The merge conflict

Tom Rochette <tom.rochette@coreteks.org>

December 21, 2025 — 3b313fe0

Dr. Sarah Chen stared at her terminal, the familiar green text glowing in the darkened lab.

```
$ git log --all --graph --oneline
* a3f9b2c (HEAD -> main) Fix climate models
| * 7c8d1e4 (origin/2157) Prevent asteroid impact
|/
* 2b4a8f3 Initial timeline
```

“Three timelines,” she muttered. “And they’re diverging.”

The Temporal Version Control System had seemed like humanity’s salvation. Jump to any point in history, create a branch, make changes, then merge back. Fix mistakes. Optimize outcomes. What could go wrong?

Everything, apparently.

Sarah’s colleague Marcus rushed in. “We’ve got a problem. The 2157 branch where we prevented the asteroid? It created a merge conflict with main.”

“Show me.”

```
$ git merge origin/2157
Auto-merging timeline.dat
CONFLICT (content): Merge conflict in timeline.dat
Automatic merge failed; fix conflicts and then commit the result.
```

Sarah pulled up the diff:

```
<<<<<<< HEAD
2089: Global climate stabilized, population 9.2B
2157: Thriving lunar colonies established
=======
2089: Asteroid prevention tech drives new space race
2157: Mars terraforming 40% complete, population 12.7B
>>>>>>> origin/2157
```

“They’re both real,” Marcus whispered. “Both timelines exist simultaneously until we resolve the conflict.”

Sarah nodded slowly. Quantum superposition at a temporal scale. The universe itself refusing to compile until they chose which future to keep-and which to discard.

Her fingers hovered over the keyboard. One timeline solved climate change through sacrifice and discipline. The other achieved it through desperate innovation sparked by near-extinction.

“What if,” she said, “we don’t choose?”

“You can’t leave a merge conflict unresolved. The timeline will remain in an unstable state-”

“Or we `git rebase` everything onto a new branch. Cherry-pick the best commits from each timeline.”

Marcus’s eyes widened. “You want to rewrite history itself.”

“We already are. We’ve just been doing it badly.” Sarah started typing:

```
$ git checkout -b unified
$ git cherry-pick 7c8d1e4  # Asteroid prevention tech
$ git cherry-pick a3f9b2c  # Climate stability
```

The lab hummed. Reality flickered.

When the command completed, Sarah checked the log:

```
* e9f2a1b (HEAD -> unified) Climate models + prevention tech
* 2b4a8f3 Initial timeline
```

Clean. Linear. Optimal.

“Git push –force?” Marcus asked nervously.

Sarah smiled. “Git push –force.”

She hit enter.

The universe accepted the merge.