omnissa™

Navigating the 8 stages of app lifecycle management

A guide for IT leaders

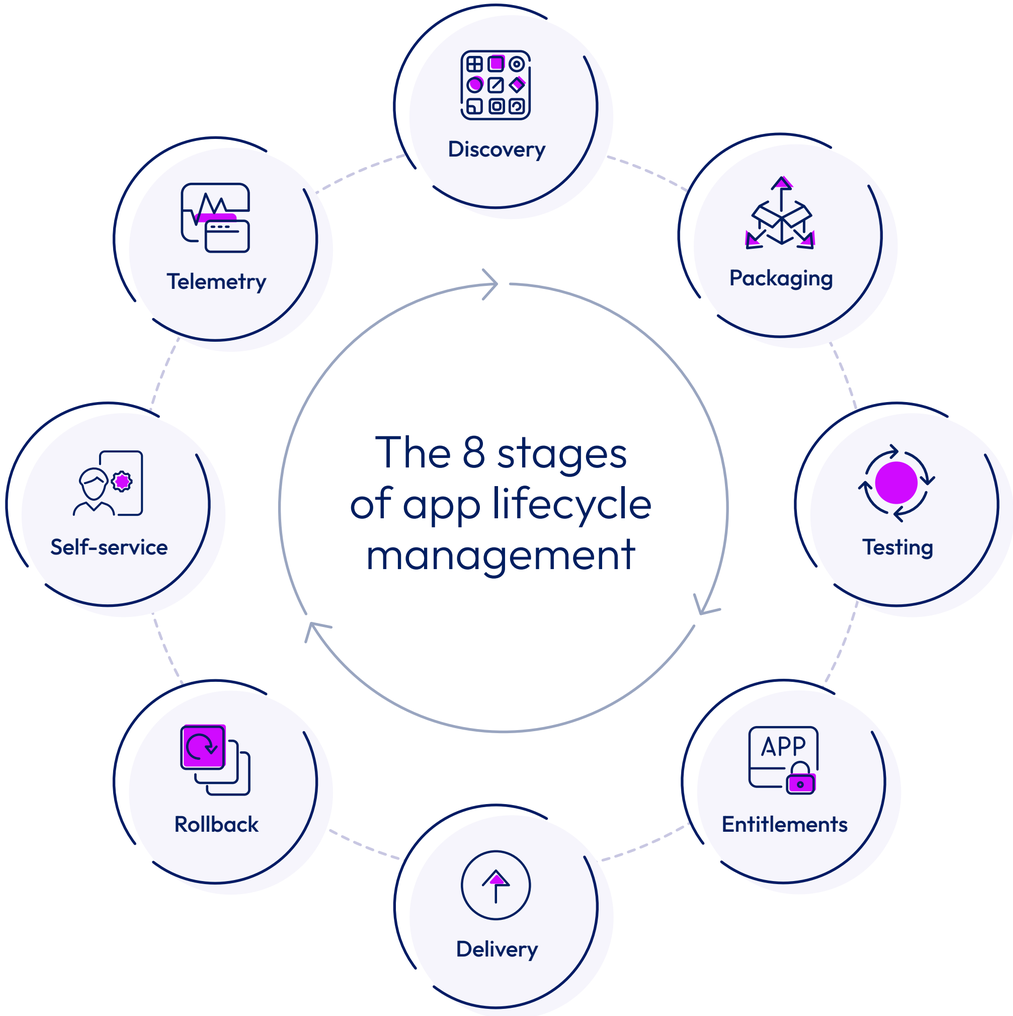
GET STARTED →

How to simplify app lifecycle management

Today's organizations face increasing pressure to deliver seamless, on-demand access to apps. However, the backend processes that support app delivery are often hindered by outdated practices, fragmented tools, and inefficiencies.

This e-book covers the eight stages of modern app lifecycle management, highlighting various ways to make the process more predictable. By examining app management pitfalls and considerations, it demonstrates how to transform your app delivery processes.

With a focus on automation, intelligent data management, and a user-centric approach, we'll explore strategies to help you streamline operations and deliver better app experiences to your end users.



omnissa

Stage 1: Discovery

Discovery is the foundational stage of app lifecycle management. It involves identifying all the apps running in your enterprise, understanding which ones your users rely on, and recognizing when new versions or critical updates—security patches, for example—are released.

