An eBook for Jira Admins

revyz

The Ultimate Guide to: Change Management

The last best practices for Change Management in Jira Cloud and Data Centre

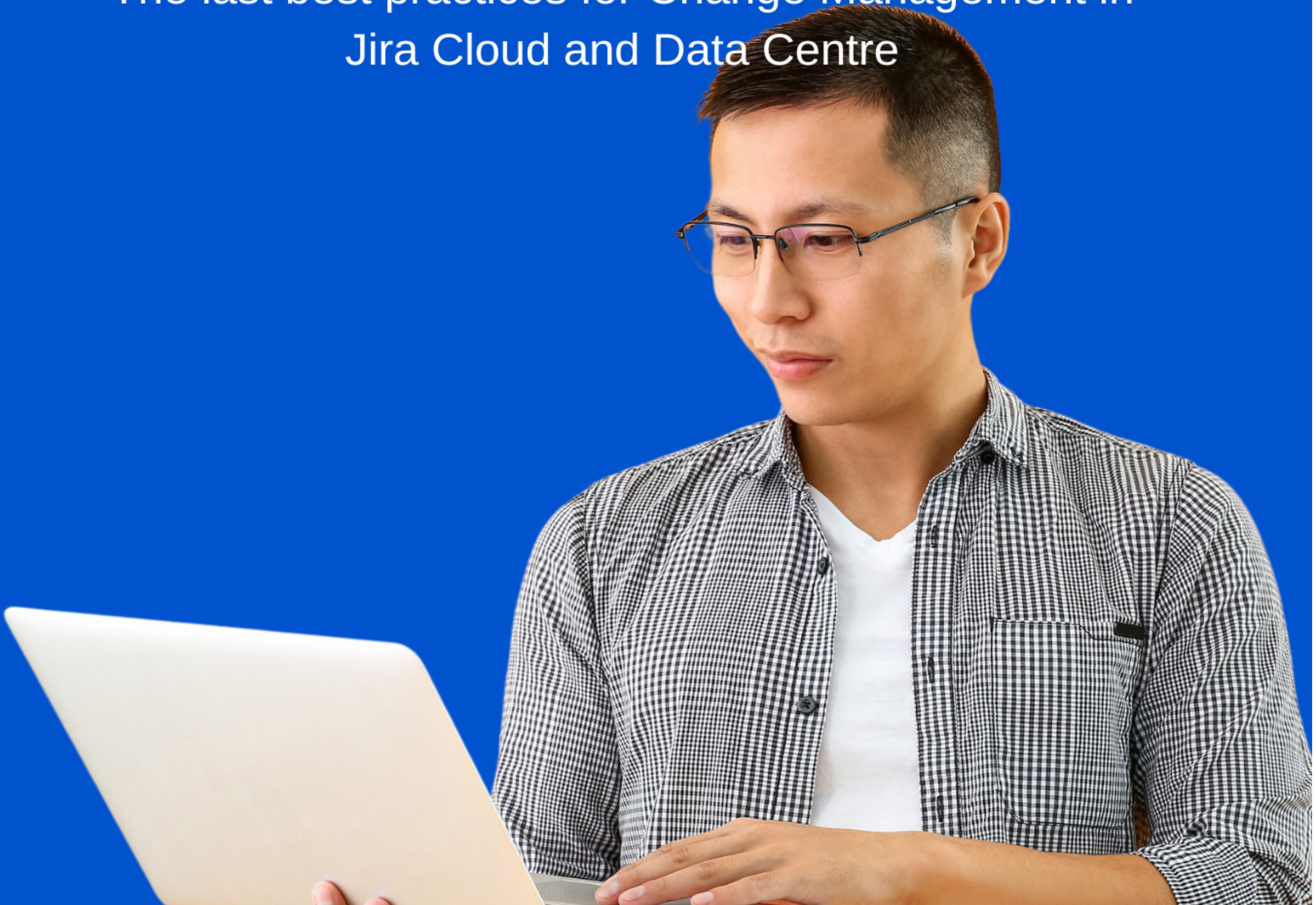


Table of Contents

About This eBook

Staying up-to-date with the latest techniques and technologies for managing Jira helps you manage it more effectively and efficiently. Particularly true when it comes to the big changes on the horizon with Jira Server End of Life.

This eBook is designed to assist you in understanding the differences in managing system changes in your Jira environment when you have shifted from Jira Data Center or Server, over to Jira Cloud.

In this eBook we will cover concepts related to change management, ITIL framework and risk management and apply these topics to the Jira Cloud environment.

By leveraging the knowledge shared in this eBook, you'll be equipped with the tools and insights needed to optimize Jira's capabilities, adapt to the future of a cloud environment, and successfully manage projects in alignment with your organization's requirements.

> “Jira provides many tools to support change management, but too many organizations are unaware of how they can be used.”
>
> Christian Lipski, Solutions Architect
> *Praecipio.com - Platinum Atlassian Solution Partner*

About The Authors

Vish Reddy
CEO & Co-founder Revyz.io
vish.reddy@revyz.io

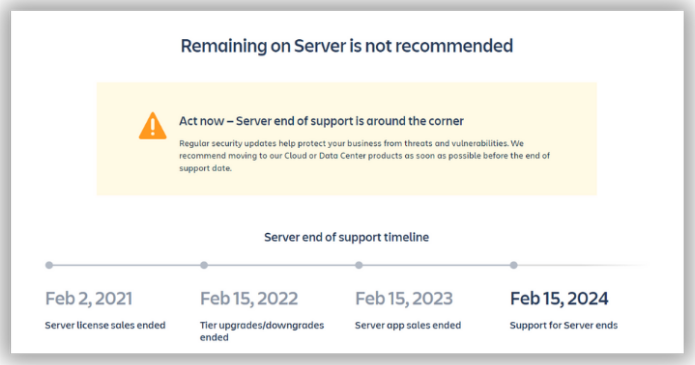
With an extensive background in large enterprise data security and data backup firms including Cisco, McAfee and Symantec, Vish has set up, managed and advised on software change management in a broad spectrum of environments. Vish is the Co-founder and CEO at Revyz Inc.

