**CM GIT Process Documentation**

CM may need to clone repository when the CM support person who is going to work on the issue is not the original creator of the repository and will to run clonerepository.shl in z/github/cm in git bash.

**CM Creation for Issue and Branch**

1. Create issue testrepr.#1 in github after developer has sent a creation request message (See sample below):

Please Create a GITHUB issue as follows:

Project /Repository:                                      testam

Issue Type:                                                     New Development

Project/Service Ticket Number:                13-0026

Description:                                                     GitHub Testing Project

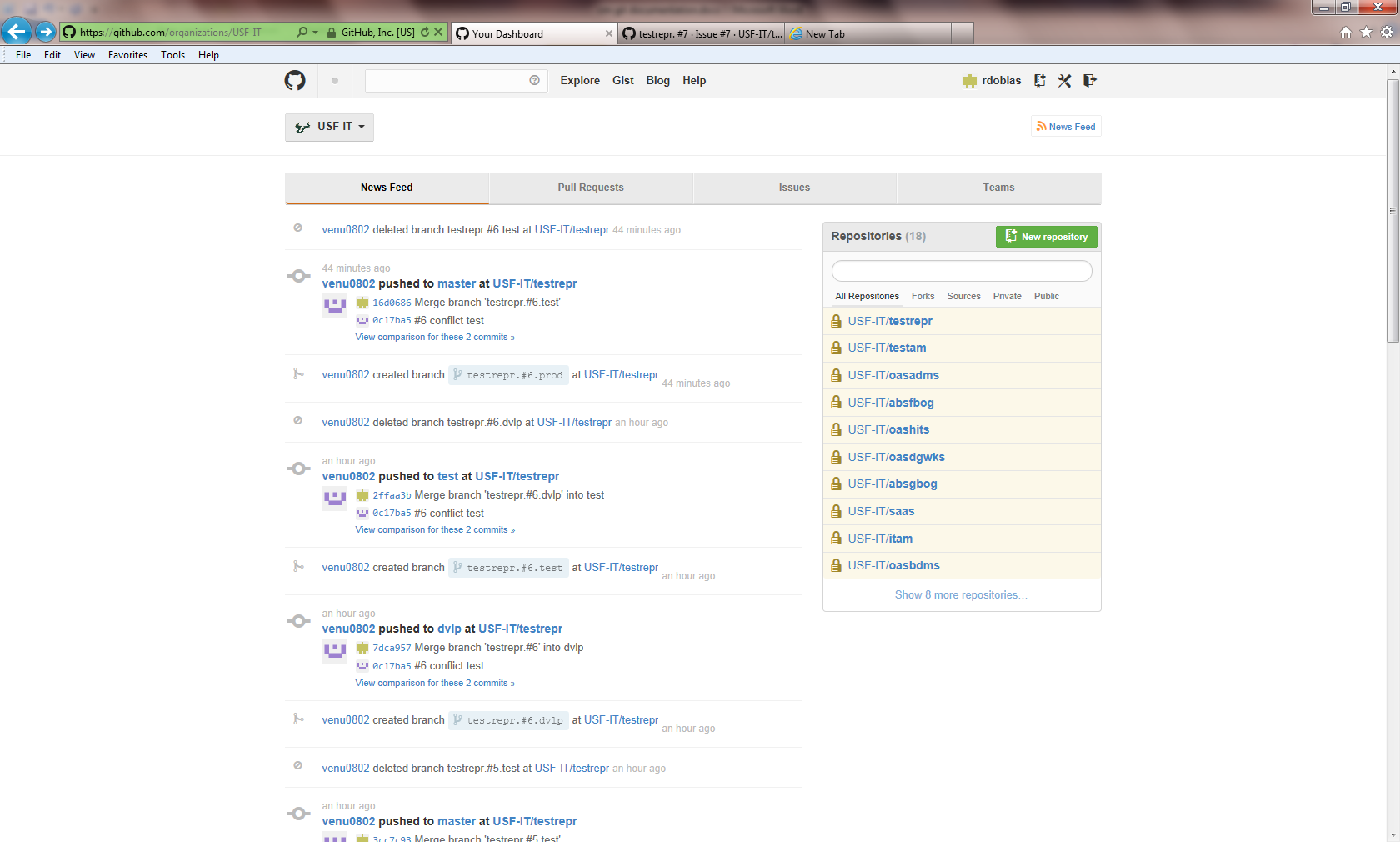
Technical Manager:                                        Anne Marie