Discovery can be both active and passive. It might come from vendor notifications, vulnerability alerts, or internal audits. Security teams, IT admins, and vendors all play a role in surfacing important app data. Tools such as Lakeside Systrack and **Omnissa Workspace ONE® Experience Management** (formerly DEEM) can automate and enhance this process by scanning systems and reporting app usage and updates across the environment.

Ultimately, discovery helps answer the key questions your organization needs to ask: What apps do we have? Who is using them? Are they secure? Are they still needed? This information is critical to forming a solid app management strategy.

Considerations for discovery

- Understand external and internal app landscapes.
- Use tools such as Omnissa Intelligence to gain insights into app usage within your organization.
- Collect information for app inventory, consumption, and effectiveness, which impacts planning and optimization efforts.

Stage 2: Packaging

Once apps are identified, they need to be prepared for distribution across locations, devices, and user groups. This is where app packaging comes in. Traditionally, this involved copying installation files or running complex scripts. Over time, tools like MSI installers, ThinApp, and App-V were developed to streamline the process.

Today, app packaging is much less complex than it used to be. Formats like VHD and MSIX App Attach offer modern capabilities, while products like **Omnissa App Volumes** simplify the process even further, turning traditional packaging into a quick capture operation. For example, App Volumes' GUI-driven and command-line agent-based packaging and automation via appcapture.exe can make packaging more efficient.

App admins or business owners typically take the lead in the packaging stage. They understand how the apps are expected to behave, and how they should be packaged for consistent deployment.

Considerations for packaging

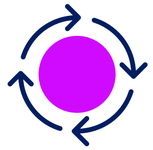
- Make packaging more efficient and reduce your dependency on packagers, especially when outsourcing.
- Promote open communication between app admins and packagers to avoid missing app info, wrong app bits, and incorrect settings.
- Smoke test your apps before reaching "super-user" or end-user testing.

Stage 3: Testing

Before an app can be released, it must be tested. This ensures the package functions correctly—not only from a technical perspective, but also in terms of user workflows and business processes.

Testing needs to account for different environments and use cases. For example, traditional MSI apps might require testing across multiple operating systems and with different combinations of installed software. **App Volumes** reduces some of this complexity by offering greater app compatibility and simplicity. For instance, the high compatibility rate of App Volumes reduces the burden of testing across OS versions.

It's crucial to involve your super users and business stakeholders in the app testing process. This helps ensure not just that the app runs, but that it can support the workflows it's intended for. This dual layer of testing—technical validation and user acceptance—is essential for a smooth rollout.



Considerations for testing

- Develop repeatable and automated processes around where and how testing is performed (e.g., single vs. multiple environments).
- Document the tools and test flows used to ensure consistent and reliable testing.
- Get rid of inconsistent or undefined testing methodologies, including manual "click around" tests and automated solutions like Access IT or custom scripts.
- Check regularly if your testing processes are yielding results, and improve them as needed.

Stage 4: Entitlements

An entitlement determines who can access an application. There are two levels of entitlement:

- **Optional access** – Users can request or install an app.
- **Mandatory access** – The app is assigned and automatically delivered.

This stage is critical in environments with cost controls or compliance requirements. For example, you might choose to require strict entitlement for apps like Microsoft Visio to avoid unnecessary license costs. And you might want to restrict access to certain apps to keep sensitive data secure. Thankfully, **App Volumes** makes it easy to manage entitlements for individual users or groups.

Entitlements also enable scalability. When integrated with a user management platform, they allow your organization to automate app distribution and ensure the right people always have access to the tools they need.

Considerations for entitlements

- Use entitlement policies and tools to provide accurate app entitlements to users or groups.
- Apply simplified app entitlement logic to help ensure expected results, especially when apps are pushed vs. pulled.
- Utilize app entitlements to boost security by reducing exposure to bad actors.

Stage 5: Delivery

Once apps are packaged and entitlements are set, the delivery phase begins. App delivery can happen in various ways: Pushing apps to physical desktops, deploying to virtual desktops or published app servers, or delivering apps via cloud-native services.