Stu Lees
VP Marketing Revyz.io
stu.lees@revyz.io

Stu has a diverse career spanning two decades involved in IT. A qualified Information Systems Auditor, System Security Officer in the banking sector and CEO of multiple tech firms, Stu now heads up marketing and partnerships at Revyz Inc.

Is Cloud Really The Future?

You only need to pop onto the Atlassian website to see how serious they are taking their strategy to move customers like you onto their cloud offering. With the end of life for Jira Server rapidly approaching, Jira Administrators around the world are having to tackle the problem of making sure that their current practices, processes and policies are thoroughly covered after your shift to Cloud.



Remaining on Server is not recommended

Act now – Server end of support is around the corner

Regular security updates help protect your business from threats and vulnerabilities. We recommend moving to our Cloud or Data Center products as soon as possible before the end of support date.

Server end of support timeline

Feb 2, 2021	Feb 15, 2022	Feb 15, 2023	Feb 15, 2024
Server license sales ended	Tier upgrades/downgrades ended	Server app sales ended	Support for Server ends

The 2020 decision by Atlassian to shift focus towards cloud-based solutions, was driven by the benefits to both the customer and the provider that the cloud provides including; operating cost savings, flexibility, scalability and simplified support.

So with the need to transition from Server, keep this timeline in mind, here is a quick summary of the key milestones:

- Server products will be supported until 2 February 2024.
- Atlassian and Marketplace Partners will no longer offer technical support, security updates, or bug fixes for critical vulnerabilities after 15 February 2024.
- You'll need to consider a transition to Atlassian Cloud or Data Center without compromising security or functionality within this timeline.
- The average time it takes to complete a migration from Server to Cloud is between 3-9 months[1].

What about Data Center?

Data Center maintains its importance for many organizations on their journey to the cloud. At the time of writing this eBook, we are not aware of any firm plans by Atlassian to end of life the Data Center product, instead, they're enhancing collaboration and insights for Data Center subscribers by including apps like Automation for Jira, Advanced Roadmaps, Team Calendars for Confluence, and more.

Priority support is also provided for Jira Software, Jira Service Management, Confluence, Crowd, and Bamboo.

For all of your existing apps in your Jira Server, you won't need to renew your subscriptions and you can unlock new functionality by upgrading to a compatible Data Center version or installing the app at no extra cost.

Atlassian sees this as a way to offer flexibility for organizations based on their specific requirements and migration timelines.

- If you're on Server, the writing is on the wall. It's time to upgrade to either Jira Cloud or Data Center regardless.
- If you're currently using Data Center, you're already falling behind the competition who have embraced the benefits of the Cloud and opted to adjust to the early niggles Cloud brings.

[1]See references on our blog

The Differences Between Server, Data Center and Cloud

We all know how vital Jira is for organizing and tracking tasks, bugs, and issues. But before we dive into the depths of this eBook, let's quickly fact-check the differences between traditional Jira (Jira Data Center and Jira Server) and Jira Cloud.

	Jira Server	**Jira Data Center**	**Jira Cloud**
Environment	Self-hosted deployment option, installed and managed on local servers and infrastructure.	Self-hosted deployment option for enterprise-grade teams designed to provide high availability, performance, and scalability.	Fully managed, cloud-based deployment option provided by Atlassian. Jira is accessed via a web browser without the need to manage your own servers or infrastructure.
Suitable for	Small to medium-sized teams or organizations with specific infrastructure requirements, customization needs, or limitations.	Large organizations or teams with high user demand, distributed teams, or requirements for uninterrupted access to Jira.	All sizes of teams and organizations. Regardless of infrastructure, customization or demand requirements.
Licensing	A one-time software license purchase with an annual maintenance fee for support and upgrades.	Requires a Data Center subscription, which offers additional benefits and support compared to Jira Server.	Cloud-based subscription plan, typically billed on a monthly or annual basis. The plan covers usage, hosting, and support.
Performance	Under the control of the end customer, performance depends on the hardware environment the software is hosted upon.	Supports clustering and load balancing, distributing the workload across multiple nodes for improved performance and redundancy. Provides features like active-active clustering and shared file systems to ensure high availability and fault tolerance.	Atlassian manages the infrastructure and ensures high availability and performance even during peak usage.
Scalability	Limited due to server capacity and the infrastructure it's installed on. It may require additional effort and downtime to scale resources as the user base or workload grows.	Allows organizations to scale horizontally by adding more nodes to handle increased user load.	Allows you to easily scale resources up or down based on your needs.
Updates and Maintenance	Control over updates and upgrades. You can choose when to apply them.	Control over updates and upgrades. You can choose when to apply them.	Updates and maintenance tasks are handled by Atlassian. Including updates, patches, bug fixes, and the introduction of new features seamlessly without manual handling.