Go to GITHUB <https://github.com>

1. Make sure it is under USF-IT

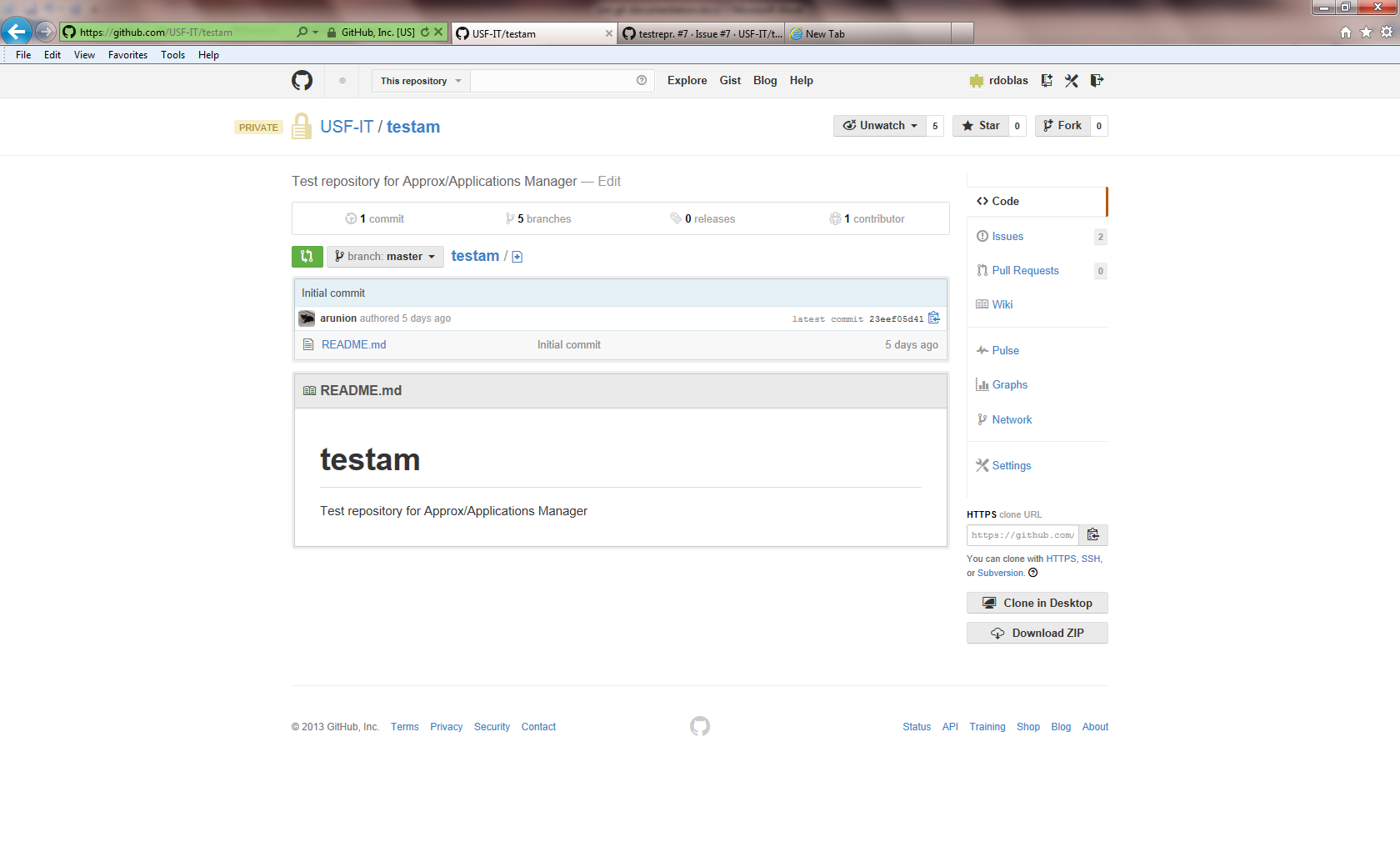
USF-IT

1. Click on the repository you would like to create the issue under:



Repositories

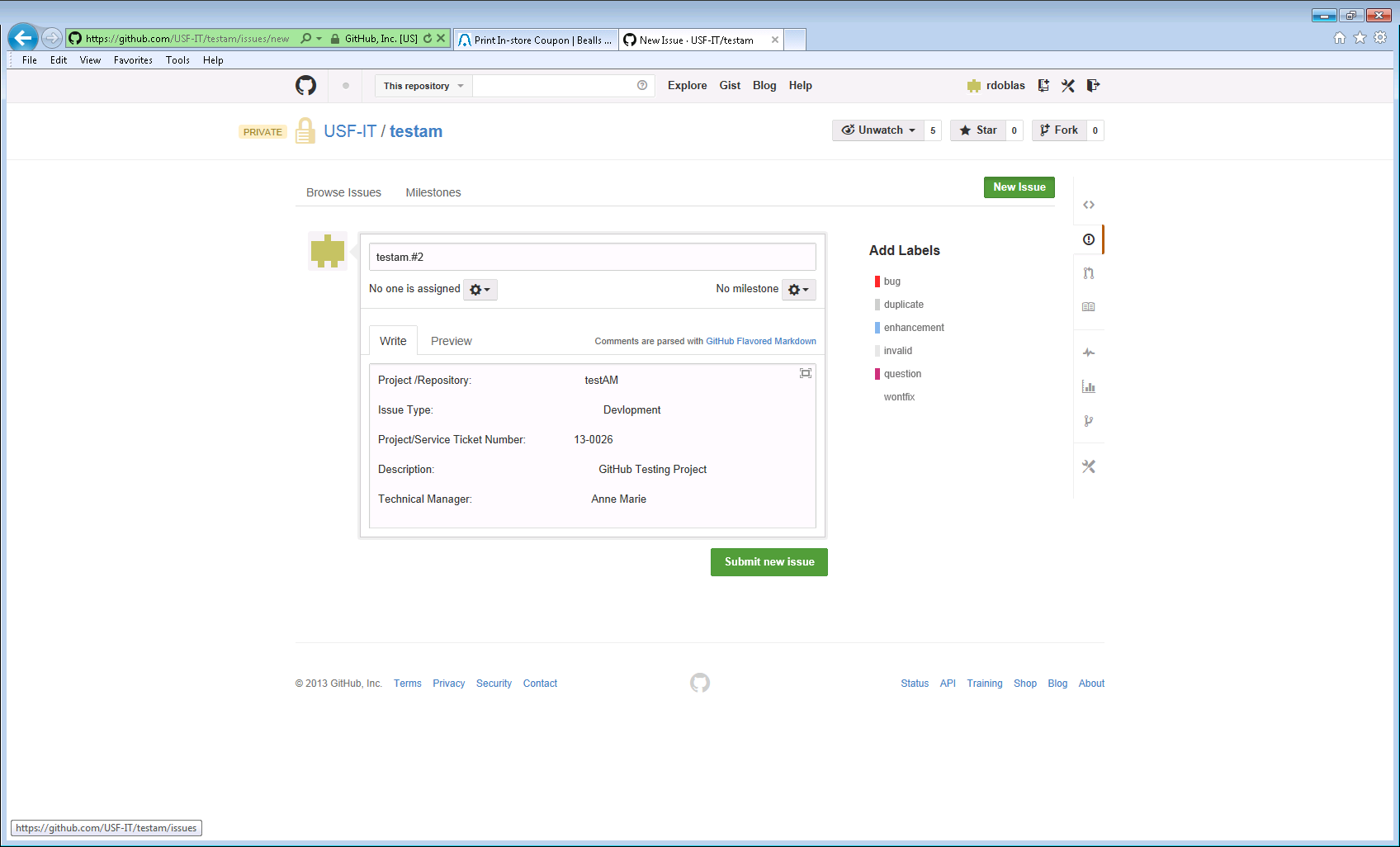
1. Click on issues and make sure you see how many already exists as the issue has to be created in numerical order by CM. In the example below 2 issues already exist under **testam** so the next issue should be testam.#3. This will vary per repository. There is not automatic issue number generation so CM has to keep up with the count.



Issues

Issue Count

1. Click on New Issue (green button) and enter the issue name as follows <reposit>:#<issuenumber> Example: testrepr.#1or testam.#2 add the information sent in the issue creation e-amil and click on Submit New Issue (green button) below.

