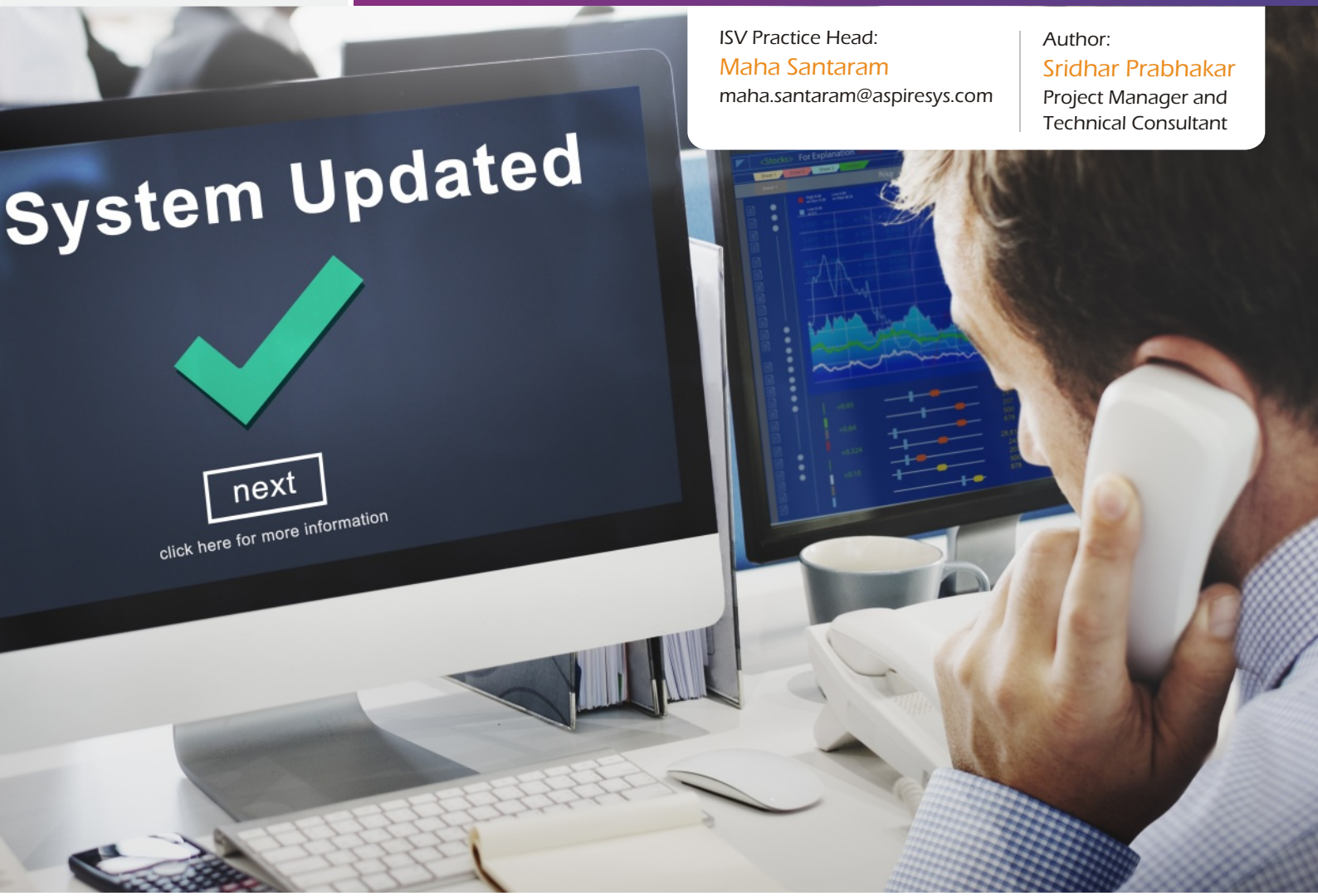
Effective Release Planning and Tracking through JIRA

ISV Practice Head:
Maha Santaram
maha.santaram@aspiresys.com

Author:
Sridhar Prabhakar
Project Manager and
Technical Consultant



Every software product goes through several upgrades and releases during its active lifecycle. With the adoption of rapid development methodologies and other tools, the release process has become crucial in the product's life span. Successful upgrades and deployment depends on release planning and it involves the following steps.

- Identify the features / requirements for the release.
- Estimate and plan for changes or development
- Arrive at a goal, plan and a release date.
- Communicate the plan to all stakeholders and track the release.
- Triage and handle failures in the execution.

With modern software development methodologies, features are implemented in a much shortened duration than the traditional models which in turn demands very rapid deployment and upgrade cycles. A lot of developers have adapted to this mode and have been delivering software products consistently. Yet, in spite of all the practice and knowledge, meeting a release deadline is almost always a challenge for any team of any size with every possible skill required. Why do teams dread release deadlines? Why do engineers work overnight? Why is effort overrun a common occurrence in a lot of projects?

In this blog, we make an effort to answer some of these questions and propose a solution (at least partially) to ease some of the pain points in the release process. The pain points that most software developers face are listed below.

- Mapping and tracking requirements to specific releases.
- Estimating the tasks and the completeness of the estimate
- Tracking effort overruns and estimating for it.
- Know the remaining work against the release goal at any point of time
- When to ramp up and ramp down the team
- With most of the modern softwares being designed with multiple inter-locked modules, upgrades happen module-wise and managing the compatibility becomes difficult.
- Backward compatibility releases.
- Improving the overall process with knowledge based on history,

Jira is a tool that attempts to solve some of these problems. This article intends to explore possibilities in JIRA that can help with this case.

REQUIREMENTS PLANNING AND ESTIMATION

Over the years, the way requirements are gathered, accumulated and planned for delivery has changed drastically. Incidentally, requirements change way too often in today's business environment. Adding to the business environment, the needs and thoughts of requirement providers have also become extremely agile. In effect, there is a need for a very quick turn around-time for requirements, before it gets changed. This brings us to the question of an ideal sprint duration. Theoretically, there is no right answer and every answer is correct as long as it works in a given case. The most common sprint duration is 2 weeks. But there are success stories with a bigger duration too.

Irrespective of the duration, the planning should include time to allow integration testing and possible retests.

RELEASE MANAGEMENT

Below is a generalized quick list of steps for effective release management. These steps are guiding steps that abstract the exact process and can be customized for every project depending on the working style and the needs of the project.

- Understand current release procedures
- Establish a release plan and establish goals
- Effective inspection of the release and adapt changes to bring the release on track
- Create an implementation plan for process and infrastructure required for release
- Test and deploy the implementation plan
- Automate as much as you can
- Revise processes and improve speed and cost effectiveness

Effective Release Planning and Tracking through JIRA

JIRA – INTRODUCTION

This section gives a quick intro to JIRA for newbies. For those who know and understand JIRA well enough, the below section can be useless and the below section can be skipped.

JIRA, is a product of a company named Atlassian. JIRA works in tandem with some of Atlassian's other products to provide a seamless project management and tracking tool set. Instead of rambling on about what JIRA is and the usefulness of it, let's quickly define the various products that JIRA has and move on to how it can be used beneficially for the problem of release management.

- Jira Core -Project and task management solution built for business teams
- Jira - Combines development tools with agile features
- Jira Service Desk - A service desk service for your team's requests
- Confluence - Documentation service that helps tight binding with other Atlassian products.
- Hipchat - A team communication service
- BitBucket - A source code repository that supports GIT and mercurial repositories.

Jira is built specifically to handle an agile workflow. But, it can still be mended to handle other development lifecycle models.

Release Planning on JIRA

Assuming that a project is created on JIRA and there is a backlog of tickets that need to be done for the product, a version (later converted as a release) can be created within JIRA. Confluence, can also be used to create a grand plan and can then be imported into JIRA.

BOARDS, EPICS, VERSIONS, SPRINTS AND TICKETS

Board

A board is a conceptual equivalent of a project. The board is a workbench for the plan and the entire gamut of changes required in the product. Again, the exact usage of the board is determined by individual projects. Some create a new board for every release while some use the same board for all releases. But, core purpose of the board remains the same. Everything in JIRA begins with creating a board that holds all the tickets (issues, bugs, enhancements and to-do items). The entire planning works on the board. All tickets, versions, and sprints are created within a board.

Epic

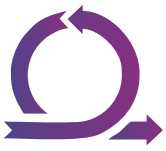
As defined on the documentation, "An epic captures a large body of work". This is again a convenience feature that allows grouping of a large feature or a planned set of work. Tickets can be grouped under an epic and eventually tracked separately. This allows a feature wise view of the progress.

Effective Release Planning and Tracking through JIRA

Version

A version as the name suggests, is a number that relates to a planned release number. It can be used either way. The version is what every ticket is tied to in order to facilitate tracking. JIRA provided version level tracking features to help plan ahead. To create a new version (assuming that the user has the permissions to do this), follow the steps in the below link https://confluence.atlassian.com/agile063/jira-agile-user-s-guide/planning-a-version/creating-a-version

A version and an Epic can be used interchangeably because both are in effect a conceptual group of tickets. But, ideologically, epic drives feature wise tracking while versions drive release wise tracking.



Sprint

A version need not be a completed in one go. It can be split into multiple sprints or smaller releases to make things easier and to set intermediate milestones.