Customization	Extensive customization options, including direct access to server infrastructure, allowing for deep-level customizations tailored to specific organizational needs. But, it requires more administrative effort and expertise to manage and maintain. Updates and upgrades may require additional testing and validation to ensure compatibility with customizations.	Similar customization capabilities as Jira DC, including the ability to make deep-level customizations and direct server access. But, it requires you to manage and maintain their own servers and infrastructure, which can be resource-intensive. Updates and upgrades need to be applied manually, potentially resulting in downtime during the update process.	Supports customization options, including custom workflows, fields, and screens. While there may be certain limitations compared to Jira Data Center, Jira Cloud still offers flexibility for adapting Jira to your specific needs.
Add-ons and integrations	Supports extensive add-ons and integrations from the Atlassian Marketplace, enabling you to customize and enhance Jira's functionality, integrate with external systems, and streamline workflows at scale.	Provides a rich ecosystem of add-ons and integrations from the Atlassian Marketplace, allowing for the extension of Jira's capabilities, integrate with various tools and systems, and tailor your instance requirements.	Provides access to the Atlassian Marketplace, where users can find numerous add-ons and integrations to extend the functionality of Jira.
Collaboration	Collaborate effectively with team members using the communication features and shared project spaces.	Enable seamless collaboration across your teams and locations with the scalable and high-availability architecture.	Foster real-time collaboration and enhanced teamwork with accessible and collaborative features
Licensing	One off Purchase with Annual Maintenance	One off Purchase with Annual Maintenance	Monthly or Annual subscription
Logical Security	Atlassian responsible for software patching and customer responsible for server, network intrusion security.	Atlassian responsible for software patching and customer responsible for server, network intrusion security.	Shared responsibility model - see resources here
Data Security	Customer responsible for all data backups and disaster recovery	Customer responsible for all data backups and disaster recovery	Shared responsibility model - see resources here
User Access Security	Customer responsible for managing all user access policies and user administration	Customer responsible for managing all user access policies and user administration	

Like any transition, there are pros and cons between each system.

- **Jira Server** is established and you've probably even become accustomed to some of its niggles, but its support is fading.
- **Jira Data Center** is a powerhouse but it's lacking the "wow factor" businesses currently need to gain a competitive edge in the market.
- **Jira Cloud** is in its infancy and has a few different ways of working you'll need to embrace. But, not only are they worth it, it's also where Atlassian is investing the most with their cloud-first strategy. Plus it has unbelievable power for scale and storage for you to capitalize on and achieve your business goals.

There's no avoiding the power of the cloud. After all, businesses globally are already seeing the positive impact it's had on their collaboration and delivery. Sure, migration will require careful planning, readiness assessments, and thorough testing. But Atlassian has plenty of resources and guidance stocked up to safely catapult you to the cloud with them.

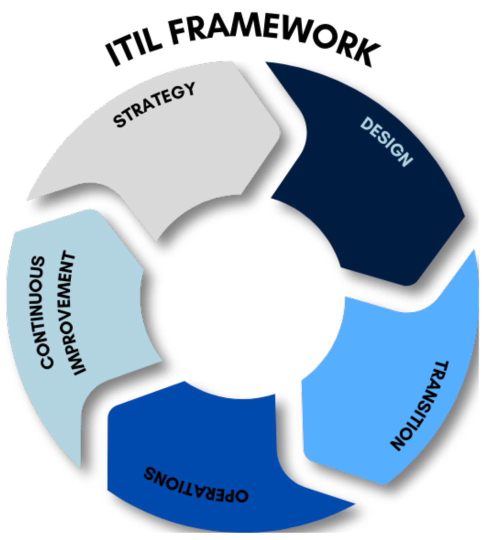
The Importance of Understanding the Shared Responsibility Model

As countless software vendors have migrated their applications from on-premise, to a cloud version, the concept of "shared responsibility" has emerged. This concept comes about when software vendors and their customers have to evolve the assumptions of who is responsible for what, after the customer migrates to a cloud version of software. There are many articles and resources on Shared Responsibility throughout the web and for Jira customers, it is crucial for you to understand how Atlassian view shared responsibility.

ITIL Framework and Change Management in Jira

The Information Technology Infrastructure Library (ITIL)[2] framework stands as a bedrock of modern IT service management (ITSM). Originally developed by the UK government's Central Computer and Telecommunications Agency in the 1980s, ITIL has evolved into a comprehensive set of best practices, guidelines, and processes that have been adopted by organizations worldwide.

The framework offers a systematic approach to managing and optimizing IT services, aligning technology operations with business objectives.



ITIL comprises a collection of interconnected processes and practices that cover the entire service lifecycle. From service strategy and design to transition, operation, and continuous improvement, each stage is meticulously defined within the framework. ITIL emphasizes the significance of a customer-centric approach, focusing on delivering value and quality while maintaining a flexible and adaptable environment.

Jira is integral to both internal and external IT processes for most organizations who use it, so it makes sense for us to give mention to ITIL when we talk about best practices for change management in Jira.

Why the ITIL Framework is Important to Consider for Software Change Management[3]:

Software change management plays a pivotal role in ensuring the seamless deployment of updates, patches, and enhancements. The ITIL framework's relevance becomes evident as organizations strive to strike a balance between innovation and stability. Here's why ITIL should be a cornerstone of your software change management strategy:

Structured Change Control: ITIL provides a structured and standardized approach to change management. By defining clear processes for requesting, evaluating, approving,

[2] See references on our blog
[3] See references on our blog

and implementing changes, the framework minimizes the risk of disruptions caused by poorly managed alterations to software systems.

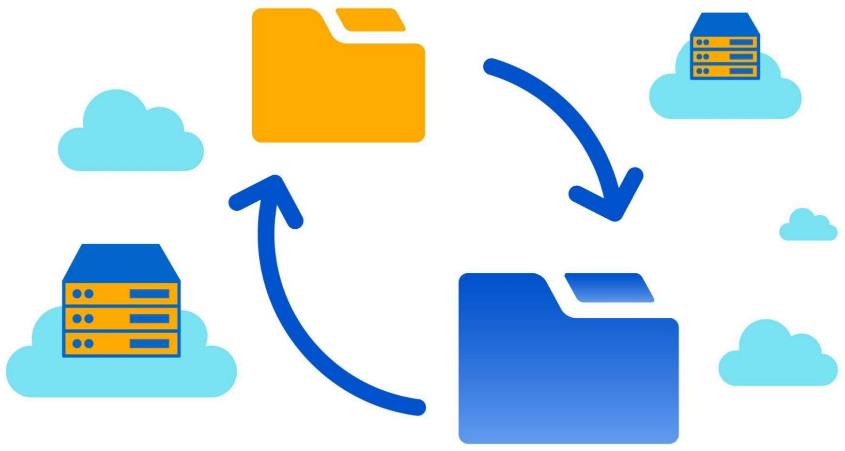
Risk Mitigation: Change management within ITIL promotes rigorous risk assessment and mitigation strategies. It helps IT professionals and senior executives identify potential issues and plan contingencies before implementing changes. This proactive approach safeguards critical systems and ensures business continuity.

Change Evaluation: ITIL emphasizes the importance of evaluating the impact and effectiveness of changes post-implementation. This evaluation allows organizations to learn from past experiences, fine-tune their change management processes, and continuously improve their software deployment practices.

Auditable Compliance: For organizations subject to regulatory compliance, ITIL offers a trail of documented processes and decisions, aiding in audits and demonstrating adherence to best practices.

Continuous Improvement: ITIL's emphasis on the "Continual Service Improvement" lifecycle stage encourages organizations to learn from each change and apply these insights to enhance future change management endeavors. This iterative approach drives efficiency and effectiveness over time.

> This section has been co-authored by Marco who is an IT Certified consultant at Platinum Atlassian Solution Partner,
>
> YOURPIC.
>
> *Https:yoururl.com*



Managing Changes In Traditional Jira vs Jira in the Cloud

Regardless of your environment choice, you'll need to lean into established and well-defined change workflows, approvals, and documentation to ensure stability and minimize risks. In Jira Cloud, you'll also need to embrace slightly new ways of doing things, as some of the traditional Jira functionality you're used to may no longer exist. We'll detail a few of the big ones shortly.

Additionally, regardless of the deployment type, proactive change management is still a key player regardless of your environment choice. So, understanding the nuances of each will enable successful change management in Jira, promoting efficient and controlled project progression.

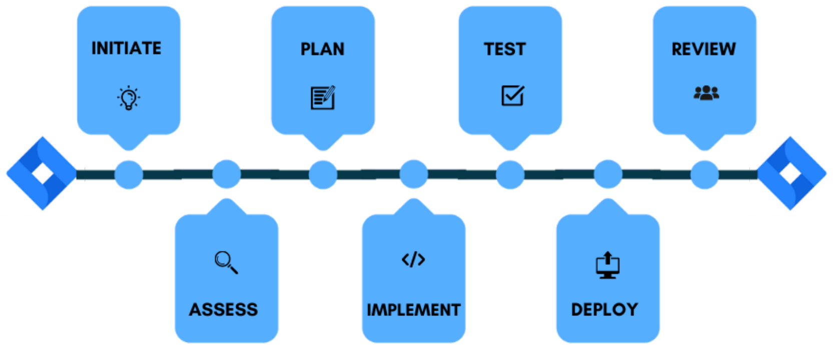
In this section we'll first outline the best practices for change management.   We will then compare the changes between Server / Data Center and Cloud given the inevitability that whether you're an existing Server and DC user, you'll need to start comparing DC to Cloud.

Best Practices in Change Management In Jira Cloud

Before we tackle the change management differences between the Jira environments, let's revisit best practices.

The standard system development lifecycle (SDLC) best practices[4] used in software development still apply to managing changes in Jira. SDLC frameworks come in large variety of framework but we have chosen to utilize the ITIL framework as it is widely adopted in many organizations and has been thoroughly proven over several decades of use. This framework is summarized as follows;;

1. Initiation: This is the first step, where the change is proposed and documented in a Request for Change (RFC). The RFC should include information about the change, such as the purpose, scope, impact, and risks.
2. Assessment: The change is then assessed by the Change Manager and other stakeholders to determine its impact and risks. The Change Advisory Board (CAB) may also be involved in this step.
3. Planning: Once the change has been approved, a plan is created for its implementation. This plan should include details of the steps involved, the resources required, and the contingency plans in case of problems.
4. Implementation: The change is then implemented according to the plan.
5. Testing: The change is tested to ensure that it is working as intended.
6. Deployment: The change is deployed to production.
7. Review: The change is reviewed to ensure that it has been successful. This may involve collecting feedback from users and stakeholders.



[4] See references on our blog

The next section goes over in detail the processes that Jira Admins should be aware of and incorporating into your Jira environment, no matter which infrastructure it is hosted upon.

Change Management Planning Checklist

Before we get into the actual steps you would follow, let's review the questions you would ask yourself before making any changes:

- [ ] What needs to be implemented
- [ ] Why was a change made and by whom and when
- [ ] What will be the impact of the change made in the environments where the changes are made
- [ ] How do we make changes without impacting our production systems
- [ ] How do we promote changes reliably from a testing environment to production
- [ ] What is our roll-back strategy
- [ ] How do we maintain a paper trail for compliance

Best Practices for Change Management

Previewing Changes and Collaboration

To avoid surprises and mishaps, it's still wise for you to put on your Administrator detective hat before making configuration changes. Previewing means you can take a peek at how your proposed changes might impact existing customizations and mitigate them ahead of deployment. Equally, collaborating with other Administrators and users helps foster collective code ownership and ensures changes are thoroughly reviewed and discussed before implementation.

Managing the Software Development Lifecycle

Implementing an effective change management process involves moving planned configuration changes through a structured software development lifecycle. This cycle typically includes stages such as plan, develop, test, deploy and maintain. By following this repeatable process, you can ensure that changes are properly tested and validated, minimizing the risk of impacting production environments.

