# Updating Commits:

L 🡪 Local commits

R 🡪 Remote commits

I 🡪 Commits to be inserted

D 🡪 Commits to be deleted

Considering an iteration that is verifying a monitored repository rep1, the steps needed to update its commits in the database are the following:

1. Retrieve the previous local snapshot from disk 🡺L1
2. Retrieve current snapshot from repository 🡺 L2
3. Identify new local commits, comparing current and previous snapshots 🡺 L3 = L2 \ L1
4. Identify commits that were deleted locally since previous run: D1 = L1 \ L2
5. Retrieve all commits from database that do not exist in at least one of the repositories related to rep1, this is, commits that were not found either in rep1 or in at least one of its partners. All other commits in the database are considered to be already synchronized 🡺 R1
6. Identify which of L3 commits do not exist in R1 🡺 I1 = L3 \ R1
7. Insert commits from I1 in the database
8. Delete commits from D1 in the database
9. Save current snapshot to disk
10. For each one of R1 commits, update the list of repositories where it is found, according to the discussion in section 2
11. Update commits from R1 in the database

# Updating the list of repositories where a commit is found

Considering a single commit X, the following table shows what can be concluded regarding its existence in the repositories related to the monitored repository. The ahead lists contain commits that exist in the monitored repository and do not exist in its partner (one list per partner). The behind list contain commits do not exist in the monitored repository but exist in any of its partners (one list per partner).

Lines marked with 'Not valid' denote impossible situations, once that a single commit X cannot be both ahead AND behind (it exists or does not exist locally). This is, if X exists locally, its status will be whether ahead (if it also exists remotely) of synchronized. It will never be behind, because it exists locally. The same reasoning applies where X does not exist locally: it will be whether synchronized or behind, but never ahead.

Table 1 – Commit existence based on behind and ahead lists

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Ahead List | | Behind List | | Existence | | |
| rep A | rep B | rep A | rep B | Local | rep A | rep B |
| F | F | F | F | see conclusion 1 below | | |
| F | F | F | T | F | F | T |
| F | F | T | F | F | T | F |
| F | F | T | T | F | T | T |
| F | T | F | F | T | T | F |
| F | T | F | T | Not valid | | |
| F | T | T | F | Not valid | | |
| F | T | T | T | Not valid | | |
| T | F | F | F | T | F | T |
| T | F | F | T | Not valid | | |
| T | F | T | F | Not valid | | |
| T | F | T | T | Not valid | | |
| T | T | F | F | T | F | F |
| T | T | F | T | Not valid | | |
| T | T | T | F | Not valid | | |
| T | T | T | T | Not valid | | |

Conclusions:

1. If commit is not in any list and exists locally, then it also exists in all repositories in the push / pull lists, otherwise it does not exist in any of the repositories in the push / pull lists;
2. If commit is in at least one behind list, then it does not exist locally and exists in all repositories that have at least one behind list containing it;
3. If commit is in at least one ahead list, then it exists locally and in in all repositories that DO NOT have an ahead list containing it.