COPADO

eBook

Big Bang Release User Guide

Managing High-Volume Salesforce Deployments with Copado

A Strategic Guide for Salesforce Teams

Welcome

Executive Summary

This guide will walk you through executing a "Big Bang" release, a large-scale deployment involving significant metadata changes across your Salesforce environment. Whether you're migrating from a legacy system, implementing a major platform upgrade, or consolidating months of development work, this approach will help you deploy with confidence and control.

What you'll learn:

- How to structure and manage large releases using Copado
- Best practices for validating changes before production deployment
- How to protect your main branch during the release process
- Strategies for transitioning to agile delivery after your initial release

What is a Big Bang Release?

A Big Bang release is a deployment approach where a substantial volume of metadata changes are bundled together and deployed to production in a single, coordinated effort. This contrasts with incremental, continuous delivery models.

When to use this approach:

- ☑ Initial Salesforce implementations with extensive configuration
- ☑ Major platform migrations or consolidations
- ☑ Accumulated development work that must be deployed together due to dependencies
- ☑ Situations where business requirements dictate a specific "go-live" date

Prerequisites

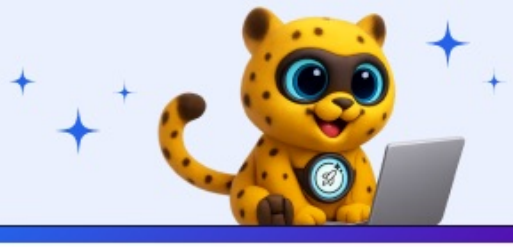
Before you begin, ensure you have the following in place:

Copado Configuration

- Deployment Flow configured with all necessary environments (Development, Integration, QA, UAT, Production)
- Connection Behaviors set up for each environment
- User permissions to create and promote user stories, bundles, and deployments
- Updated Copado packages to take advantage of all current features
- Access to your git repository

Team Readiness

- Development work organized into user stories in Copado
- Sprint planning completed
- Clear understanding of dependencies between components
- Designated release manager or coordinator



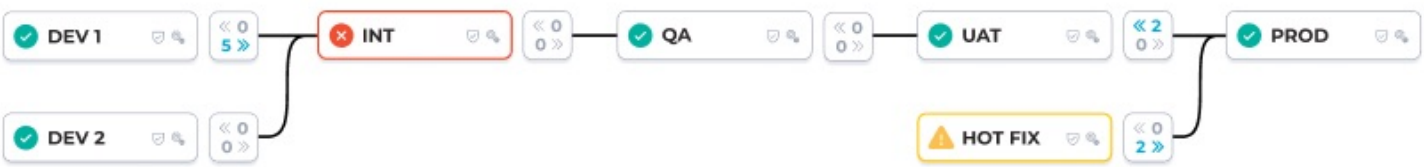
Environment Preparation

- All environments in the deployment flow are accessible and functional
- Integration, QA, and UAT environments are refreshed or aligned with production (if needed)
- Backup of production environment completed

Pre-Copado Metadata (if applicable)

- List of metadata created/updated/deleted before Copado was implemented
- Best practice: Use Copado from the beginning of the project to avoid this complexity

Example Pipeline



Critical Setup:
Protecting Your Main Branch

Before you begin development, you need to set up safeguards to prevent work from accidentally being merged to your main branch during the Big Bang release.

Configure Base Branches

For all user stories during the Big Bang release:

1. Set the Base Branch field to `uat` (not `main`)
2. This ensures all work merges to the UAT branch first
3. Changes will only merge to main when you're ready for production deployment

Set Up Validation Rules

Why this matters: Validation rules prevent developers from inadvertently deploying changes past the base branch directly to main, which could disrupt your controlled release process.

How to configure:

1. Navigate to Setup in your Copado org
2. Go to Object Manager → User Story
3. Create a validation rule with these settings:
   - **Rule Name:** `Prevent_Deployment_Past_Base_Branch`
   - **Error Condition Formula:** Configure to prevent promotion beyond the base branch during the Big Bang period
   - "This user story is part of the Big Bang release and cannot be deployed past UAT until the release is complete."

When to disable: After your Big Bang deployment to production is complete and you transition to agile delivery (see Phase 5 below).

The Big Bang Release Process

The Big Bang approach uses Copado's bundling and promotion capabilities to move your changes through environments in a controlled, sequential manner. Here's how it works:

Sprint 1 Bundle → **Sprint 2 Bundle** → **Sprint 3 Bundle** → **Sprint N Bundle** → **Validate to Production** → **Deploy to Production**

Each sprint's work is packaged into a bundle, validated in sequence, and ultimately deployed to production.