Test in a Sandbox and Staging Environment

Testing in a sandbox environment is crucial in that it allows you to test changes before committing them and catch any potential issues or conflicts early on. Additionally, setting up a staging environment that closely mirrors your production setup lets you validate major changes and , ensuring they work well and have the desired impact before rolling them out.

Versioning and Rollback

You can still maintain your cosmic undo button with versioning and rollbacks in Jira Cloud. With this capability, you can easily navigate the path of configuration modifications, understanding what was implemented and confidently rolling back to a previous state if necessary.

Auditing and Compliance

Having the capability to save post-configuration changes is an invaluable tool for creating a comprehensive audit trail. It provides a record of what changes were made, who made them, and when, which is indispensable for meeting compliance requirements and conducting thorough analysis if issues arise. This approach empowers you to maintain accountability and confidently navigate complex compliance and governance stand-offs. It’s also one of the reasons we’re proud to have built Revyz to address this.

Updates and Patches

When you are running on premise platforms (such as Jira Server and Jira Data Center) then change management best practices should include the installation, testing and introduction of these patches to production. In general terms, vendor supplied changes should be treated in the same way as your own custom changes.

As a cloud-based service, Jira Cloud is managed and maintained by Atlassian, there are not patches or updates to apply.

Monitor System Activity

By regularly reviewing logs, you can identify any suspicious or unauthorized actions. These could be indicative of security breaches or errors resulting from changes. Monitoring system activity provides valuable insights into the overall health and security of your Jira Cloud instance, allowing you to take proactive measures to address any potential risks or issues promptly. Bake logging and monitoring system activity into your Jira Cloud security as standard.

Regularly Back Up Data

By performing regular backups, you can restore your system to a previous state if any critical data is lost or if unexpected problems occur. To ensure consistency and efficiency, it's advisable to automate the backup process. In doing so, you can minimize the risk of human error, maintain a reliable backup schedule, and have peace of mind knowing that your Jira Cloud data is securely backed up and can be easily restored when needed.

Limit Admin Access

Only grant administrative privileges to people who genuinely require them for their roles or responsibilities. This practice minimizes the risk of accidental or intentional changes made by unauthorized individuals. By limiting access, you maintain tighter control over changes in your Jira Cloud environment. Ultimately, this reduces the likelihood of unauthorized modifications that could compromise system integrity or data security. Implementing strong access controls ensures that only trusted personnel with appropriate permissions can make changes within the admin interface.

Educate and Train Users

Provide solid training and documentation to your users. Help them understand the best practices for change management within Jira Cloud. Encourage them to report any concerns or issues they encounter along the way and create a culture of awareness and responsibility among your users.

Staying Safe From Errors When Managing Changes in Jira

In this section, we will go through the variety of migration approaches in cloud and non cloud Jira environments to help Jira Admins more fully understand how to act after you receive a configuration change request from a stakeholder to add a new workflow, add new custom fields, or change screens in Jira server / data center and cloud.

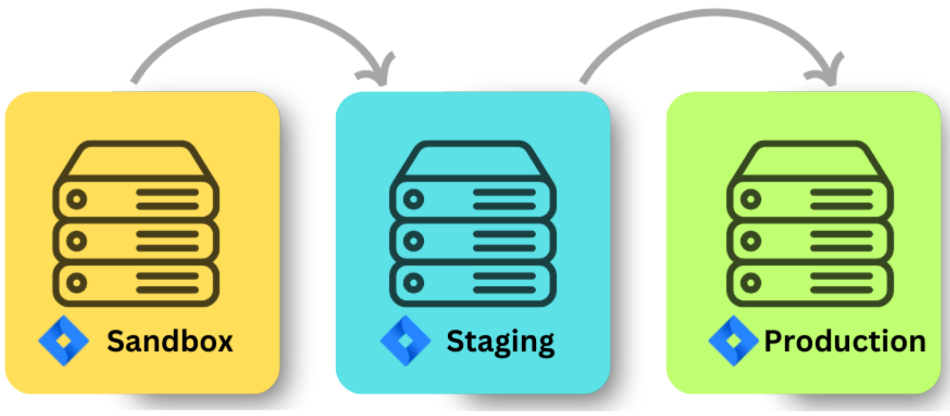
In the next few pages, we will cover off the following scenarios;

1. Change Management in Jira Server / Data Center
2. Change Management in the Cloud - Manual technique
3. Change Management in the Cloud - with a Utility app

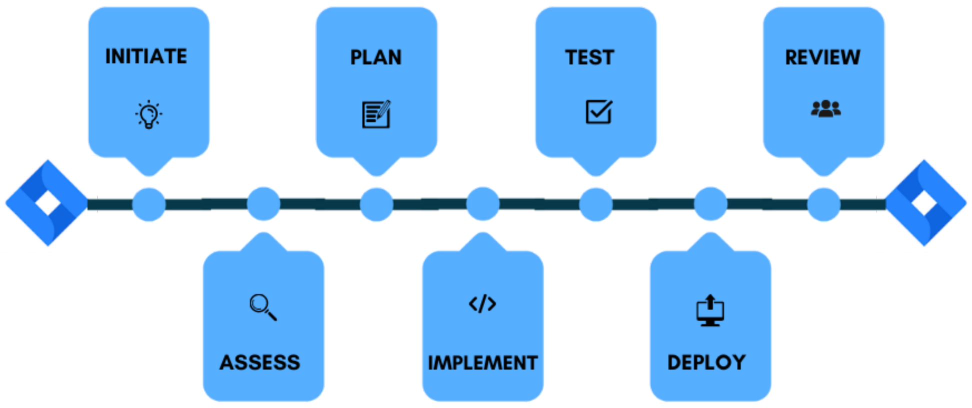
> I've really only seen a handful of organizations that perform change management for Jira according to a defined process. The vast majority, are making configuration changes on the fly. It's not just that they are too busy to follow a process, it's that they aren't aware of the methods to do change management right.
>
> *Kyle Moseley - Principle*
> *Blue Ridge Consultants (blueridgeconsultants.cx)*

Change Management in Jira Server / Data Center

A typical Jira server / data center environment has at least three setups to develop and manage changes in the Jira application and the app "Configuration Manager for Jira" - Data Center / Server version installed in all three environments and to get started it is critical to have the configuration on the Staging and Production setups to be identical[5].



