[**1 冲突的产生 1**](#_Toc436230933)

[**2 确认历史日志是否最新 2**](#_Toc436230934)

[**3 保存工作现场 3**](#_Toc436230935)

[**4 图形工具完成冲突解决 4**](#_Toc436230936)

[**4.1 图形工具P4Merge安装与配置 4**](#_Toc436230937)

[**4.2 图形工具完成冲突解决（冲突单对应的提交log是本地库中的最新log） 5**](#_Toc436230938)

[**4.3 图形工具完成冲突解决（冲突单对应的提交log不是本地库中的最新log） 8**](#_Toc436230939)

[**5手工编辑完成冲突解决 10**](#_Toc436230940)

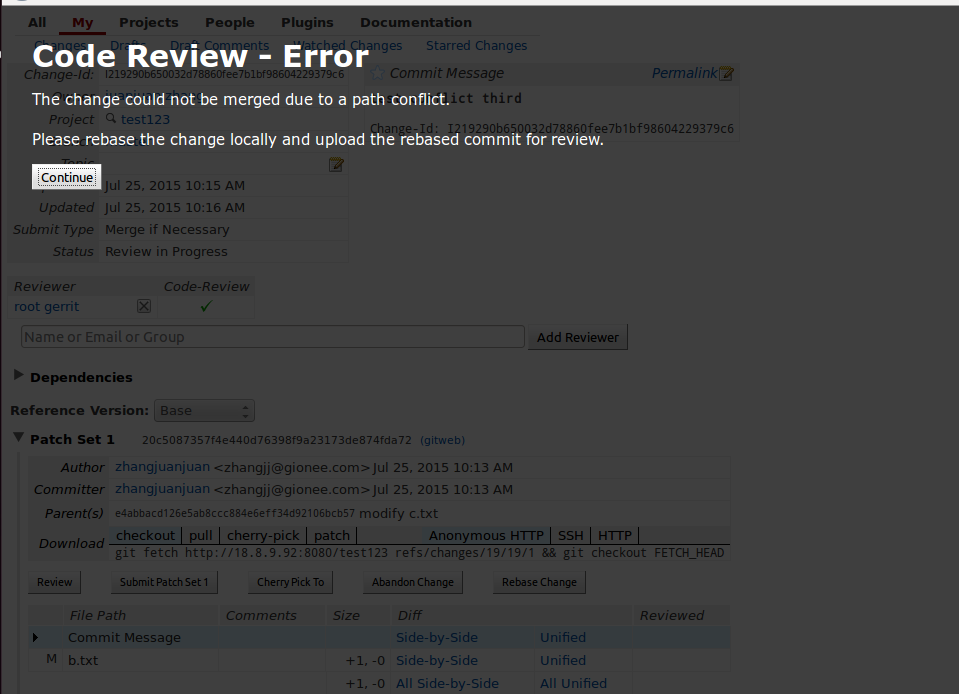
[**5.1 （手工编辑）历史日志最新 10**](#_Toc436230941)

[**5.2 （手工编辑）历史日志不是最新 12**](#_Toc436230942)