Step-by-Step Instructions

PHASE 1

Sprint Development & Deployment (Repeat for Each Sprint)

For the purposes of this example we will assume the following;

- **2 Project Teams working in Dev 1 and Dev 2 sandboxes**
- **5 developers on each team**
- **Large Deployment after 5 Sprints (~5 months of work)**
- **50 User Stories per Sprint**

Let's walk through the process for Sprint 1. You'll repeat this same process for Sprint 2, 3, 4, 5, and beyond.

Sprint 1: Development (50 User Stories)

1 Configure user stories

- ✓ Set Base Branch = uat on all user stories
- ✓ Verify validation rule is active to prevent deployment past base branch
- ✓ Assign user stories to developers

2 Development work

- ✓ Developers work on their stories in Dev1 and Dev2 sandboxes
- ✓ Best practice for versioned components (Flows, OmniScripts):
  - *Assign to individual developers to avoid conflicts*
- ✓ Best practice for shared components (Layouts, Pages, Apex, Aura):
  - *Use Copado's partial commit process when multiple developers work on the same components*

3 Deploy to Integration

- ✓ Developers deploy changes to Integration as they are completed
- ✓ Developers review and verify changes in Integration environment

4 Deploy to QA

- ✓ Changes are deployed to QA environment
- ✓ Stories are assigned to QA team for testing
- ✓ QA team validates functionality and signs off

5 Deploy to UAT

- ✓ Once QA signs off, deploy changes to UAT
- ✓ Business stakeholders review and sign off on completed items

6 Create Sprint 1 Bundle

Once all Sprint 1 changes are in UAT and signed off:

Navigate to Bundles in Copado

↓

Click New Bundle

↓

Name: `Big Bang Release - Sprint 1`

↓

Add all 50 user stories from Sprint 1

Important: The bundle combines all 50 user stories into a single bundled user story on a new feature branch. This simplifies deployments and allows future bundles to build on it.

7 Validate to Production

- ✓ Create a validation deployment to Production
- ✓ This ensures everything is deployable without actually deploying
- ✓ Review validation results and resolve any issues
- ✓ Do not deploy yet—you're just validating

Sprint 2: Development (50 User Stories)

1 Configure user stories

- ✓ Set Base Branch = uat on all Sprint 2 user stories
- ✓ Validation rule remains active

2 Repeat the development process

- ✓ Developers work in Dev1 and Dev2
- ✓ Deploy to Integration → QA → UAT
- ✓ Business sign-off in UAT

3 Create Sprint 2 Bundle

Name: `Big Bang Release - Sprint 2`

↓

Add all 50 user stories from Sprint 2

↓

CRITICAL STEP:
Set Base Branch of Bundle 2 = Bundle 1 feature branch

- ✓ This creates the chain that ensures smooth merging when all bundles deploy to production
- ✓ Each bundle builds on the previous one

4 Validate all bundles to Production

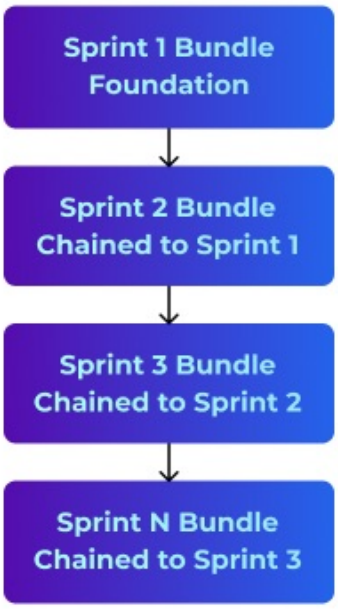
- ✓ Create a new validation to Production
- ✓ Include all bundles (Bundle 1 + Bundle 2)
- ✓ Verify successful validation
- ✓ Do not deploy yet

Sprint 3, 4, 5, etc.: Continue the Pattern

For each subsequent sprint:

1. **Base Branch = uat for all user stories**
2. **Follow the same development → Integration → QA → UAT flow**
3. **Create bundle after UAT sign-off**
4. **Set the bundle's base branch to the previous bundle's feature branch**
   - ✓ Sprint 3 bundle base branch = Sprint 2 bundle feature branch
   - ✓ Sprint 4 bundle base branch = Sprint 3 bundle feature branch
   - ✓ And so on...
5. **Create validation to Production including all bundles created so far**

Visual representation of bundle chaining:



PHASE 2

Final Validation

Before deploying to production, perform a comprehensive final validation.

Pre-deployment validation checklist

- ✓ All bundles successfully deployed and tested in UAT
- ✓ All bundles included in a single validation to Production
- ✓ Validation to Production completed successfully with no errors
- ✓ Business stakeholders have signed off on UAT results
- ✓ Production backup completed
- ✓ Deployment window scheduled and communicated
- ✓ Rollback plan documented and ready

Run final validation

1. Create a promotion that includes all bundles (Sprint 1 through Sprint N)
2. Select Validate Only (do not deploy)
3. Review validation results carefully
4. Resolve any validation errors or warnings
5. Re-validate if changes were needed

PHASE 3

Deploy to Production

This is your final step—deploying all bundles to production.