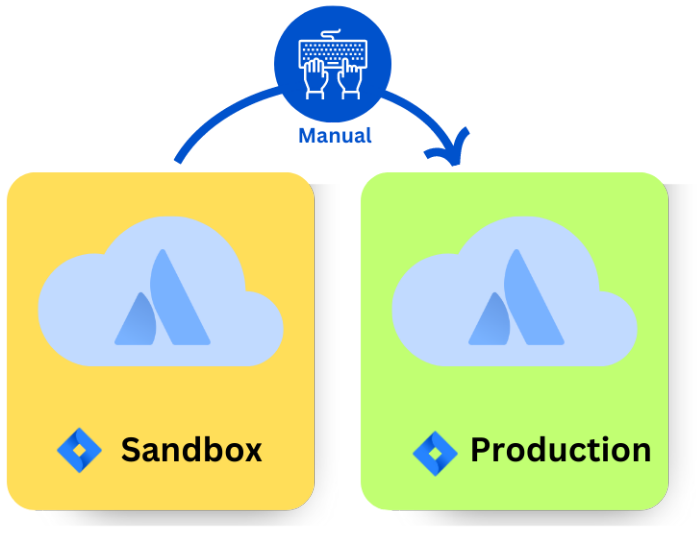
Here are the steps you would follow at a high level to address the change request you would get from a stakeholder.



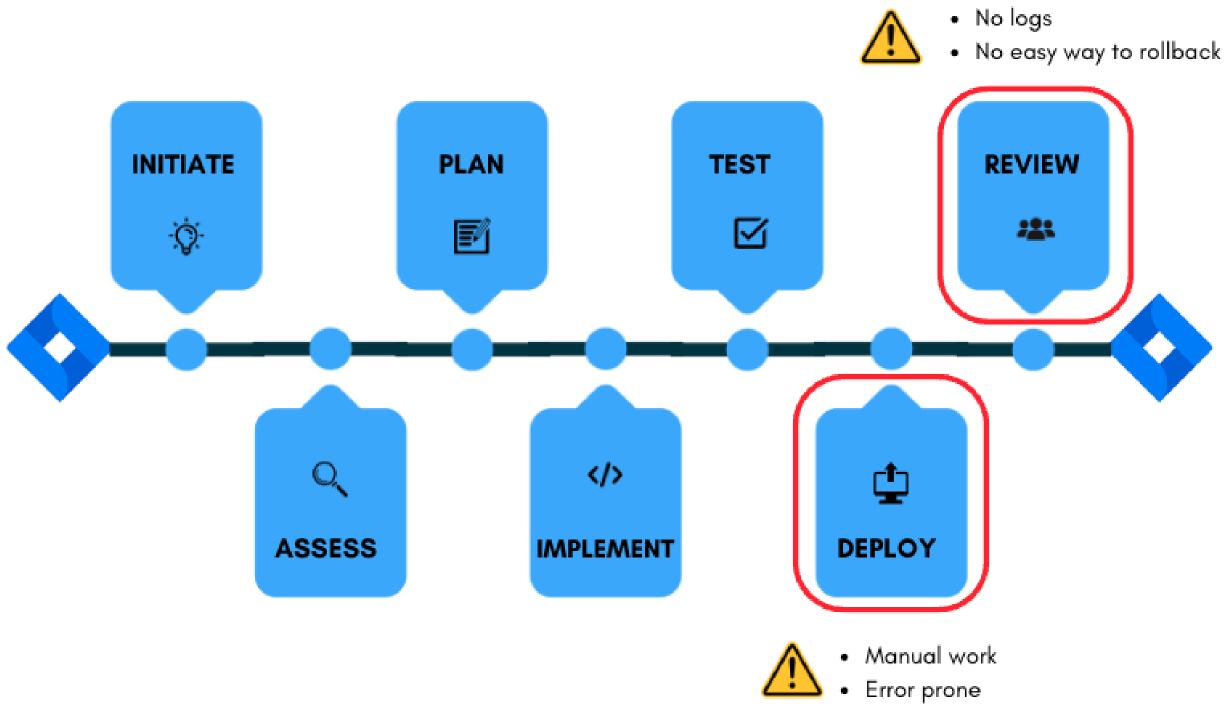
[5] See references on our blog

Initiation	**1**	Document the changes proposed / received from stakeholders in a Request for Change (RFC). The RFC should include information about the change, such as the purpose, scope, impact, and risks.
Assessment	**2**	Implement the changes requested in your Development setup to determine possibilities, impact and risks. Develop a detailed change document including rollback plans.
Planning	**3**	Get plan approved by CAB and secure times for when the changes will be implemented in Sandbox, Staging & Production
Implementation	**4**	Implement changes in Sandbox setup and test
	5	Merge changes from Sandbox to Staging setup using a tool such as Jira Configuration Manager (Data Center / Server version)
Testing	**6**	Test the changes made and perform user acceptance testing and get sign-off from stakeholders
Deployment	**7**	Take a backup of the Production setup - a critical step in case you need to roll back
	8	Merge changes made in Staging into Production using a tool such as Jira Configuration Manager (Data Center / Server version)
Review	**9**	Validate changes in production and get sign-off
	10	Download the audit logs and store them away for future reference
	11	In case there is a need for a roll-back, one could go back to the backups taken or use the Configuration Manager for Jira app to roll back the configuration in your production environment

Change Management in Jira Cloud - Manual Technique



Below are the steps you would follow at a high level to address the change request you would get to make in the Jira Cloud environment. We have highlighted the areas where there are differences in the image below when you compare making changes to configuration in a Data Center environment and Cloud.



