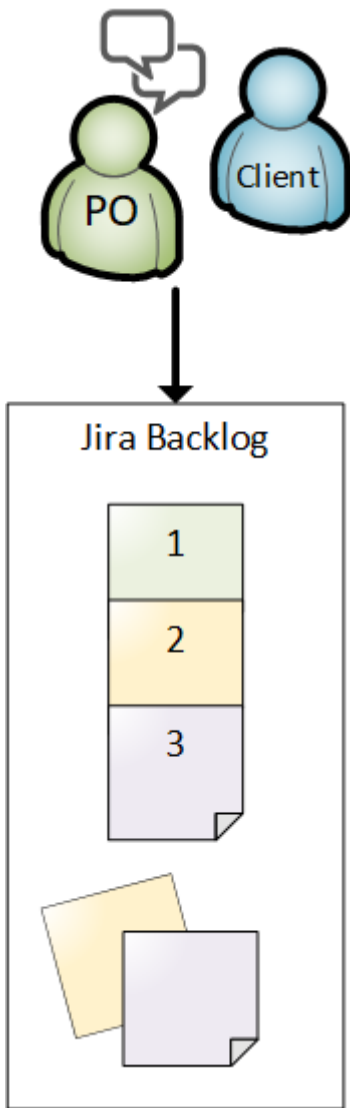
Where the ticket is not understood technically, we will raise a time-boxed spike to investigate.

*

For these reasons we prefer to plan our sprints well in advance, so that all preliminary work can be undertaken first.

Once a ticket is well understood the development team will collaboratively 'score' the ticket, reaching a multi-faceted consensus on the amount of work, the complexity of the task and their experience with tasks of that nature.

Our Product Owner and the Stakeholders will collaborate to prioritise the backlog, with stories that are 'Ready to Play' at the top.



Sprint Planning

The top tickets in the backlog are moved into the current sprint until it contains tickets with a combined score equal to the team's velocity (based on past experience).

It is worth noting that two, equally performant teams might allocate different scores to the same ticket, if they were both presented with it, and will have different sprint velocities accordingly. Our scoring system is deliberately unitless.

During a Sprint

The functional test team will create a test plan from the User Stories in the backlog.

Development

Developers will write code to fulfil the selected user stories, technical tasks and so on. Our code is supported by the automated tests we write, to prove our code works first in isolation and then when integrated with the rest of the solution.

In parallel to this, our automated testers will produce the scripts to validate the work produced at a UI level.

Review

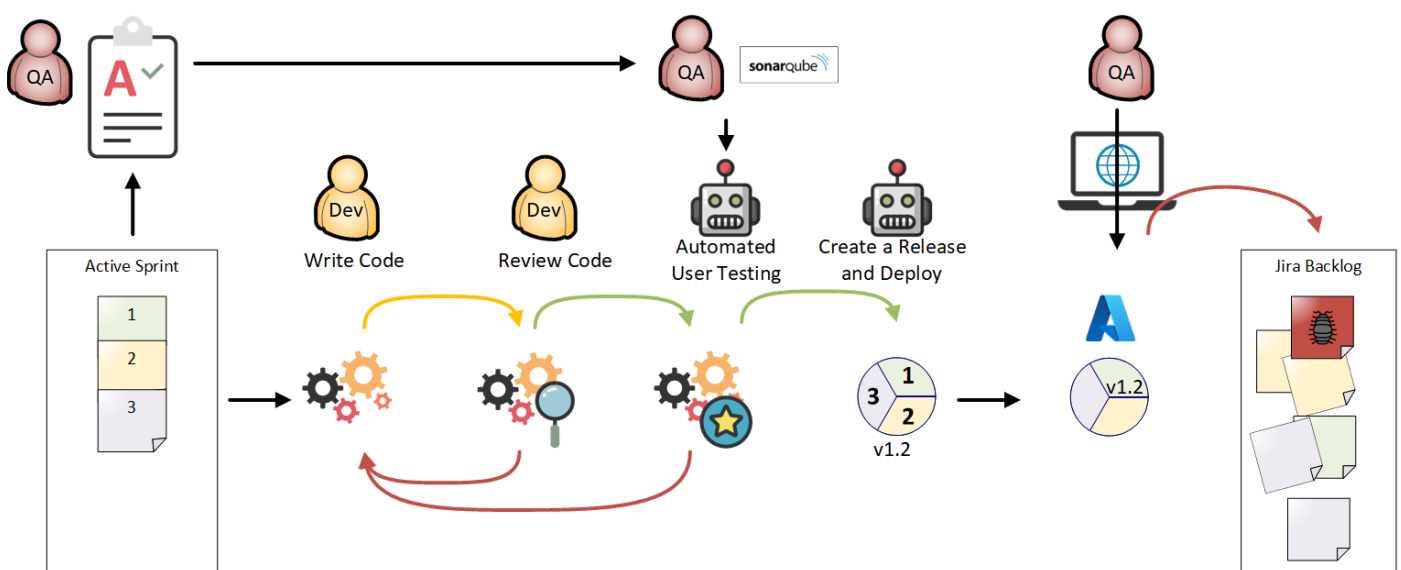
Once a ticket has been completed, we automatically run every unit and integration test for entire project, to make sure the changes we've made in one area of the project don't adversely affect another.