**git解决冲突文档**

# 1 冲突的产生

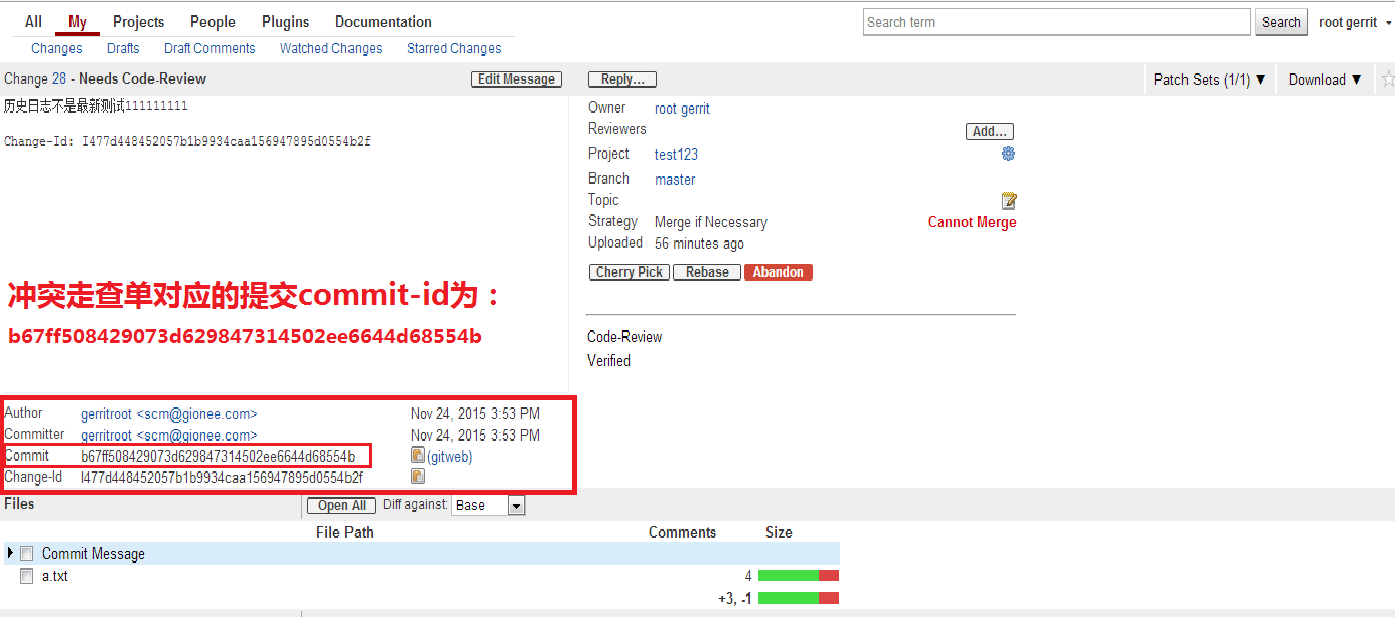
用户A和用户B修改了相同文件的相同区域，管理员先将用户A提交的修改合入代码库，合入过程是正常的，再将用户B提交的修改提交代码库时会可能提示冲突。



# 2 确认历史日志是否最新

先通过git log查看历史日志顺序，确认冲突走查单对应的那次提交是本地库的最新提交，历史日志位于最上面，按照3.1或者4.1操作 ，如果不是，按照3.2或者4.2进行操作。

（1）进入Gerritroot界面查看冲突走查单所对应的commit-id。

****

（2）在本地版本库中执行**git log** 查看历史日志顺序，如下如所示：



此图中Change-Id: I477d448452057b1b9934caa156947895d0554b2f对应的历史日志位于第3，不是最新。

# 3 保存工作现场

如果在执行更新之前，发现本地库中有修改未提交的文件，也就是说，第一次修改提交推送的代码单还未走查完毕，已经在本地库开始新的修改。

此时你收到第一次修改代码走查不通过的邮件，现在要紧急处理走查不通过的问题，但是本地工作区又有新修改，这些新修改还未提交到本地库，但是现在又要更新本地库，在更新之前，先将第二次的修改缓存起来。

$ **git stash** (保存当前的工作现场)

在解决完上一次走查不通的代码之后，恢复工作现场

$**git stash pop（**恢复工作现场**）**

# 4 图形工具完成冲突解决

当冲突文件较多时，我们使用VI编辑器会比较麻烦，那我们使用P4Merge进行冲突文件的融合，会更加直观。

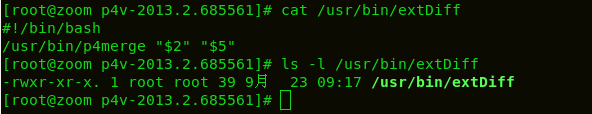
## 4.1 图形工具P4Merge安装与配置

（1）<http://192.168.110.97/svn/svn/trunk/git_gerrit/tools/p4v.tgz>取到本地，将p4v.tgz进行解压，得到lib和bin两个文件，将这两个文件夹覆盖到/usr/目录，操作如下：

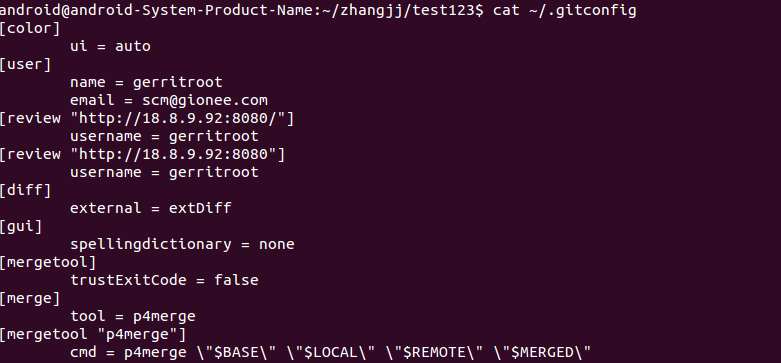


（2）P4Merge的配置

在/usr/bin/目录下，创建文件extDiff，文件内容如下，并执行chmod +x /usr/bin/extDiff赋予执行权限:



（3） 编辑~/.gitconfig文件，内容如下：



**注意：步骤1 2 3 只需要配置一次就ok了。**

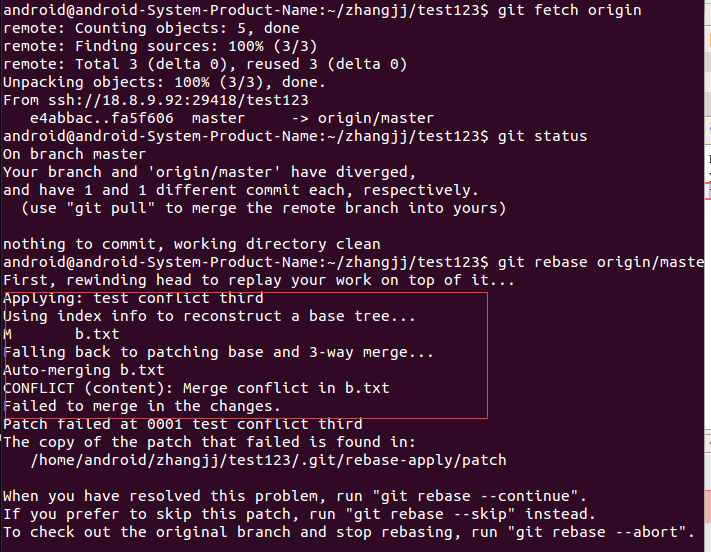
## 4.2 图形工具完成冲突解决（冲突单对应的提交log是本地库中的最新log）

使用P4Merge解决冲突的具体过程如下：

**1) cd test**

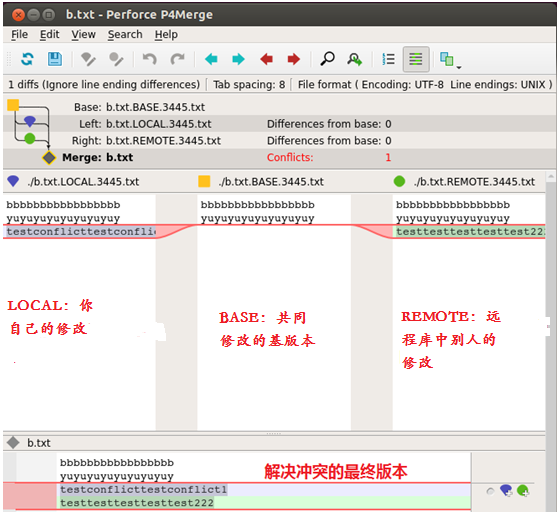
**2) git fetch origin（获取最新版本到本地）**

**3) git rebase origin/branchname（branchname根据需求填写，将本地分支变基到远程分支）**



**4)** **git mergetool**

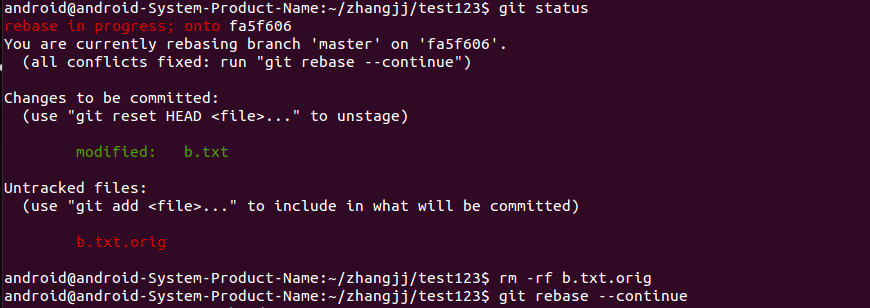
启动GUI进行代码的融合：中间三个区域分别对应着LOCAL BASE REMOTE三个版本（其中，BASE为共同的基版本，LOCAL你自己的修改，REMOTE为远程代码库中别人的修改），最下面的区域即为解决冲突后的最终版本，保存退出。