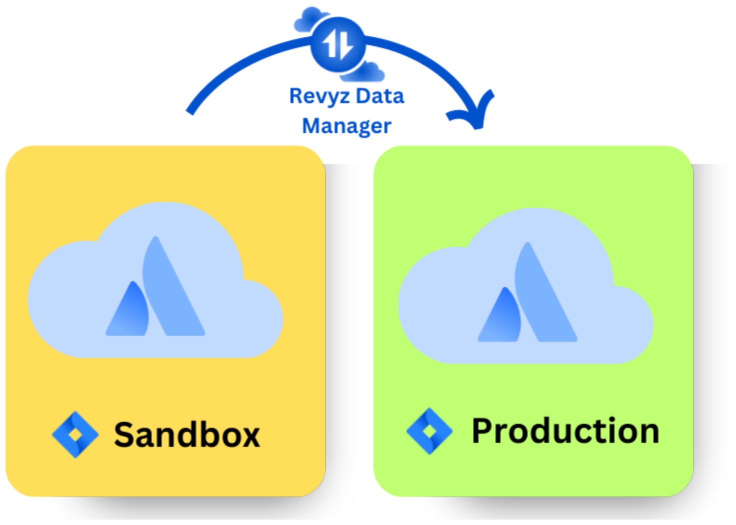
Insert image

Assuming you have a Jira Premium / Enterprise license which includes a Sandbox site or if you have a Jira Standard license then the choices you have are to either create a separate site for test purposes or use the existing site, if using the existing site - tread carefully, create separate projects and have the configuration to be unique to that project i.e. not have any shared objects as much as possible so as to not affect projects being used in production.

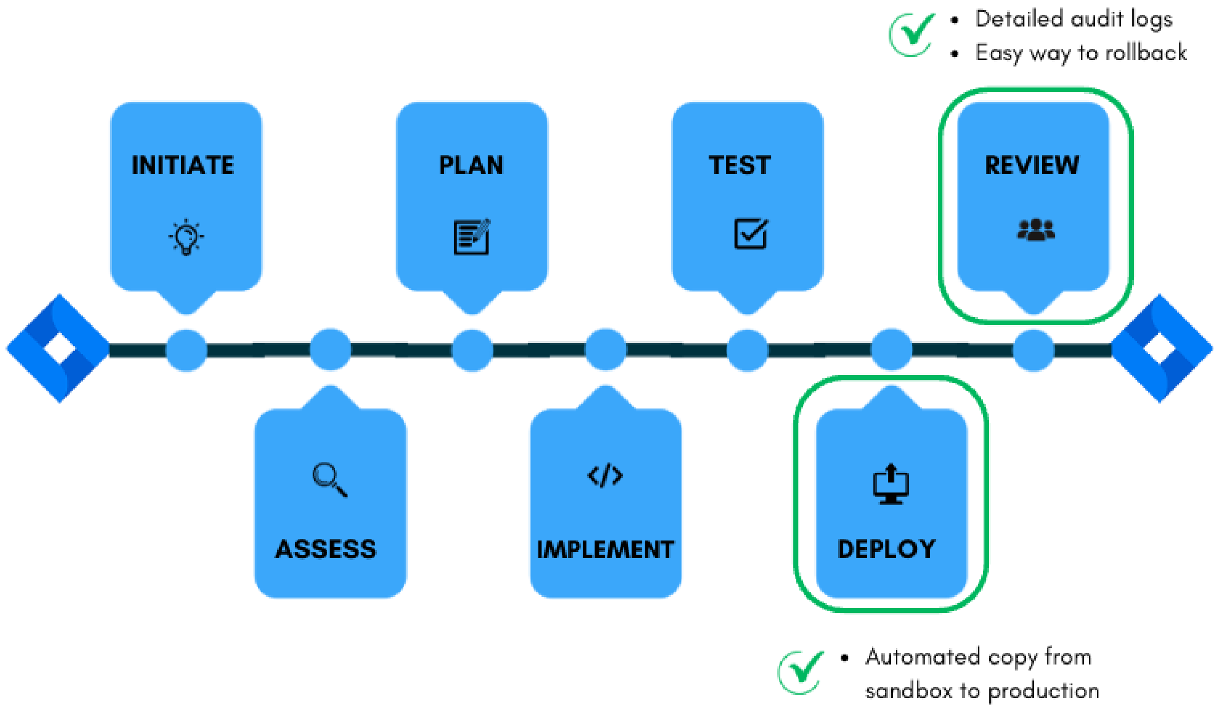
The steps followed are largely inline with the steps followed in the Server / DC case, the key differences are that you don't have a staging environment and the changes made are all manual vs. using a tool. ***Manual changes are error prone and lack audit logs leading to compliance and governance issues and of course not minimize the time spent in doing manual work across multiple setups.***

Initiation	**1**	Document the changes proposed / received from stakeholders in a Request for Change (RFC). The RFC should include information about the change, such as the purpose, scope, impact, and risks.
Assessment	**2**	Implement the changes requested in your Development setup to determine possibilities, impact and risks. Develop a detailed change document including rollback plans.
Planning	**3**	Get plan approved by CAB and secure times for when the changes will be implemented in Sandbox & Production
ImplemenInsert tation	**4**	Implement changes in Sandbox setup
Testing	**5**	Test the changes made and perform user acceptance testing and get sign-off from stakeholders
Deployment	**6**	Take a backup of the Production setup - a critical step in case you need to roll back
	7	Recreate changes made in the Sandbox in Production manually
Review	**8**	Validate changes in production and get sign-off
	9	In case there is a need for a roll-back, one could go back to the backups taken

Change Management in Jira Cloud with a Utility App (Revyz Data Manager)



Below are the steps you would follow at a high level to address the change request you would get to make in the Jira Cloud environment along with the Revyz Data Manager for Jira app at a high level very similar to the steps articulated in the previous sections but with the Revyz Data Manager, you reduce the risk of errors and the amount of time spent and have a detailed log of the changes implemented. Thus helping with reducing the possibility of downtime while adhering to the compliance and security related best practices.



