Einführung in die Softwaretechnik 2018
Sheet 10

Maximilian Frühauf

8th July 2018

1. You are working in a new team with unexperienced developers. Your new colleague Bob wants to share his changes using git. However, other developers cannot find the changes. Explain 3 typical reasons for this problem. Explain for each reason how the problem could be solved.

   - The changes have not been added to the staging area. Any change has to be added to the git staging area with the `git add <FILE>` command.
   - The changes have not been commited. Any staged changes have to be commited to the version control system (git) this packages them up as a single unit and creates a separate version from these files. Any further changes to this version are prohibited.

     Changes can be commited with the `git commit -m "commit message"` command.
   - The changes have not been pushed. After changes have been commited they are still local to Bob's computer. For the other developers to see them, they have to be commited to the central version control server.

     This can be done with the `git push origin` command.

2. After you have explained how to share the changes, Bob has a merge conflict. Explain why this merge conflict could have happened. Further, explain how to prevent such merge conflicts in the future in your own words using best practices for distributed version control.

   A merge conflict can happen if multiple commits change the same parts of a file and push the changes to the remote server. Then git cannot find a definite ordering for both commits. Therefore it is up to Bob to manually select the right changes. After the desired changes have been selected, a merge commit is created, merging the two diverged branches back into one.

   A merge conflict can be prevented by separating the logical parts of a system into separate files. Then if parts of a system have to be changed, these changes stay local to the corresponding files and git can automatically merge the commits. However this is only possible if the developers on the team stick to the convention of handling one feature at a time.

3. The following figure shows an informal model of the build and release management workflow. How would you model this as an object model?

   Solution on Artemis.