The code is then subject to the review of at least two other development team members. We verify that the work completed is of good quality and that it will fulfil the definition of ready, as specified by the ticket in Jira.

Internal Signoff

Our code is then deployed to a working environment where we can then run every UI test for the entire project.

If QA are happy with the result they will sign off the ticket. If bugs are found after this point, they must be added as new tickets on the backlog and go through the usual triaging process with the product owners and stakeholders.



Release management

As the development process continues, we accrue new completed work in our development environment, for internal sign off, that is not in our staging area, for external sign off.

External Signoff

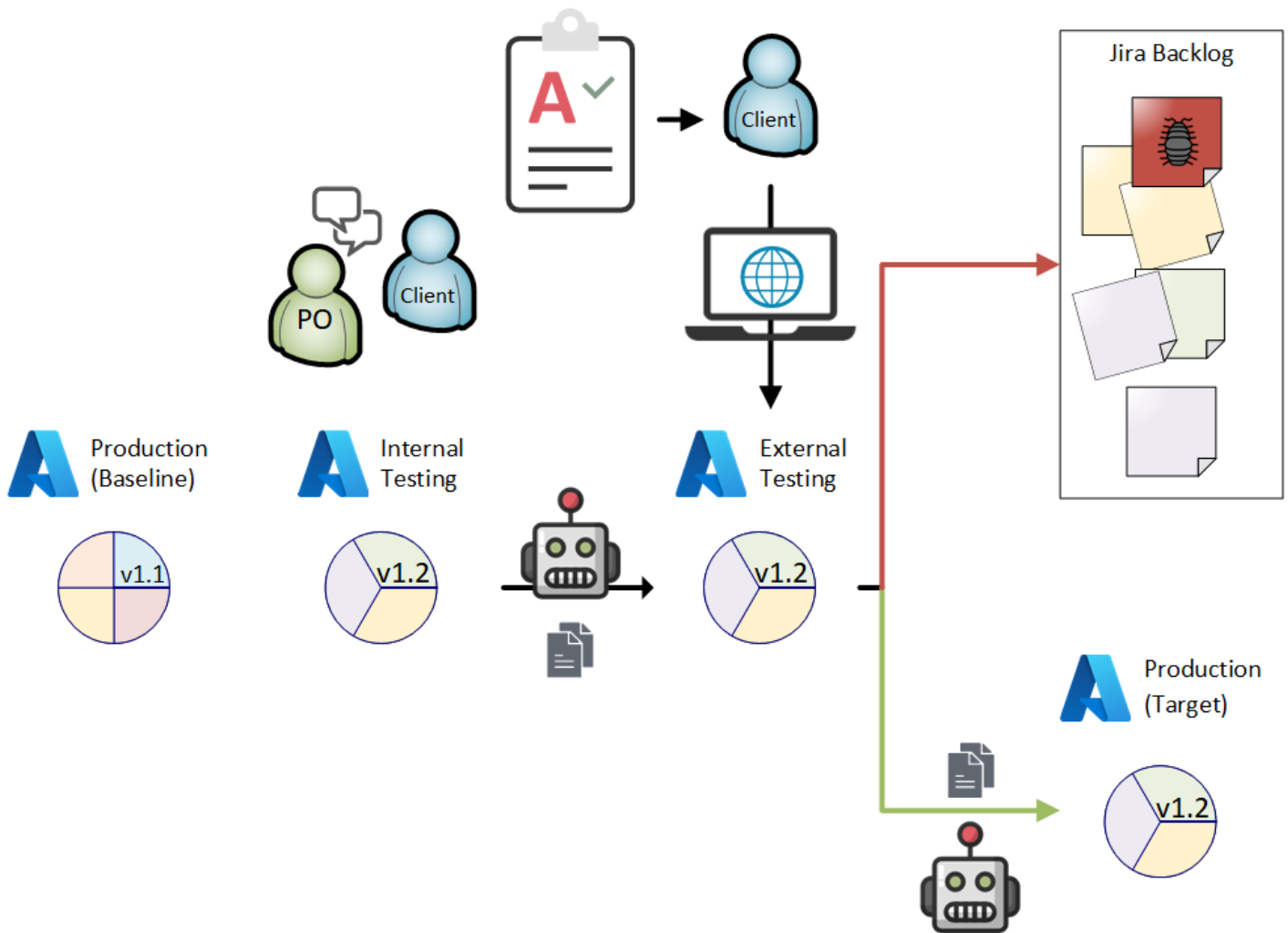
Stakeholders can communicate that they would like to see a new version of the product at any time.

The work that is signed off internally is bundled into a release build and migrated the external sign off environment. We will provide our own functional test plan to the stakeholders, as a good starting point, but the stakeholders are then free to test the system as they see fit.

Going live

If the stakeholders are happy with the product then the version that is currently in our external testing environment is promoted to our live environment.

This process can happen at any time, although it is advisable to do this earlier in the week in case the need to rollback our work arises.



