Copado Deployment issue due to destructive commit.

There is an issue with Copado. Instead of doing the destructive commit first, Copado does the metadata commit first and then the destructive commit that causes deployment failures as well issues during back promotion failures:

An example is the **US-0679174**

Details of the SPA-2639 (**US-0679174**):

Requirement: Create a criteria-based sharing rule on case objects to share specific records with a group of people when they met certain criteria.

Challenge: There are 50 criteria-based sharing rules already present on the case object. As per Salesforce limitation, one can create a max of 50 criteria-based sharing rules.

Resolution:

1. Conducted a quick analysis of all the existing criteria-sharing rules on case objects.

2. Identified the possibility of merging three existing criteria-based sharing rules into single criteria rules. So that it frees up some limits and allows us to create new criteria-based sharing rules as per the requirement.

3. As part of the user story commit process,

a. Committed the new sharing rules and public groups (These changes are upsert operations)

b. Committed the destructive change, selected the environment where the criteria rules are still available (in this case INT environment, because the existing changes should be removed in the dev sandbox to create a new sharing rule)

4. When the deployment got initiated (promote and deploy) to higher environments, copado creates the deployment steps. As per official documentation, copado created a deployment step and one cannot have control over its order.

5. As observed, copado gives priority to upsert operations, and the new sharing rules are getting deployed to the higher environments and causing the limit issue (i.e., 52 which is a no-no in the Salesforce ecosystem)

6. So, after the deployment failure, one can re-order the deployment steps to prioritize the deletion operation over the upsert operation.

7. And finally, deploy all the changes to re-initiate the deployment process. By now, deployment steps are reordered, Deletion happens first (so, sharing rules count gets 47 by this time) and next when the upsert operation gets executed, there will be no limit issue that comes along the way. So, Deployment succeeded.

Step 6 is the workaround here. The deletion of the metadata should be done first and then the commit should be done. So we should go to the deployment record and reorder the sequence.