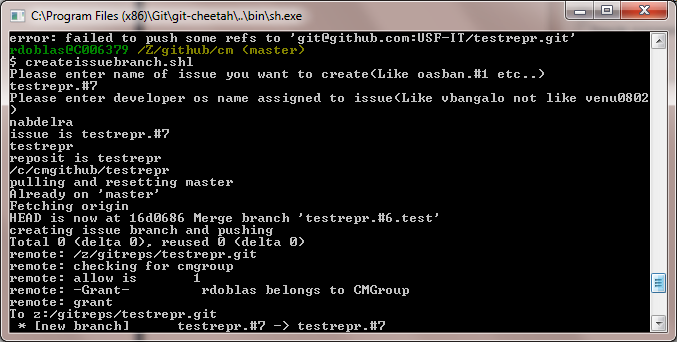


Submit New Issue

Assign Developer to the Issue

New Issue

2. After the issue has been created open z/github/cm in gitbash, and run **createissuebranch.shl** issue.#? In this case it is number 7 (testrepr.#7). Also need to provide the developer name (github user id such as **nabdela**). A list with all the developers will be needed by CM to use at the time the branch is created. See current user list below:

****

**GitHub user IDs:**

**AIS users:**

Sunitha – sunithaarun

Sylvia Pagan– spagan

Julie – julies

Christine - harperc

**ABS users:**

Charlie Holland **–** csholland

Tracy Weber-tracyweber

**OASIS users:**

Revati Vogeti – rvogeti

Eva – wanzheng

Melodi Roberts – mroberts

Nada - nabdelra

Sharon Day – sday

Helen Ngo – hhngo

Anne Marie Runion – arunion

Dennis Griffith - dgriffit

Annapoorna Masina**-** annamasina

**UFA users:**

Steve Runion– srunion

Doug Anderson– douglasanderson

Kristian Walden- kwalden

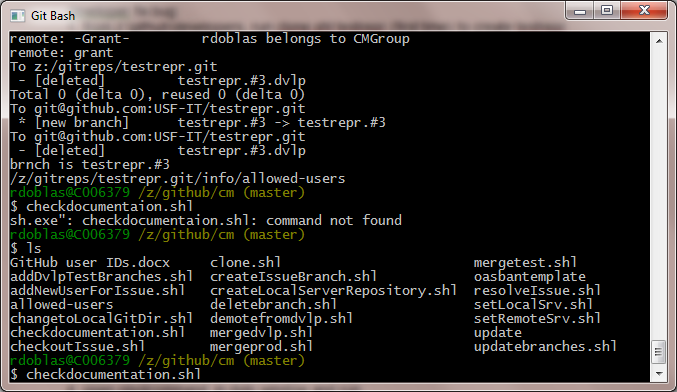
**Important:** Each CM support person needs to clone a repository if a new one is created. For example Raquel created all the repositories that we have now so no need for her to clone them however, Sharon and Helen will need to. Also, when the CM support process changes hands they should run updatebranches.shl to make sure they have the latest changes. Also developers like Sharon or Helen they may also perform a CM backup role need to check and add themselves as CM to the project they are going to be processing the issue under, otherwise it will created a problem if this step is not done prior to starting the CM process.

\***Enter the 2 commands here when checking and adding to the CM role**

**Note:** *At every state whether it is in DVLP, PPRD or PROD, first, open /z/github/cm in git bash, and run checkdocumentation.shl to check instructions (especially for conflict resolved comment) and also run updatebranches.shl as the first two steps.*

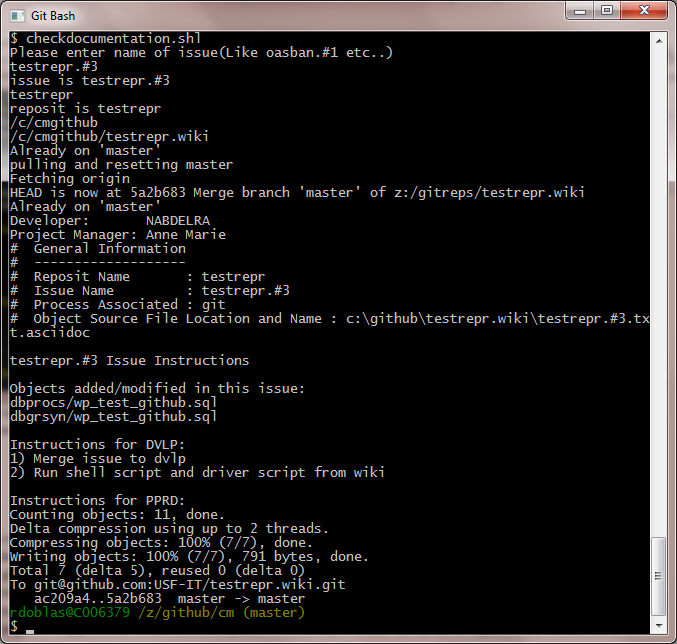
**Processing an issue in DVLP**

Open /z/github/cm, run checkdocumentation.shl. Enter issuename when prompted.

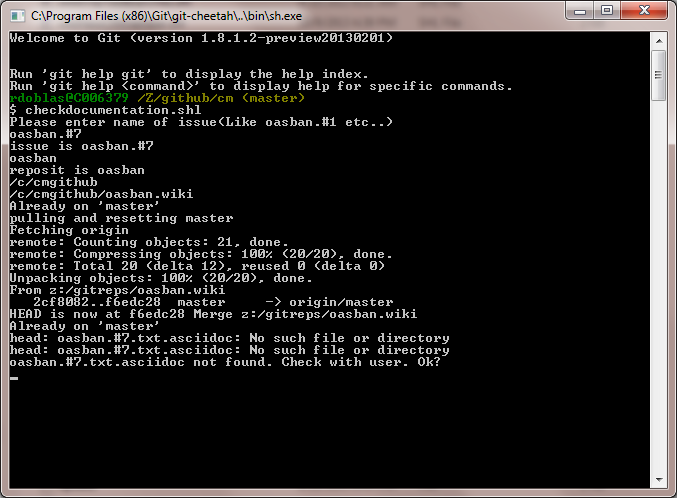


1. Enter issue name as follows:

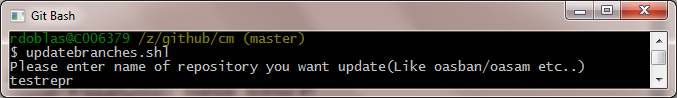
<reposit>:#<issuenumber> example: **testrepr.#3**



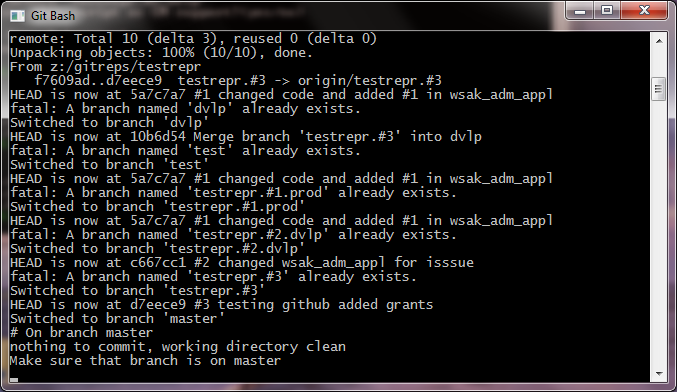
Sometimes we may get an error at the check documentation step so we need to check with developer to determine the problem may be.