**5)git status**

**6)** **rm –rf b.txt.orig** （解决好冲突文件后，将.orig文件进行删除）

**7) git rebase --continue** （此处mergetool已经自动为我们执行了git add 命令，无需再执行该命令**）**



**8) git status**

**# On branch master.**

**# Your branch is ahead of 'origin/master' by 1 commit.**

**# (use "git push" to publish your local commits)**

**9)git push origin branchname:refs/for/branchname （ branchname根据实际分支名填写，推送成功之后在原来的冲突单中会多一个patch包，走查人员走查新的patch包合入即可）**

## 4.3 图形工具完成冲突解决（冲突单对应的提交log不是本地库中的最新log）

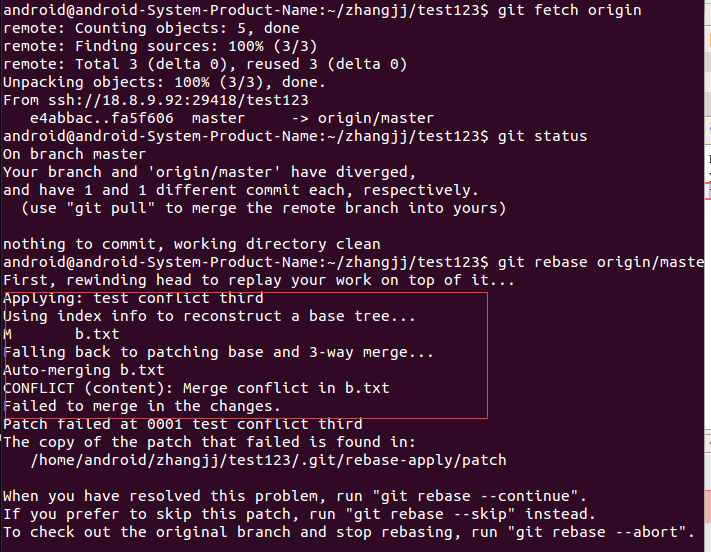
使用P4Merge解决冲突的具体过程如下：

1. **cd test**

**2）git checkout commit-id (commit-id用冲突走查单对应的id替换)**

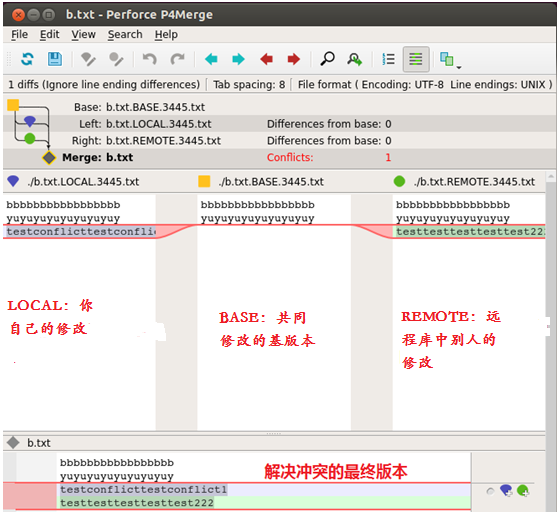
**3) git fetch origin（获取最新版本到本地）**

**4) git rebase origin/branchname（branchname根据需求填写，将本地分支变基到远程分支）**



**5)** **git mergetool**

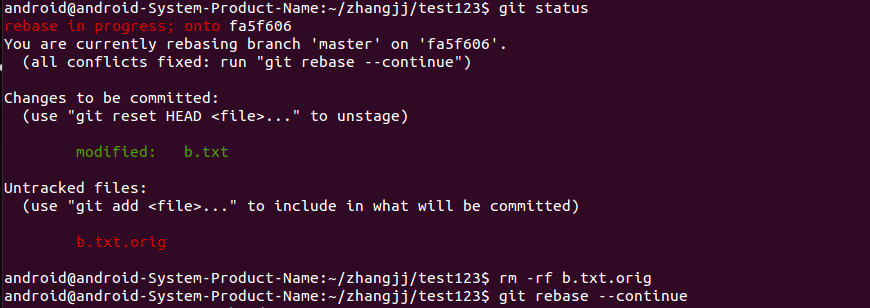
启动GUI进行代码的融合：中间三个区域分别对应着LOCAL BASE REMOTE三个版本（其中，BASE为共同的基版本，LOCAL你自己的修改，REMOTE为远程代码库中别人的修改），最下面的区域即为解决冲突后的最终版本，保存退出。