Production deployment

1. Navigate to your successfully validated deployment
2. Click Deploy to promote the validated deployment to Production
3. Choose your deployment timing:
   - *Deploy Now for immediate deployment*
   - *Schedule for a specific date/time window*
4. Monitor the deployment closely
5. All bundles will deploy in sequence based on their chaining

Post-deployment verification

- Verify critical functionality in production
- Monitor for errors or unexpected behavior
- Communicate success to stakeholders
- Document any issues and resolutions

PHASE 4

Post-Deployment Cleanup

After your successful production deployment, you need to reset your Copado configuration for ongoing agile delivery. Cleanup steps are following:

Refresh all sandboxes

- Refresh Dev1, Dev2, Integration, QA, and UAT from Production
- This ensures all environments are aligned with the new production state

Recreate all org branches

- Reset git branches to align with refreshed environments
- Ensure branch structure is clean

Turn off validation rule

- Disable or delete the validation rule that prevented deployment past base branch
- This rule is no longer needed for agile delivery

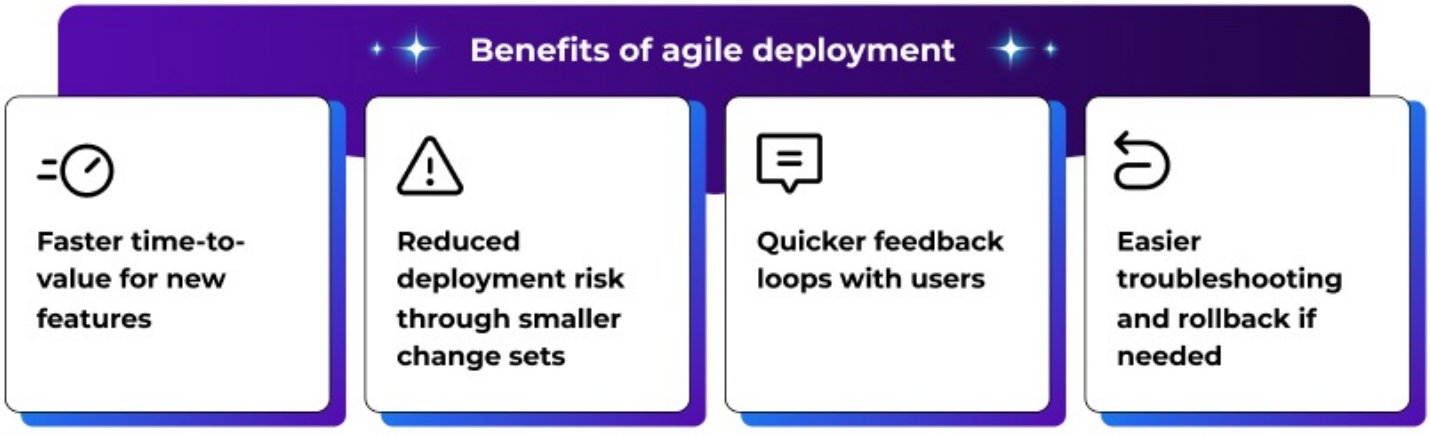
Update base branch configuration

- Change all base branches to `main` for future user stories
- This allows normal continuous delivery flow

PHASE 5

Transition to Agile Deployment

After your Big Bang release, shift to a more agile, continuous delivery model to gain faster feedback and reduce risk.



How to operate in agile mode:

1 Deploy after every sprint (or more frequently) user stories

- ✓ Don't accumulate multiple sprints of work
- ✓ Promote individual user stories or small feature bundles as they're completed

2 Use the standard promotion flow

- ✓ Dev → Integration → QA → UAT → Production
- ✓ Base branch = main for all user stories
- ✓ No validation rules blocking deployment

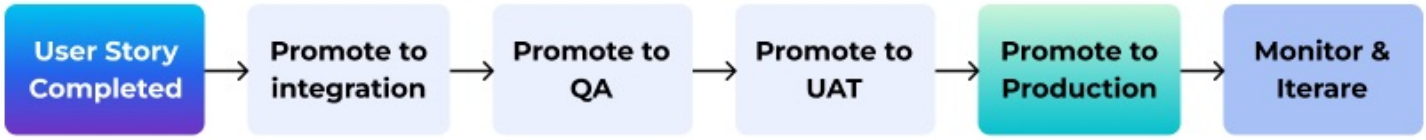
3 Leverage Copado's automation

- ✓ Set up automated validations between environments
- ✓ Use quality gates to ensure only tested changes reach production
- ✓ Consider scheduled deployments for regular release windows

