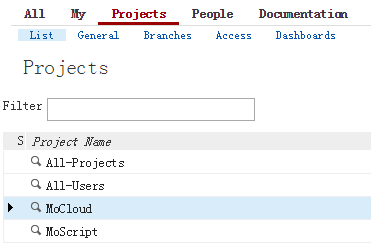
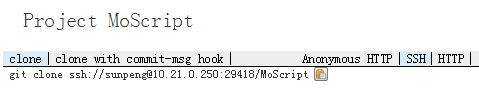
Gerrit使用指南

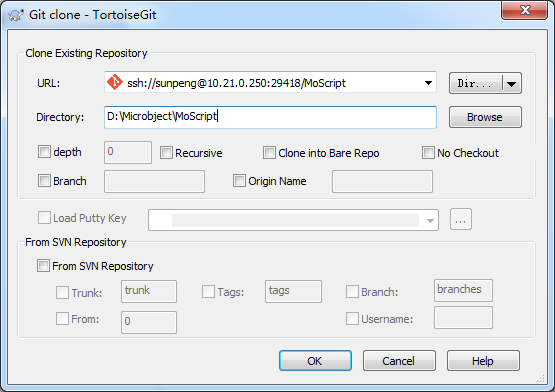
1. 准备工作
   1. 已经按照“【eai】Gerrit配置指南.docx”完成Gerrit的设置。
2. 创建代码仓。
   1. 创建空目录。建议在D盘下创建名为Microbject的空目录。
   2. 登录Girrit站点（<http://10.13.0.250/>），依次点击Projects- → List出现如下图面



* 1. 在Project Name 列表里点击对象Project，出现如下示例内容



* 1. 点击表头的“SSH”之后再点击地址后补的复制图标。这样就把对象代码仓的地址复制出来。
  2. 在创建好的空目录里，点击右键，选择“Git Clone”后出现如下示例界面。



* 1. 在上述界面上点击“OK”即可把对象Project的代码克隆到本地。
  2. **为了能够正常push代码，每次clone之后必须执行以下步骤。**
     1. 进入到clone Project目录下。

例如clone Project为MoCloud，2.1创建的空目录为D:\ Microbject；则clone Project的目录为D:\ Microbject\MoCloud\

* + 1. 在2.7.1置顶的目录下点击右键，执行Git Bash，打开命令窗口。

注：打开Git Bash的位置应该是对象Project的代码仓的根目录下。

* + 1. 在打开的Git Bas命令窗口中输入如下命令并执行。完成后将在clone Project的目录下创建.\ .git\hooks\commit-msg文件

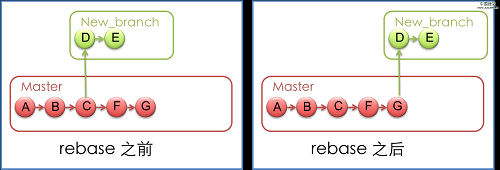
Scp -p -P 29418 [Gerrit用户名]@10.13.0.250:hooks/commit-msg .git/hooks/

示例：Gerrit用户名为user1则命令为

Scp -p -P 29418 [user1@10.13.0.250:hooks/commit-msg .git/hooks/](mailto:user1@10.13.0.250:hooks/commit-msg%20.git/hooks/)

**注意：上述命令最好手动录入，COPY的时候可能会执行失败。**

1. 代码的更新、提交与审核设定。
   1. 代码更新
      1. Fetch，rebase方式更新代码。
         1. 在对象Project目录内，点击右键选TortoiseGit → Ferch 更新代码。
         2. 代码更新后，执行TortoiseGit → rebase来修改本地分支指向的位置



* + 1. Pull更新**禁用**。在对象Project目录内，点击右键选TortoiseGit → pull更新代码。这种更新会自动merger，容易造成本地修改的代码丢失，因此**禁用**。
  1. 代码提交
     1. TortoiseGit → commit 把修改的内容提交到本地分支上。注：当前修改没有提交到服务器的代码库中。
     2. TortoiseGit → push把代码提交到服务器的代码库中。

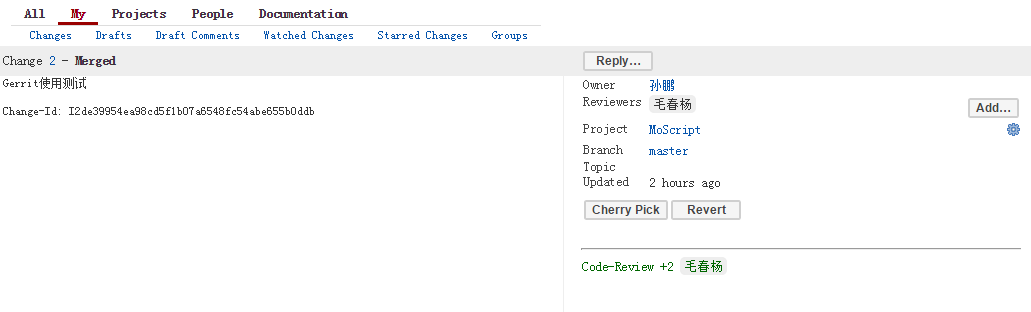
提交时需要指定本地的代码分支（Local）和服务器端代码分支（Remote）



说明：提交进行code review的分支应该是refs/for/dev(或其他分支名)

注：提交分支不清楚的情况下向组长询问。

* 1. 代码审核。通过TortoiseGit → push提交代码之后，要进行代码审核。审核流程如下。
     1. 登录Girrit站点（<http://10.13.0.250/>），点击My → Changes出现如下示例界面。



* + 1. 点击右侧Reviewers -> Add.. 按钮，输入提请审查的人员（名字、账号、Email皆可）。
    2. 审查人如果在系统上登录了邮箱，系统会自动邮件通知。但为了防止人肉通知相关人员审查代码。
    3. 审查通过后提交会列在My -> Changes -> Recently closed中。

1. 审核
   1. 前提

审核的负责人

* 1. 审核通过
  2. 审核未通过

1. 其它
   1. GIT是分布式代码管理系统，大部分操作都不需要服务器的参与，只有少数操作需要服务器或第三方的参与(push, fetch, pull, clone)。
   2. Push时有直接提交权限的用户组可直接用分支名提交无需经过审核。