With innovative capabilities like Apps On Demand from **App Volumes**, delivery can now be dynamic. Instead of every user receiving every app, users get only what they need, when they need it. When a user needs an app, they simply double-click the app icon, at which point the app gets deployed on demand. This not only improves performance, but also reduces overhead on IT and infrastructure teams.

While delivery often involves infrastructure roles (OS and desktop admins), it's important to recognize that a great end-user experience is the ultimate desired outcome. Delivery must be reliable and seamless, regardless of the device and the user. For example, leveraging App Volumes for support of hybrid deployments and cross-platform portability can simplify app delivery across teams with a few clicks.

Considerations for delivery

- Check if the app delivery process is defined to improve the end-user experience.
- Ensure there are no missing dependencies (DLLs, .NET versions, drivers, etc.) during delivery.
- Optimize your legacy delivery models for modern app environments to avoid any unforeseen issues.

Stage 6: Rollback

Not every update goes as planned. App rollback provides a safety net when an update negatively impacts systems or users. The faster you can roll back apps, the more agile and resilient your app management and performance becomes.

App Volumes offers near-instant rollback capabilities, as simple as updating a marker—no reinstall needed. This stands in stark contrast to traditional rollback methods, which might involve powering down servers, reverting snapshots, or uninstalling apps manually.

A well-executed rollback strategy is like having mini-disaster recovery built into your app management framework. It reduces downtime and gives your organization the confidence to iterate quickly.

Considerations for rollback

- Develop a strategy for how to manage partial deployment failures (e.g., 800 successful installs, 200 failures). Do you roll back all 1,000 deployments and risk ending up in a broken or inconsistent state? Or do you roll back only the 200 failed ones, requiring a targeted, manual approach? Make it clear what the proper approach should be.
- Weed out any traditional deployment and rollback processes that haven't kept pace with modern app management.
- Outline simple yet effective steps for app rollback to help eliminate late-night calls and scale your app delivery.

Stage 7: Self-service

Self-service puts power into the hands of end users. Instead of calling the help desk to request software, users can simply go to a catalog, select what they need, and launch the app—instantly.

Omnissa Workspace ONE Intelligent Hub enables users to find, launch, or request apps in seconds. This greatly reduces IT workloads and increases productivity. It also enhances employee satisfaction by eliminating frustrating delays.

From an agility standpoint, self-service is a game-changer. It removes barriers between users and the tools they need to succeed.

Considerations for self-service

- Foster user independence with an app catalog that promotes self-service and visibility.
- Serve various types of apps (virtual, SaaS, native) with catalog features such as categorization for improved experience and consistency.
- Ensure better control of your IT environment by giving users access only to the apps they need.

Stage 8: Telemetry

Telemetry brings the lifecycle full circle. It provides real-time insight into how apps are performing, which ones are being used, and how users feel about their digital experience.

This data isn't just about system health—it's about user sentiment, experience quality, and app relevance. With tools like **Omnissa Intelligence** and **Workspace ONE Experience Management** that collect telemetry, your IT teams can assess user satisfaction, identify performance issues, and make informed decisions for future updates or retirements.

Telemetry feeds directly back into the discovery phase, helping teams decide what to optimize, keep, or decommission. It's the intelligence engine behind modern app lifecycle management.

Considerations for telemetry

- Integrate telemetry tools, app usage, and performance data into your entire app lifecycle management for better informed decisions and actionable insights.
- Complement raw telemetry data with user sentiment integration for a 360-degree view.
- Investigate the need for better, more targeted metrics to inform improvements and app strategy.

It’s time to transform app delivery and management

The complexities of modern app delivery are often rooted in legacy systems and outdated workflows, making it difficult for organizations to scale effectively. Various considerations identified throughout this e-book, ranging from packaging to entitlements to testing, highlight the need for change in the way apps are delivered and managed.

To succeed, your organization must move beyond traditional methods and embrace a more automated, data-driven, and agile approach. By doing so, your IT teams can reduce operational overhead while empowering employees with the tools they need to thrive in a dynamic, digital workplace.

omnissa™

To learn more about how **Omnissa** can help your organization improve app delivery and management, **contact an Omnissa associate** today.