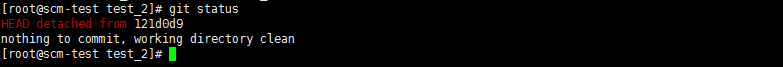
**6)git status**

**7)** **rm –rf b.txt.orig** （解决好冲突文件后，将.orig文件进行删除）

**8) git rebase --continue** （此处mergetool已经自动为我们执行了git add 命令，无需再执行该命令**）**



**9) git status**



**10) git push origin HEAD:refs/for/branchname （ branchname根据实际分支名填写，推送成功之后在原来的冲突单中会多一个patch包，走查人员走查新的patch包合入即可）**

**冲突解决完成后，可以执行git checkout branchname （branchname根据实际分支名填写），将分支branchname检出到工作区继续工作。**

# 5手工编辑完成冲突解决

冲突较少时使用vi等编辑器手工编辑是最快的解决方法。

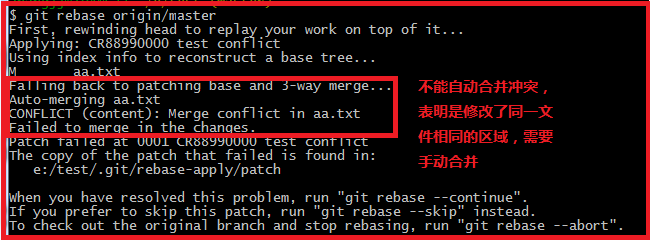
## 5.1 手工编辑（冲突单对应的提交log是本地库中的最新log）

冲突走查单对应的那次提交是本地库的最新提交，历史日志位于最上面，则按照以下步骤操作：

**1) cd test**

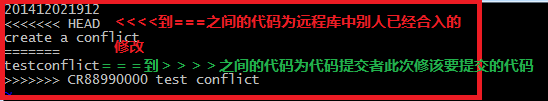
**2) git fetch origin（获取最新版本到本地）**

**3) git rebase origin/branchname（branchname根据需求填写，将本地分支变基到远程分支）**

****

**4)修改冲突文件aa.txt**

**使用vi或其他编辑器编辑，修改之前要搞清楚文件里面的哪些部分是别人修改，哪些是自己的修改，然后调整自己的修改，删除冲突标识符所在的行＜＜＜　＝＝＝　＞＞＞，保存退出。**

****

**5)git status**

**6) git add aa.txt**

**7) git status**

**8) git rebase --continue （完成变基后会自动执行git commit命令，不需要再执行该命令，可直接push）**

**9) git status**

**# On branch master.**

**# Your branch is ahead of 'origin/master' by 1 commit.**

**# (use "git push" to publish your local commits)**

**10)git push origin branchname:refs/for/branchname （ branchname根据实际分支名填写，推送成功之后在原来的冲突单中会多一个patch包，走查人员走查新的patch包合入即可）**

## 5.2 手工编辑（冲突单对应的提交log不是本地库中的最新log）

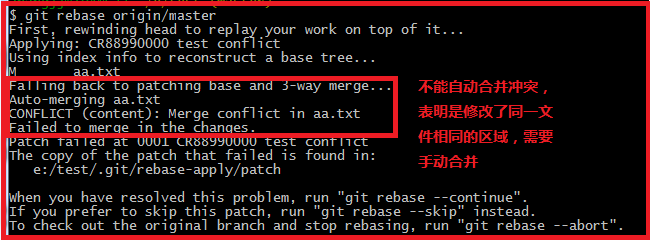
冲突走查单对应的那次提交不是本地库的最新提交，历史日志不是位于最上面，则按照以下步骤操作：

**1) cd test**

**2）git checkout commit-id (commit-id用冲突走查单对应的id替换)**

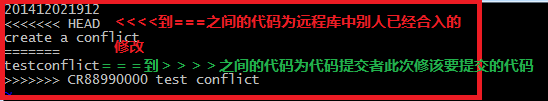
**3) git fetch origin（获取最新版本到本地）**

**4) git rebase origin/branchname（branchname根据需求填写，将本地分支变基到远程分支）**

****

**5)修改冲突文件aa.txt**

**使用vi或其他编辑器编辑，修改之前要搞清楚文件里面的哪些部分是别人修改，哪些是自己的修改，然后调整自己的修改，删除冲突标识符所在的行＜＜＜　＝＝＝　＞＞＞，保存退出。**

****

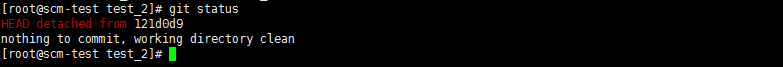
**6)git status**

**7) git add aa.txt**

**8) git status**

**9) git rebase --continue （完成变基后会自动执行git commit命令，不需要再执行该命令，可直接push）**

**10) git status**



**11)git push origin HEAD:refs/for/branchname （ branchname根据实际分支名填写，推送成功之后在原来的冲突单中会多一个patch包，走查人员走查新的patch包合入即可）**

**冲突解决完成后，可以执行git checkout branchname （branchname根据实际分支名填写），将分支branchname检出到工作区继续工作。**