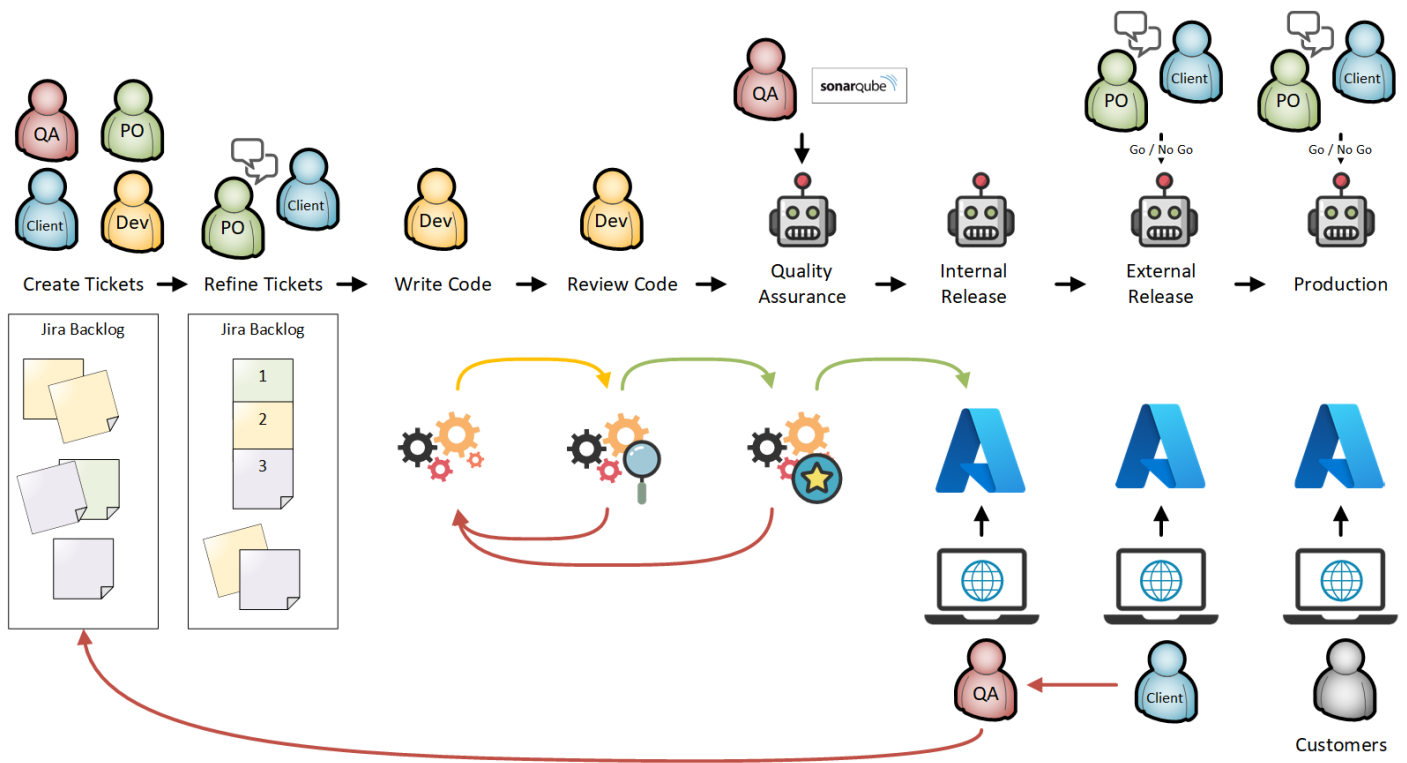
Q: Where are the Gant charts?

Our development, testing and infrastructure team practice continuous deployment.

We can deliver code to our external testing and production environments rapidly, and on demand.
We can also roll back changes just as quickly.

Our formal meetings to review each sprint's work may be a good time for stakeholders to request a new version of the product, although this could also happen at any other time.

The backlog in Jira delivers transparency over our work and allows for dynamic re-prioritisation of that work by the stakeholders.



Research & Design: Discovery Plan

Scoping, Resourcing & Planning Discovery

Pre-Discovery Planning

Everything that needs to be in done by R&D before Discovery Starts

- Identify key Stakeholders
- Identify Key Services Involved.
- Plan & Schedule Primary Research
- Kick-off brief session with Core Discovery team
- Plan Stakeholder / Service Workshops
- Define Discovery brief & Manage expectations

Discovery Team

Core Discovery

- Product / Project Manager
- User Researcher
- Service Designer

Supportive Discovery

- Technical Architect
- Data Analyst
- UX Designer

Discovery Phase

We can't make an informed decision

- Primary Research
- Current State Blueprint
- Expert Review
- Competitor Analysis
- Existing Systems & Processes

- Existing Analytics
- Identify best practice guidelines & principles for the service
- Workshops with the Services, Contact Centre
- Technical Discovery

Discovery Conclusion

We can make an informed decision

Key Owner: **Architect, UX, UR & SD**

- Current State Blueprints
- Primary Research
- Existing Architecture
- Technical Discovery
- Expert Review
- Competitor analysis
- Existing Analytics

Key Owner: **Product Manager**

- Defined problem statement
- Hypothesis
- Key unknowns & Assumptions
- Refined Value Proposition
- High level outcome roadmap
- Experiment backlog
- Indicative size of opportunity