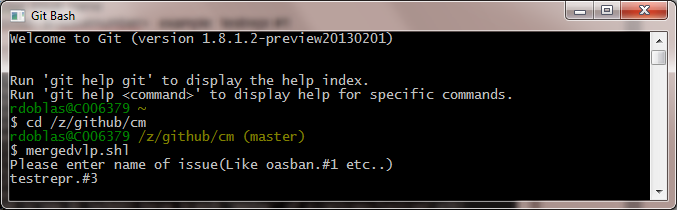
1. Run **updatebranches.shl**, and enter repository name (Example: **testrepr**)



1. Check if branch is on master and just press enter if branch is on master.

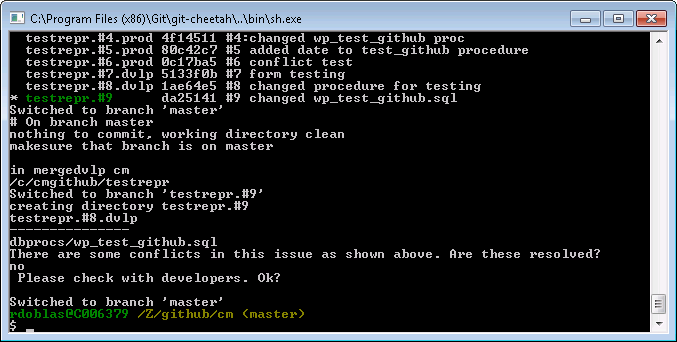


1. Run **mergedvlp.shl**, enter issuename (like testrepr.#3) when prompted

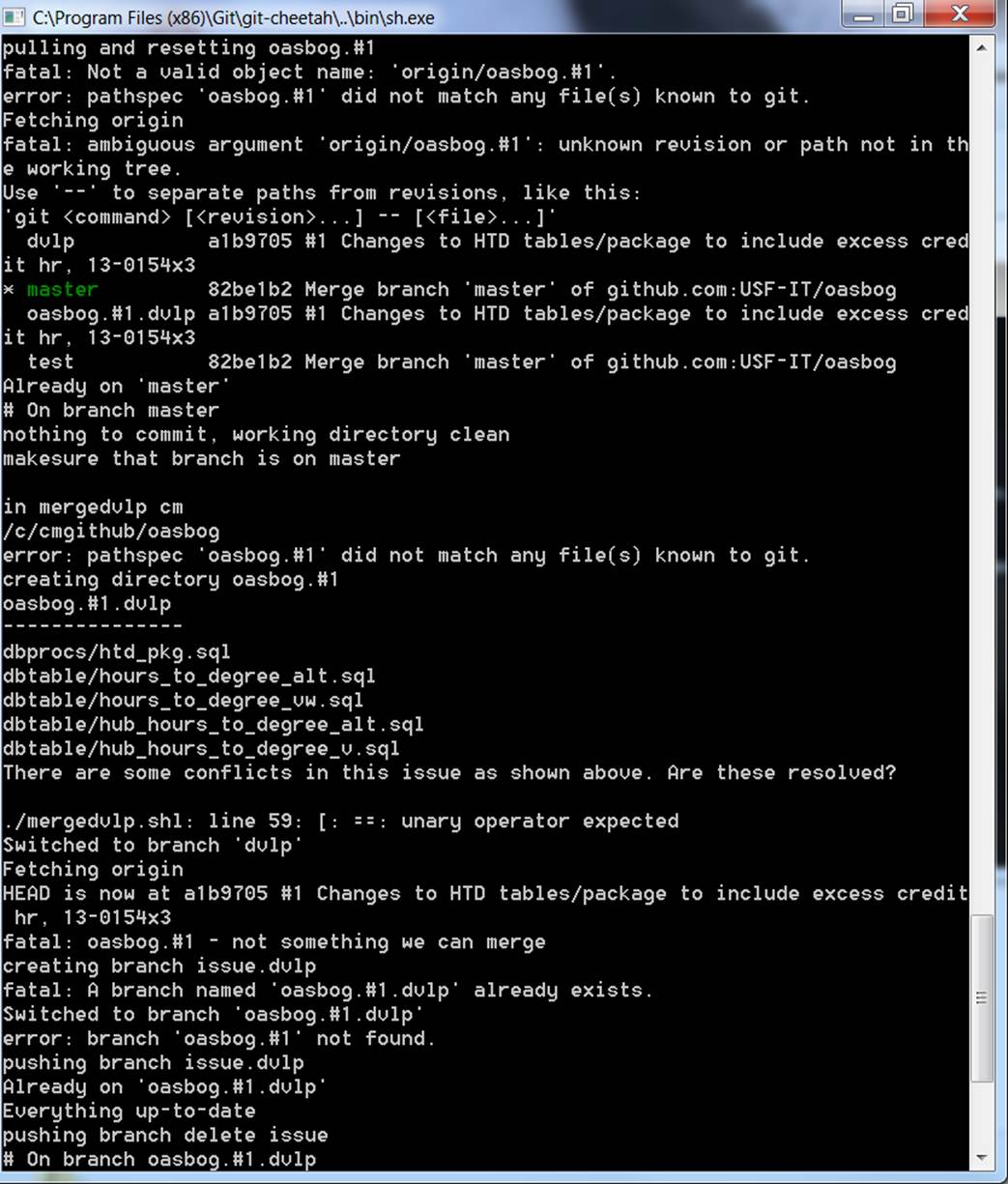


Note: *At this point during the merge process a conflict can occur when two issues have the same modified object and get merged to the same branch. A message as the one below will appear to alert CM of a possible conflict. At this point need to notify developer of the conflict.*

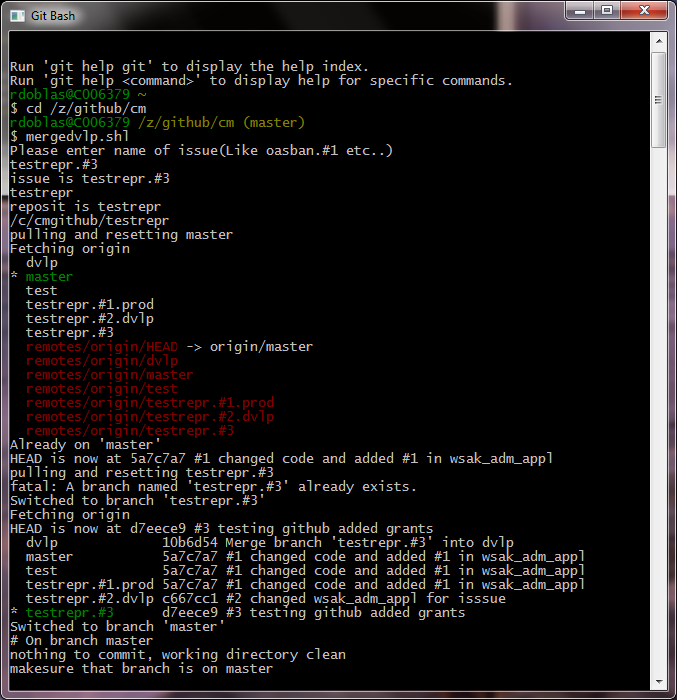
Example 1: Message of possible conflict



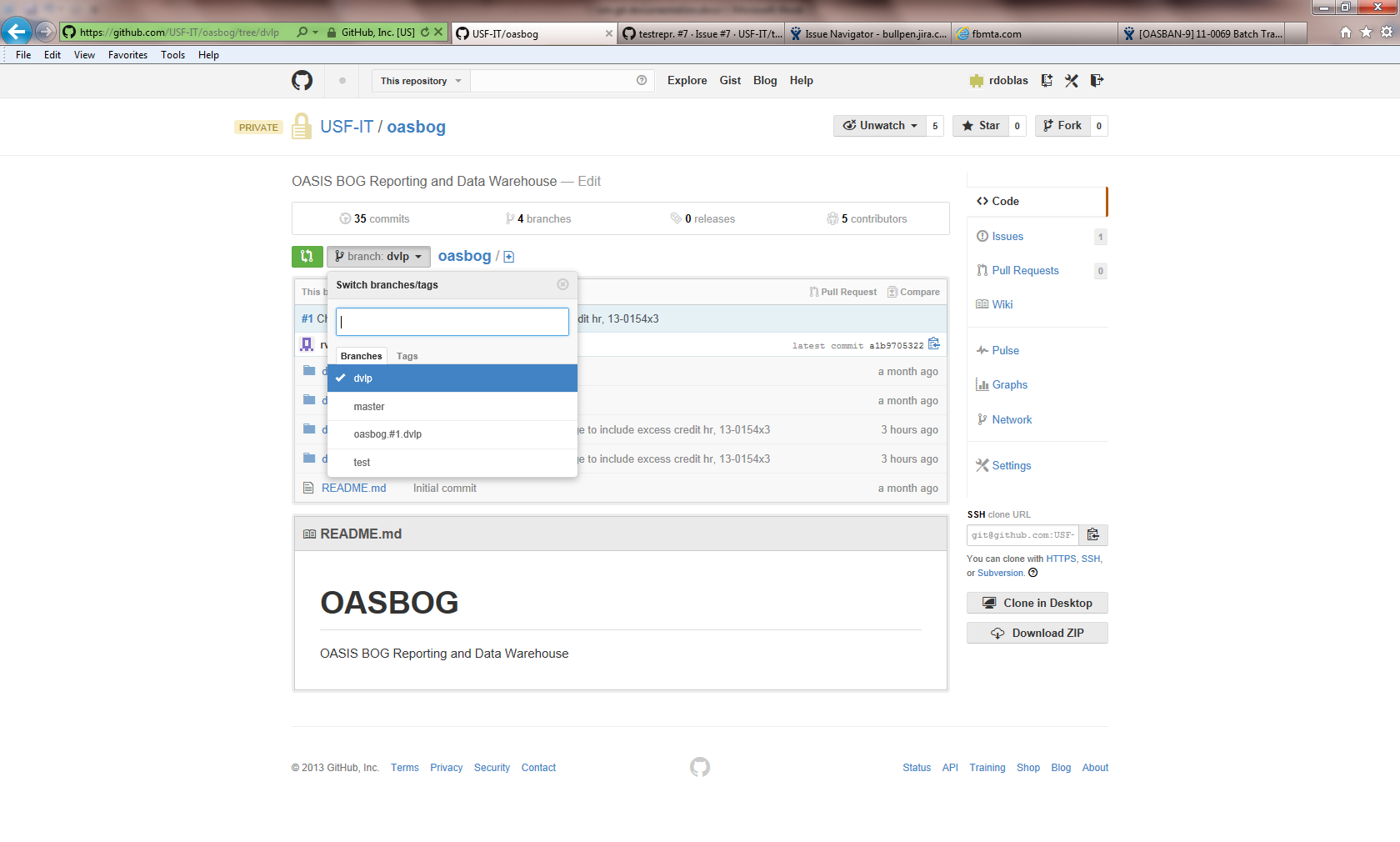
Example 2: Message of possible conflict



1. If there is no conflict proceed by clicking Enter when branch is on master



1. After the merge process is completed go to the issue project and verify that there is an issuename.dvlp with the changes and the dvlp branch also shows the changes.

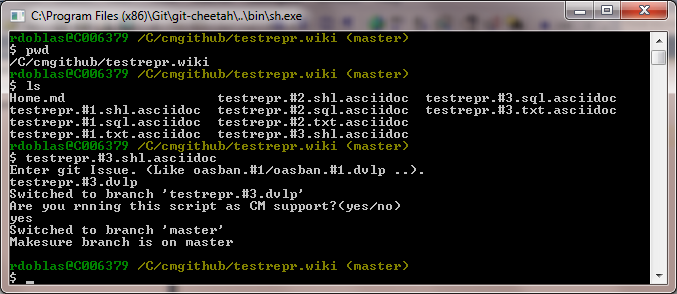


oasbog.#1.dvlp

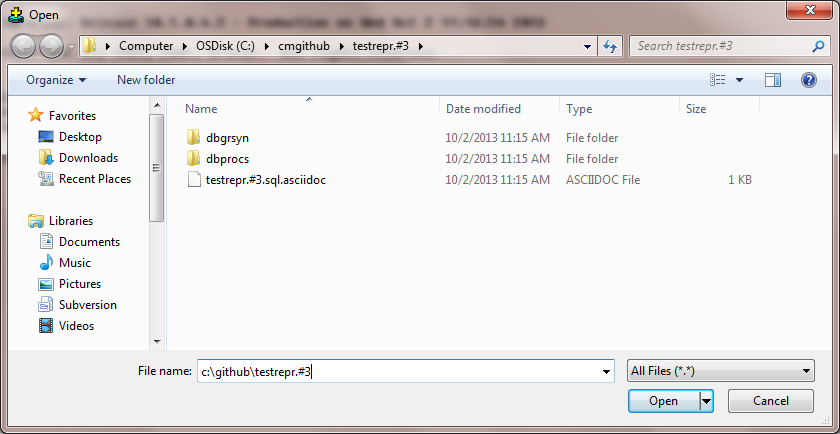
1. Open /c/cmgithub/<repository>.wiki in gitbash. (Example /c/cmgithub/testrepr.wiki)



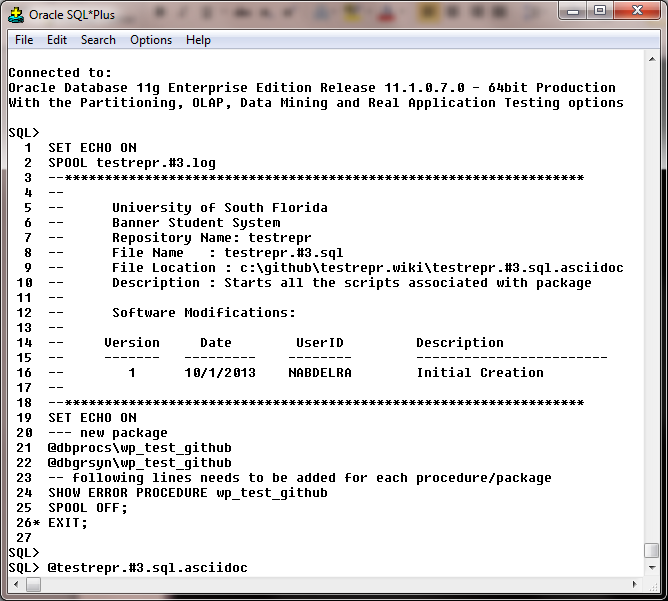
1. Run issuename.shl.asciidoc (Example testrepr.#3.shl.asciidoc) and make sure to enter issuename.dvlp. Example (testrepr.#3.dvlp) when prompted.



1. This creates a folder with the same issue name containing a file with the issue name and the extension like testrepr.#3.sql if it is a database issue or for example testam.#2.txt if it is and Application Manager issue. The folder will also contain subfolders the objects to be process as seeing below.

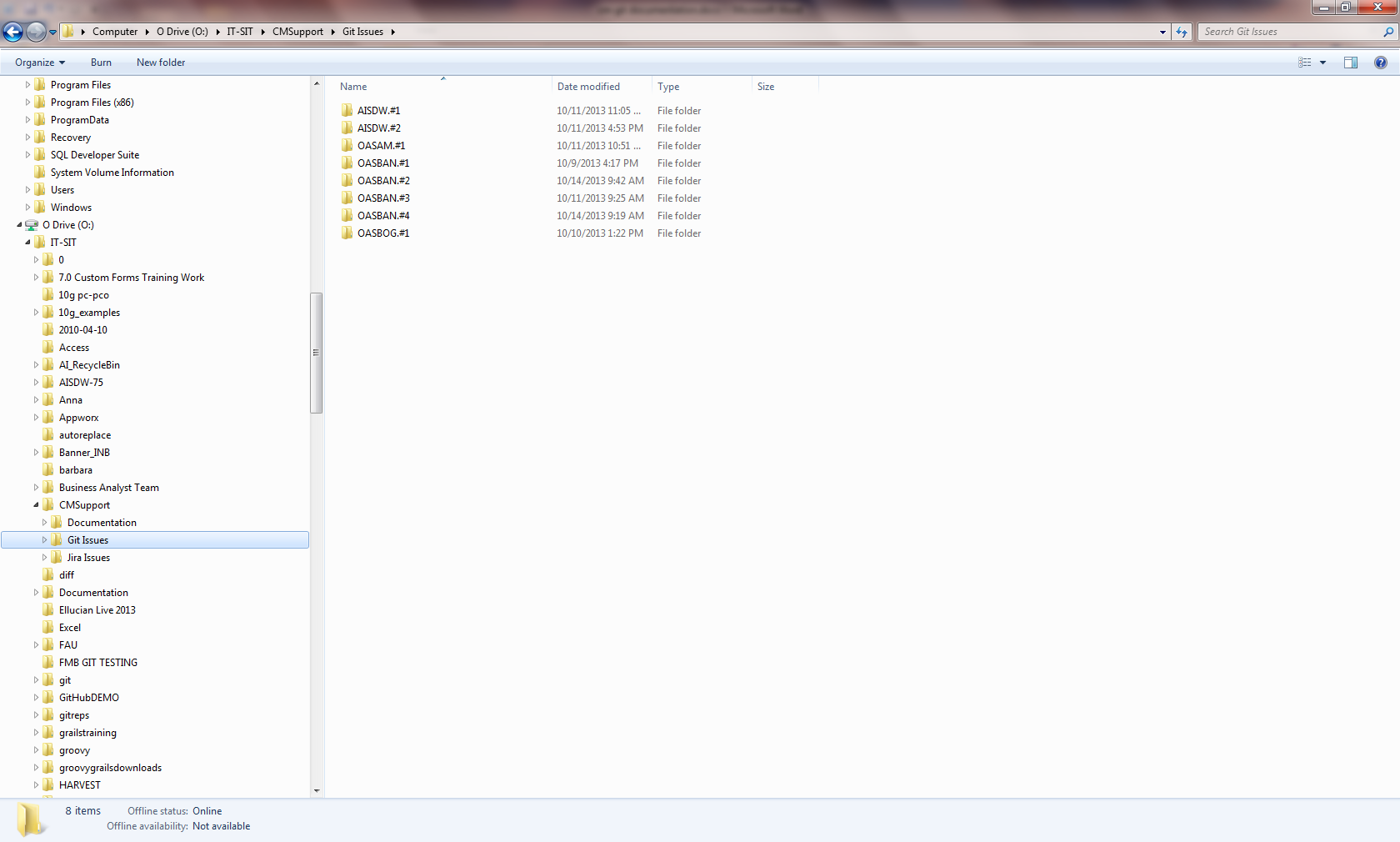


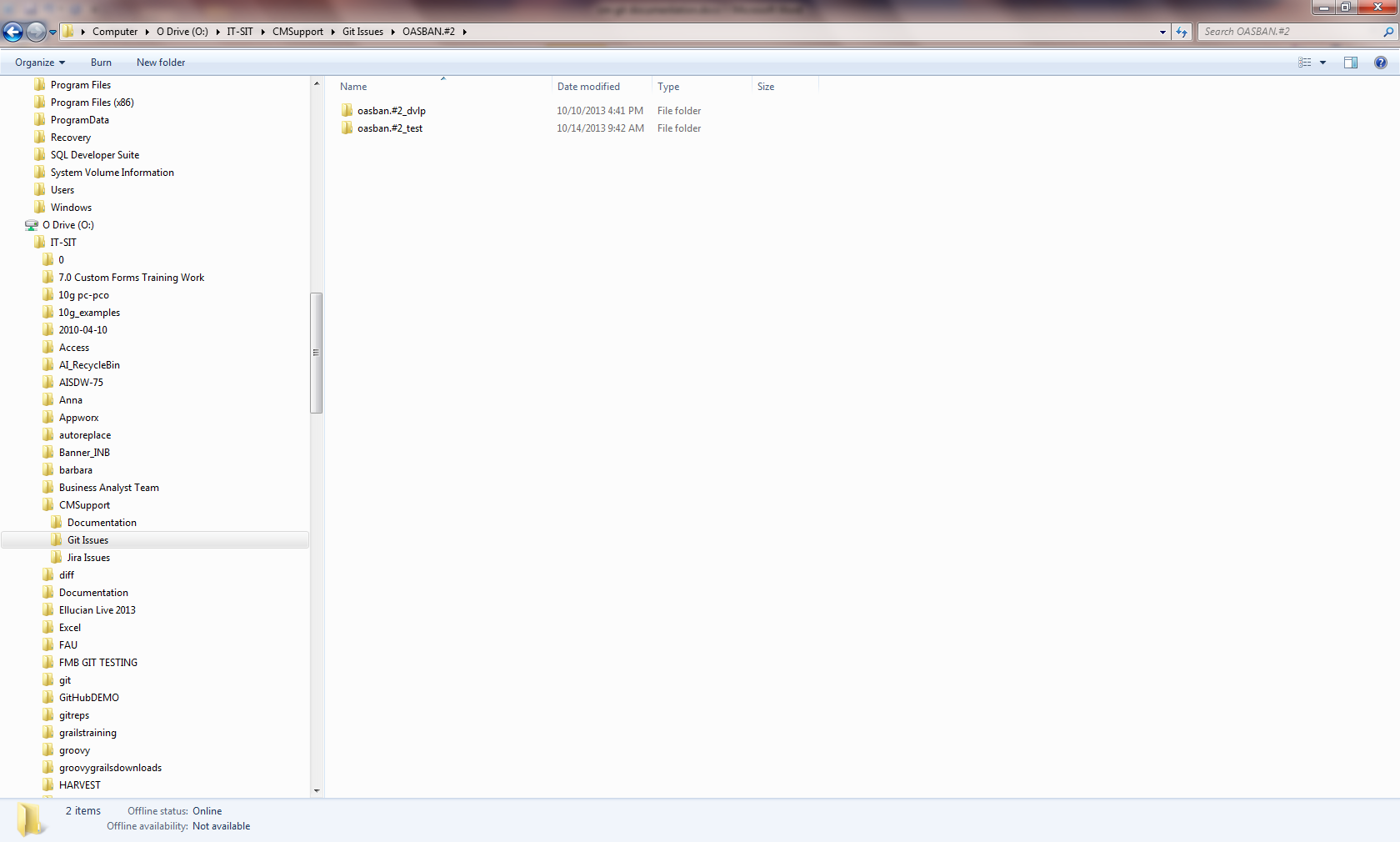
1. Open sqlcommand in dvlp and open c:\cmgithub\<<issue>> folder and then load <<issue.sql.asciidoc>> file. Example (c:\cmgithub\testrepr.#3 and load testrepr.#3.sql.asciidoc). Also see notes for applying a database issue using the dbcommit.shl process.



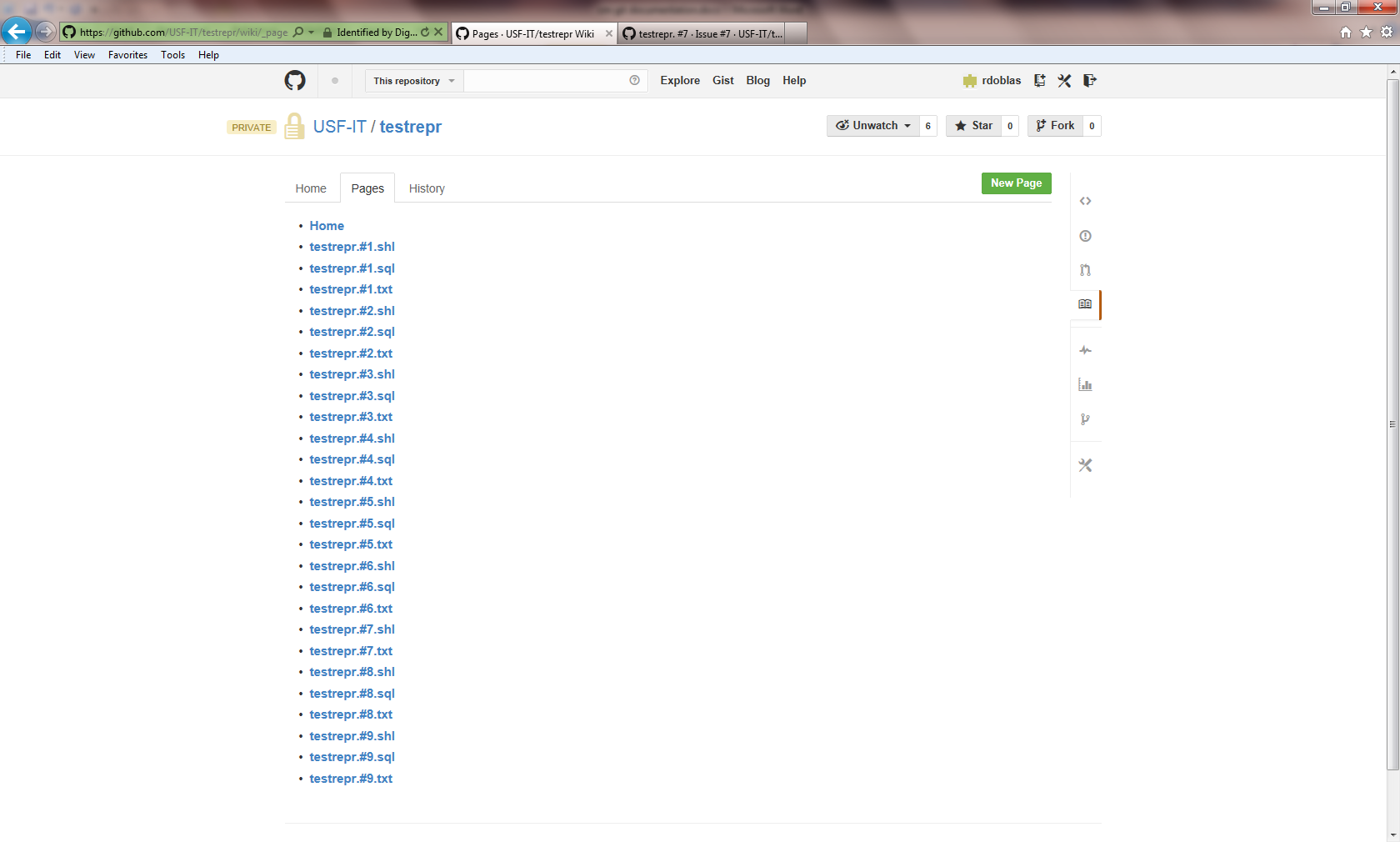
\*(We may not need to apply the issue in DVLP just to do the merge to DVLP. Still to be decided)

