Contrôle de Source (version control) dans les équipes H2I et EMO au LHEEA

* 1. Logiciels
     1. Gerrit

Gestion administrateur du code (Code Review)

Les commits des développeurs sont pushés une fois seulement que tout le monde est d’accord.

* + 1. Git

En local pour chaque développeur (Version Control).

Les commits sont pushés sur le serveur Gerrit mais pas dans le dépôt. Une ID est donnée à chaque commit pour que Gerrit reconnaisse les commits modifiés resoumis et les définisse comme des patch sets (voir hook)

Les autres développeurs peuvent fetcher ces commits, les modifier et les recommiter = ils seront reconnus par Gerrit comme des patch sets

Outils en local

* Git Bash clonage, fetch des commits des autres développeurs
* Gitk visualisation de l’arborescence
* Git Gui opérations locales de développement (commit, push…)

Commandes principales lancées dans Git Bash ou via Git Gui

* git clone pour clôner un dépôt depuis le serveur Gerrit
* git checkout pour se déplacer dans les commits ou dans les branches ou pour créer une branche
* git commit
  + 1. TortoiseGit

En local, permet dans l’explorateur Windows d’avoir des raccourcis pour renommer les fichiers, ajouter des fichiers,…

* 1. Installations

Compte Gerrit

* serveur gerrit.lheea.ec-nantes.fr
* identifiants de connexion
* Rafraîchir (F5) après entrée des identifiants

Gerrit

* menu Projects/List pour afficher la liste des dépôts auxquels on a accès
* menu Settings accessible en cliquant en haut à droite sur la flèche à côté du nom utilisateur
  + SSH Public Keys
* dans chaque dépôt clôné, il faut ajouter un hook qui ajoutera l’instruction change-ID au commit par Git-Gui (fichier « commit-msg.changeID » à mettre dans .git\hook\ en enlevant l’extension .changeID ou clonage incluant ce hook)
* menu All/Open pour afficher une liste des commits

Interface utilisateur Windows (Git Gui, Gitk, Git Bash)

* download sur site web <https://git-scm.com/>
* menu Downloads
* menu Windows
* enregistrer le fichier Git-2.11.0.3-64-bit.exe
* installer en tant qu’administrateur (plusieurs options à choisir, les choix par défaut sont OK. J’ai choisi de tester difftool)
* vérifier la présence dans l’explorateur Windows des menus « Git GUI Here » et « Git BASH Here » lors du clic droit sur un dossier

Clé SSH. Nécessaire pour la communication entre Gerrit et Git qui se fait via ssh plutôt que http. Pour l’obtenir

* lancer Git Gui,
* menu Help/Show SSH Key/Generate Key, pour générer la paire de clés (une clé publique qu’on donne au serveur gerrit (extension .pub), une clé privée qu’on donne à Tortoise, sans extension)
* copier la clé publique générée (sans espace supplémentaire)
* dans Gerrit, aller dans le menu Settings/SSH Public keys, coller la clé et cliquer sur Add
* Si un message d’erreur apparaît, tenter de se déconnecter de Gerrit et de se reconnecter. La clé a pu être prise en compte malgré le message d’erreur.
* Pour configurer Tortoise Git, clic droit sur un dossier/Tortoise Git/Settings, menu Git/Remote cliquer sur « origin », entrer la clé privée créée ci-dessus. Sauvegarder en cliquant Save.
  1. Utilisation
     1. Clonage
* dans Gerrit, aller dans Projects/List et cliquer sur le dépôt à clôner
* choisir entre un clone simple (cliquer sur menu « clone ») ou un clone avec hook (menu « clone with commit msg »). Le second choix est préférable, ça évite une étape à faire ensuite.
* cliquer sur le menu « ssh » et copier la ligne de commande

|  |  |
| --- | --- |
| Exemple | git clone ssh://bonnefoy@git.intranet.lheea.ec-nantes.fr:29418/Rienecker\_Fenton && scp -p -P 29418 [bonnefoy@git.intranet.lheea.ec-nantes.fr:hooks/commit-msg Rienecker\_Fenton/.git/hooks/](mailto:bonnefoy@git.intranet.lheea.ec-nantes.fr:hooks/commit-msg%20Rienecker_Fenton/.git/hooks/) |

* lancer Git Bash sur le répertoire où sera clôné le dépôt (clic droit sur le répertoire dans un explorateur Windows, menu Git Bash Here
* coller la ligne de commande
* répondre « yes » à la question sur l’authenticité
* on peut lancer ensuite Gitk à l’aide de la commande « gitk --all& »
* Si on a choisi un clonage simple, alors on doit, dans le clone, ajouter le fichier « commit-msg.changeID » dans le répertoire .git\hook\ en enlevant l’extension « changeID »
  + 1. Fetch
* dans Gerrit, menu All/Open, la liste des commits des développeurs s’affiche
* en cliquant sur l’un des commits, on obtient des détails
  + à gauche, le texte du commit et la liste des fichiers modifiés
  + à droite, on retrouve la liste des commits
* en cliquant sur le commit le plus récent (ou un autre qu’on veut fetcher particulièrement), on clique sur le menu « Download » en haut à droite, on copie la commande intitulée Checkout

|  |  |
| --- | --- |
| Exemple | git fetch ssh://bonnefoy@git.intranet.lheea.ec-nantes.fr:29418/Rienecker\_Fenton refs/changes/34/634/1 && git checkout FETCH\_HEAD |

* dans Git Bash (ouvert dans le répertoire du dépôt), on copie cette ligne
* le fetch est fait dans un état indéterminé « detached HEAD » (voir le message affiché dans Git Bash, toujours de bon conseil) : il faut se placer dans une branche.
* on se place dans une branche (sinon,…). Pour cela
  + on se déplace dans une branche existante « git checkout BRANCH\_NAME »
  + on crée une nouvelle branche
    1. Commit local (depuis Git Gui)
* les commits doivent faits de la manière suivante
  + petits commits plutôt qu’un gros
  + on peut commiter ligne par ligne (Git Gui\Indexer les lignes)
* Le texte du commit obéit aux règles suivantes
  + Première ligne de titre
  + Lignes suivantes de description détaillée
  + Signature obligatoire (cliquer sur le bouton Signer qui ajoute l’option –S à la commande « git commit »)
  + Instruction ChangeID (ajoutée automatiquement grâce au hook)
    1. Push
* dans Git Bash (ouvert dans le répertoire du dépôt), l’instruction est

|  |  |
| --- | --- |
| Exemple | git push origin HEAD:refs/for/master |

* le mot « origin »
* le mot « master » est le nom de la branche du serveur sur laquelle on veut commiter sous Gerrit
  1. Idées
* on peut signer automatiquement les commits : voir par exemple les samples de hook prepare-commit-msg et commit-msg
* cette procédure sous Git/Gerrit (accès des nouveaux ?) dans Basin\_Tools