Insert Image here

Initiation	**1**	Document the changes proposed / received from stakeholders in a Request for Change (RFC). The RFC should include information about the change, such as the purpose, scope, impact, and risks.
Assessment	**2**	Implement the changes requested in your Development setup to determine possibilities, impact and risks. Develop a detailed change document including rollback plans.
Planning	**3**	Get plan approved by CAB and secure times for when the changes will be implemented in Sandbox & Production
Implementation	**4**	Implement changes in Sandbox setup
Testing	**5**	Test the changes made and perform user acceptance testing and get sign-off from stakeholders
Deployment	**6**	Take a backup of the Production setup - a critical step in case you need to roll back
	7	Connect the Sandbox site with your Production site using the Revyz Data Manager app and use the ***configuration clone*** feature in the

		app to select the specific configuration objects that need to be copied over to the production setup
	8	Review, analyze and clone the configuration objects that need to be copied over from the sandbox to the production setup’s
Review	9	Validate changes in production and get sign-off
	10	In case there is a need for a roll-back, one could go back to the backups taken or use the Revyz Data Manager for Jira app to roll back the configuration in your production environment

Summary of Change Management Differences Between Platforms

Migration Step	DC & Server	Jira Cloud Manually	Jira Cloud with Utility
Ability to export objects to local computer and	✔	✘ Manual copy	✔
Ability to copy configuration objects from sandbox to production	✔	✘ Manual Recreate	✔
Ability to review dependencies of migration objects in sandbox prior to migration	✔	✘	✔
Ability to take a backup of production prior to making a configuration change	✔	✔	✔
Ability to take another backup of production after migration	✔	✘ Will overwrite pre-production backup	✔
Audit trail of all migration activities for corporate compliance and also to enable to clear and smooth roll-back should errors be detected	✔	✘ Need to be manually documented	✔
If post migration error occurs, ability to review product object involved and its related objects to make debug and rework faster	✘	✘	✔
Ability to restore individual objects and not have to restore entire site in the event of a migration related issue	✔	✘	✔

Revyz Data Manager - Designed to Drastically Reduce Time and Risk Jira Configuration Management

By reading this eBook, we hope that you will now understand the options, the inherent risks and the inevitable gaps that the Jira Cloud platform presents admins when it comes to change management within Jira.

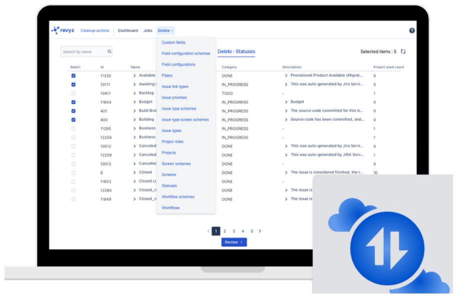
As an award-winning Atlassian Marketplace partner with an established data backup and recovery tool, we were asked by our clients to see if we could help streamline and derisk configuration change management.

To do this, we added new capabilities within the Revyz Data Manager for Jira app to help Jira administrators in addressing the following use cases:

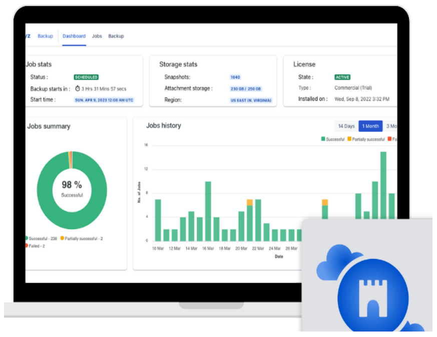
- Copy / Clone specific configuration objects i.e. Workflows, Status, Screens, Issue Types etc.. in granular manner from a sandbox Jira site to your production Jira site and for that matter to any site
- Understand the impact of cloning configuration objects before they are cloned by way of looking at the differences
- Address security and compliance needs of logging all changes that are being made

The Revyz Data Manager for Jira does not require you to code or script, all the changes can be implemented via the UI provided in the app.

- Easy to learn and use with no-code and no scripting
- Migrate data and config from one site to another
- Selectively migrate data and/or config from one site to another
- All migration activities fully logged for audit purposes
- Natively integrated into Jira
- Fully back up your production site prior to migration (see Data Backup tool below)


Cloud to Cloud Clone Tool is part of the award-winning Revyz Data Manager App available on the marketplace

The Revyz Data Manager app also has the most comprehensive data backup and granular restore tool available on the Marketplace. Features include;



Data Backup and Restore is part of the award winning Revyz Data Manager App available on the marketplace

- Scheduled full and/or partial data backups
- Granular restore of any part of your backed up data
- Alerts and notifications integrated into Jira
- All backup and restore activities fully logged for audit purposes
- Easy use with no-code and no scripting
- Natively integrated into Jira

References and Resources

This eBook was researched in detail. You can find a full list of the reference sites on our blog page.

https://www.revyz.io/blog/15-essential-resources-to-understand-change-management-in-jira