Tickets

Tickets are the basic low-level requirements for the product. The ticket can be used to do the following

- Create a basic requirement
- Provide a medium to add description and clarifications for questions on the ticket.
- Addition of estimates and tracking work log
- Attachments such as screenshots, documents etc to support the ticket

Workflows

JIRA provides a way to customize the way tickets transition through the development stages. These are called workflows and they can be customized within JIRA to adapt to the needs of the project/product. The below link should provide a fair description of the workflows and the ways to customize it.

https://confluence.atlassian.com/adminjiraserver072/working-with-workflows-828787890.html

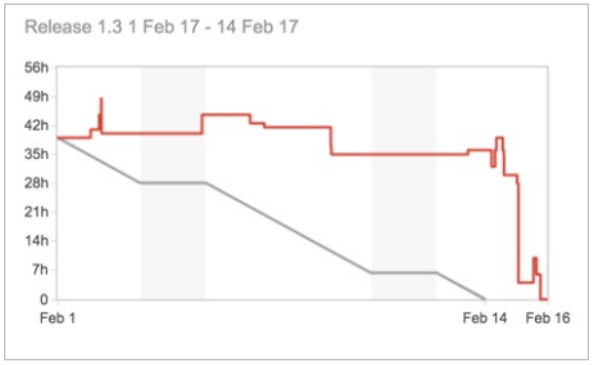
Planning and tracking the release

- Release planning begins with creating a board and creating a backlog of tickets and items. This should create a basic planning board to start with.
- The next step is to add high level estimates to the tickets so that a plan can be arrived at.
- Based on how far ahead the product is envisioned, versions and epics can be created. The estimated tickets can then be prioritized and moved under versions and epics.
- Given the size of the epic or version (in terms of effort), the number of sprints to be created can be arrived at. It would be good to plan for additional stabilization sprints in between to manage contingencies.

- Ideally, research and the difficult tasks need to be started in the earlier sprints so that any fallout from these can taken during the later sprints.
- It is typical practice to discuss, clarify and revise estimates of each ticket before a sprint begins. This gives the developers an insight into the requirement and a chance to revise the effort based on clarifications.
- During development, logging work in terms of effort is a must.
- As development progresses with each of the ticket, the completion status of the version/release keeps itself updated.
- It is important to log estimates, effort and other clarifications within this ticket to be able to refer back to it later.

Release Tracking

JIRA provides a feature (When actual effort is logged into JIRA) where the effort breakdown is provided in the form of a burndown chart. With every effort log, the system calculates the effort remaining in the sprint / release based on the estimates and the chart is updated accordingly. This burndown chart is provided based on the timeline set for the sprint or release. This gives a realistic estimate of the amount of work to be done in the remaining time versus the amount of work done since the start.



The burndown chart gives a general trend of the progress and can be very helpful in making decisions by simply extrapolating. In the sample image below the grey line indicates the planned burndown while the red line is the actual burndown.

- As seen in the graph above, sometimes, developers have a tendency to keep tickets with them and push all of them together to the testers. This leads to a sudden surge of work to the testers towards the end of the sprint. Developers should close tickets periodically and engage testers throughout the cycle.
- It is recommended that developers close tickets and then move on to the next ticket instead of working on multiple tickets parallely.
- Estimate overrun in every sprint or release needs to be analyzed and the root cause identified so that the next set of estimates are better.

Sprint Burndown and Release Burndown

JIRA can help visualize effort burndowns for not only a sprint but also a complete release. Both these have their unique advantages. The rollup of the burndown up to the release gives a strong signal to the product owners on the possibility of achieving the release goals. Sprint burndowns help developers track the current sprint more closely. The difference is just the granularity of the data.

Effective Release Planning and Tracking through JIRA

Practical Challenges

Changing Scope – Agile is many a time interpreted as the flexibility to change the scope at any point in time. There should be a defined period minimally for a sprint for which the scope remains frozen.

Accuracy of Estimates – Estimates should be done in collaboration with other teams so that dependencies and impacts are considered and tackled appropriately. The accuracy will increase sprint on sprint based on the learning curve.

Team Solving Culture – The culture of openness to raise impediments on the work and facilitation to solve the impediments on a daily basis should be established.

Reduce Tech Debt and Rework – Dedicating a part of the team (especially the senior folks) to work one sprint ahead on the impact analysis and technical design will reduce tech debts and also increase the productivity of the team.

Effort Overrun – You should expect effort overrun as the plan is made on an agile requirement in order to control the scope and ramp up and down the team size accordingly

Effective Usage of the Tool – In many cases the tool features are not effectively used due to change management issues. Every planning, tracking, communication and collaboration on a release should be done in the tool.

CONCLUSION

JIRA provides a very customizable way to manage releases that can be adapted to every product / project. JIRA is not free. But it justifies the cost by helping with seamless release management and a better project management process. It fits into the larger agile methodology for software development teams to embrace Continuous Integration and Continuous Delivery.

ABOUT ASPIRE

Aspire Systems is a global technology services firm serving as a trusted technology partner for our customers. We work with some of the world's most innovative enterprises and independent software vendors, helping them leverage technology and outsourcing in our specific areas of expertise. Our core philosophy of "Attention. Always." communicates our belief in lavishing care and attention on our customer and employees.

SINGAPORE +65 3163 3050 | NORTH AMERICA +1 630 368 0970 | EUROPE +44 203 170 6115 | INDIA +91 44 6740 4000 | MIDDLE EAST +971 50 658 8831

For more info contact info@aspiresys.com or visit www.aspiresys.com