Predicting Development Trajectories to Prevent Collaboration Conflicts

Yuriy Brun†, Kivanç Muşlu†, Reid Holmes‡, Michael D. Ernst†, and David Notkin†
†University of Washington
‡University of Waterloo

Problem: Collaboration Conflicts

• Version control helps developers work concurrently.
• Conflicts offset the benefits of concurrent work.
• Collaboration conflicts distract developers from their core tasks.

State-of-the-Practice: Detect on Merge

• Textual conflicts are discovered during merges.
• Compilation and testing conflicts are often discovered even later.

Shortcomings:

• Version control isolates developers from others’ actions until they try to merge.
• Merging is often painful.
• The longer it takes to discover a merge conflict, the more difficult it is to resolve.
• Fear of conflicts often discourages concurrent work [4].

State-of-the-Art: Detect before Merge

• Awareness tools estimate when developers’ changes interact.
  o FastDASH [1], Syde [5], CollabVS [3], Safe-commit [7], YooHoo [6].
• By computing future merges, Crystal [2] reports conflicts precisely, eliminating false positives and negatives.
  o speculatively merges code in the background
  o detects textual, compilation, and testing conflicts
  o informs developers of conflicts as soon as they are created

Our Vision: Predict before Creation

Today: Analyze actual changes to detect conflicts.
Tomorrow: Predict and analyze future changes.

Goal: warn before a potential conflict is created

Legend:
→ Alice continues course to a merge conflict
→ Alice changes course to avoid impending conflict
→ Alice changes course much earlier to avoid impending conflict

References: