Deployment Strategies

Abhyas Mall
CS @ UChicago

Recreate Deployment

How it works

- Terminate all old instances of version A. [1] [2]
- Then start new instances of version B. [1] [2]
- Because nothing serves traffic during the "gap," downtime is expected (unless you have an external workaround like maintenance mode). [3] [4]

Advantages

- Simple and straightforward deployment process. [4]
- Avoids version skew in production**.** No temporal inconsistency / interface mismatch caused by mixed versions, because you're not running A and B concurrently. [4] [C]
- Application state entirely renewed. [3]

Disadvantages

- Causes downtime as all existing pods are terminated before new ones are created. [4]
- Not suitable for applications requiring high availability. [4]
- User disruption due to service unavailability during the update. [4]

When to use it (use cases)

- Your application cannot run multiple versions simultaneously. [5]
- You have breaking changes that are incompatible with the old version. [5]
- You're deploying to development or test environments where downtime is acceptable. [5]
- Your application manages state that cannot be shared between versions. [5]

Rolling Update

How it works

- A new instance is created, and the new version (Service B) is installed. [C]
- The system starts sending requests to this new instance. [C]
- Then one old instance (Service A) is drained and removed. [C]
- This process is repeated until all old instances are replaced. [C]
- During this time, both old and new versions run together. [C]

Advantages

- Reduces downtime, maintaining service continuity. [4] [C]
- Lower cost than Blue-Green: Rolling update uses at most N+1 instances, while Blue-Green needs 2N. [C]
- With a rolling update, you can detect problems in the new version while the old version is still running, so you can stop or roll back more easily than with Blue-Green, where the old version may already be deleted. [C]

Disadvantages

- Version skew: Old and new versions run at the same time, which can cause inconsistent versions in production. This can lead to temporal inconsistency or interface mismatch between services. [C]
- The system must be designed to handle mixed versions safely during the rollout. [C] [4]
- Longer update process compared to other methods. [4]

When to use it (use cases)

- When you need service continuity (no service outage during deployment). [4]
- When you want to avoid the high cost of running two full environments like Blue-Green. [C]
- When your service can safely handle old and new versions running together, often using feature toggles to control new behavior. [C]

Blue-Green Deployment

How it works

- First, create N instances of the new version (Service B). [C]
- Then, route incoming requests to Service B instances instead of Service A. [C]
- After traffic is moved, drain and delete instances of Service A. [C]
- This means the system switches traffic from old (blue) to new (green) in one step. [C]

Advantages

- If done correctly, there is no reduction of service to clients during the switch (near-zero downtime). [4] [C]
- Clean and fast switch: Traffic moves all at once from old to new, so users are not slowly mixed between versions. [4] [C]
- Avoid versioning issue, change the entire cluster state in one go [3]

Disadvantages

- High cost: Blue-Green needs 2N instances at peak, while Rolling needs only N+1. [C]
- Slower rollback if old is deleted: If you discover an error after switching, Service A may already be deleted, so rolling back can take time. [C]
- Needs careful timing: If you delete the old version too early, recovery becomes harder. [C]

When to use it (use cases)

- When you need very safe and clean releases with a clear switch point (for example, banking APIs or healthcare systems). [C]
- When you can afford extra servers for a short time (because it needs 2N capacity). [C]
- When you want a clear separation between old and new versions instead of mixing them like in rolling updates. [3] [C]

Canary Deployment

How it works

- Canary testing means selecting a small group of users who will use the new release first. [C]
- These testers can be power users, preview users, or internal testers from the organization. [C]
- The testers get access to the new version through DNS settings or discovery-service configuration. [C]
- After testing, either the system is returned to the old version or the new version is rolled out to all users. [C]
- Most users stay on the old version, and only a small group is sent to the new version. [6] [C]
- There is no full downtime, because only a small part of traffic is affected. [6] [C]

Advantages

- Only a small group of users sees the new version first, so problems affect fewer people. [6] [C]
- The new version is tested by real users, not just in staging. [C]
- Easy decision point: After testing, you can either roll out to everyone or go back to the old version. [6] [C]

Disadvantages

- More complexity: You must run two versions at the same time and control who gets which one. [6] [C]
- Needs good monitoring: You must watch the canary users carefully to decide whether to expand or roll back. [6] [C]

When to use it (use cases)

- When you want to test a risky change on real users before giving it to everyone. [6] [C]
- When you want to limit damage if something goes wrong by exposing only a small group. [6] [C]

A/B Testing Deployment

How it works

- A/B testing is used to run an experiment with real users to see which version gives better business results. [C] [7]
- A small but meaningful number of users get a different version, while the rest stay on the original version. [C] [7]
- The two groups are compared using a business metric (for example, conversion rate or engagement). [C] [7]
- The implementation is the same as canary testing: discovery services send requests to different versions. [C] [7]
- The different versions are monitored to see which one performs better from a business perspective. [C] [7]
- Feature toggles can be used to control which version users see. [C]
- Users are split into groups, and each group sees a different version. [C] [7]
- There is no downtime, because both versions run at the same time. [C] [7]

Advantages

- Data-driven decisions: You can choose the better version based on real user behavior and business metrics. [C] [7]
- Real production testing: Both versions are tested with real users, not just in staging. [C] [7]
- Flexible control: Feature toggles let you easily switch or control which version users see. [C]

Disadvantages

- More complexity: You must run multiple versions at the same time and split users correctly. [C] [7]
- Needs good measurement: If you don’t track the right business metrics, the results can be misleading. [7]
- User experience differences: Different users see different behavior, which can be confusing or unfair if not handled carefully. [C] [7]

When to use it (use cases)

- When you want to compare two or more versions of a feature and pick the one that performs better. [C] [7]
- When the goal is business impact, not just technical correctness. [C] [7]
- When you can split traffic reliably using discovery services or feature toggles. [C]

References:

[C] University of Chicago. (n.d.). Lecture slides: Deployment & Container Orchestration (MPCS 56550: Introduction to DevOps, Winter 2026). University of Chicago MPCS.

[1] https://www.groundcover.com/blog/kubernetes-deployment-strategies

[2] https://medium.com/@muppedaanvesh/rolling-update-recreate-deployment-strategies-in-kubernetes-%EF%B8%8F-327b59f27202

[3] https://blog.container-solutions.com/kubernetes-deployment-strategies

[4] https://octopus.com/devops/kubernetes-deployments/kubernetes-deployment-strategies/

[5] https://www.qovery.com/docs/configuration/deployment/strategies

[6] https://octopus.com/devops/software-deployments/canary-deployment/

[7] https://nearshore-it.eu/articles/deployment-strategies-101-types-pros-cons-devops-more/