4 Adopt feature flags (optional)

- ✓ Deploy code to production but keep features hidden until ready
- ✓ Gradually roll out features to users
- ✓ Reduce dependency on deployment timing

Example agile flow



Best Practices & Tips for Big Bang Release

PLANNING YOUR RELEASE

1 Set up proper controls

- ✓ Configure base branches to UAT for all Big Bang user stories
- ✓ Implement validation rules to prevent accidental deployment to main
- ✓ Document your bundle chaining strategy upfront

2 Break down work logically

- ✓ Group user stories by sprint or functional area
- ✓ Keep bundles manageable—typically one bundle per sprint
- ✓ Document dependencies between bundles

3 Communicate clearly

- ✓ Keep stakeholders informed of progress through each environment
- ✓ Set clear expectations for deployment windows
- ✓ Document any deviations from the plan

DURING DEVELOPMENT

1 Manage versioned components carefully

- ✓ Assign Flows and OmniScripts to individual developers
- ✓ Avoid multiple developers working on the same versioned component simultaneously

2 Use partial commits for shared components

- ✓ When multiple developers work on Layouts, Pages, Apex, or Aura components in the same sandbox
- ✓ Use Copado's partial commit feature to avoid conflicts

3 Validate frequently

- ✓ Run validations to Production after each sprint's bundle is created
- ✓ Don't wait until the end to discover deployment issues
- ✓ Fix validation errors immediately

DURING DEPLOYMENT

1 Monitor actively

- ✓ Watch deployment logs in real-time
- ✓ Have your team available during production deployment
- ✓ Be ready to execute rollback plan if needed

2 Validate thoroughly

- ✓ Test in each environment before promoting forward
- ✓ Don't skip QA or UAT validation—they're your safety net
- ✓ Involve business users in UAT testing

AFTER DEPLOYMENT

1 Clean up properly

- ✓ Don't forget to refresh sandboxes and recreate branches
- ✓ Remove validation rules that are no longer needed
- ✓ Update base branch settings to main

2 Document lessons learned

- ✓ What went well?
- ✓ What could be improved?
- ✓ How can you streamline the next release?

3 Plan your transition

- ✓ Schedule a retrospective with your team
- ✓ Identify opportunities to move toward continuous delivery
- ✓ Update your processes based on what you learned

Troubleshooting Common Issues

User story won't deploy past UAT

Possible causes

- Validation rule is active and preventing deployment
- Base branch is set to UAT instead of main

Resolution

- **This is expected behavior during Big Bang release**
- **User stories should only reach Production via bundles**
- **If you need to deploy outside the Big Bang process, temporarily disable the validation rule (with proper approval)**

Bundle won't promote

Possible causes

- User stories not committed to version control
- Merge conflicts in the target environment
- Missing permissions or connection issues
- Base branch not set correctly on the bundle

Resolution

- **Verify all user stories show committed metadata**
- **Check that bundle base branch points to the previous bundle's feature branch**
- **Check deployment logs for specific errors**
- **Ensure connection behaviors are correctly configured**

Validation to production fails

Possible causes

- Metadata dependencies not met in Production
- Validation rules or required fields in Production blocking deployment
- Insufficient permissions in Production org
- Bundles not chained correctly

Resolution

- **Review validation error messages carefully**
- **Check for missing dependencies (custom objects, fields, etc.)**
- **Verify bundle chaining—each bundle should build on the previous one**
- **Consider deploying dependencies in an earlier bundle**
- **Test in UAT thoroughly before validating to Production**

Merge conflicts between bundles

Possible causes

- Multiple developers modified the same component in different sprints
- Bundle base branches not set correctly
- Changes not properly merged through the chain

Resolution

- **Verify bundle chaining is correct (Sprint 2 base = Sprint 1 feature branch, etc.)**
- **Resolve conflicts in git repository**
- **Re-validate after resolving conflicts**
- **Consider using Copado's conflict resolution tools**

Summary

You've learned how to:

- Protect your main branch with validation rules during Big Bang releases
- Configure base branches correctly for controlled releases
- Structure a Big Bang release using Copado bundles
- Chain bundles to maintain deployment sequence and smooth merging
- Validate to Production before deploying
- Deploy to production with confidence
- Clean up and transition to agile, continuous delivery

Remember: While Big Bang releases are sometimes necessary, continuous delivery offers faster value and lower risk. Use this approach when needed, then evolve toward more frequent, smaller deployments.

Good luck with your release!

Need Help?

If you encounter issues not covered in this guide:

- **Copado Documentation:** docs.copado.com
- **Copado Support:** Open a case through your Copado org
- **Copado Professional Services:** Contact your Copado Architect
- **Community:** Copado Community forums for peer support
- **Copado Salesforce Alliances Team**