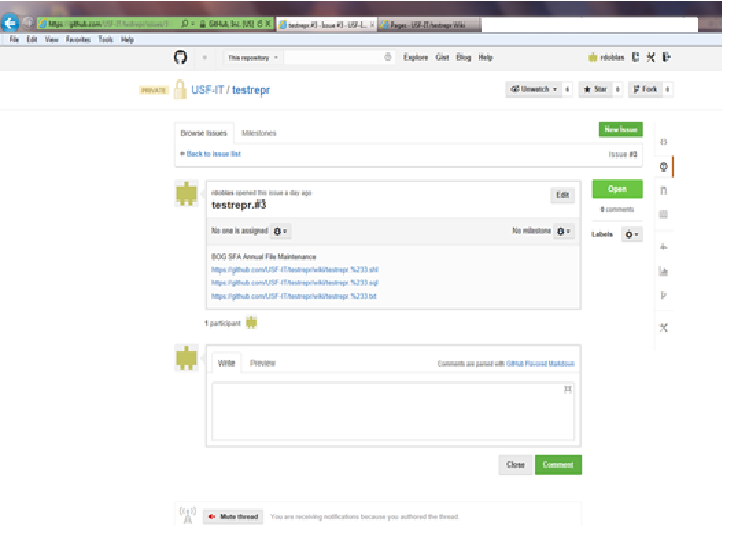
1. Send log file from c:\github\<<issuefolder>>\<<issuename.log>> to developer.
2. Move Issue folder from c:\github to O:\IT-SIT\CMSupport\GitIssues folder after renaming folder name to <issue\_dvlp>. (oasban.#2\_dvlp folder)





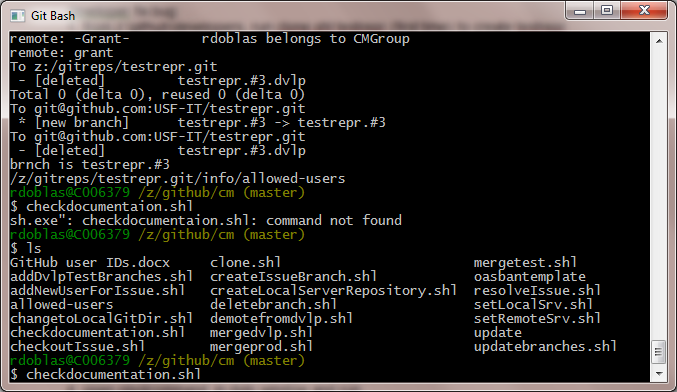
1. Update github with the issue documents like issue.shl, issue.txt and issue.sql. Open 2 tabs of the issue you are going to update the documentation for. With one tab go to wiki (book icon) then go to pages and copy links/copy shortcut of each of the files corresponding to the issue then paste on the notes area. Then click on update issue for the links to appear. This is done only one time.





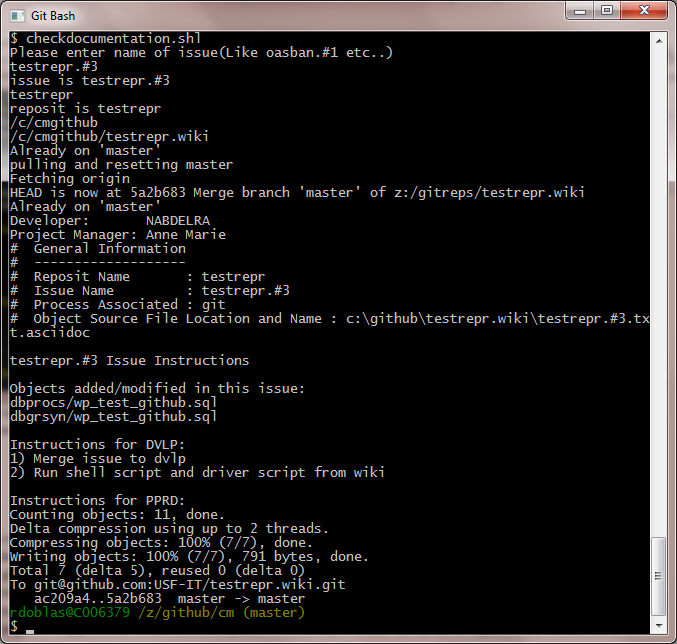
**Processing an issue in Test:**

1. Open /z/github/cm, run checkdocumentation.shl. Enter issuname when prompted.

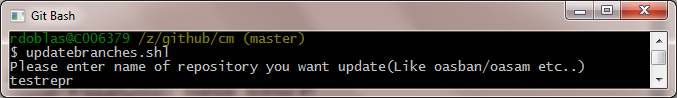


1. Enter issue name

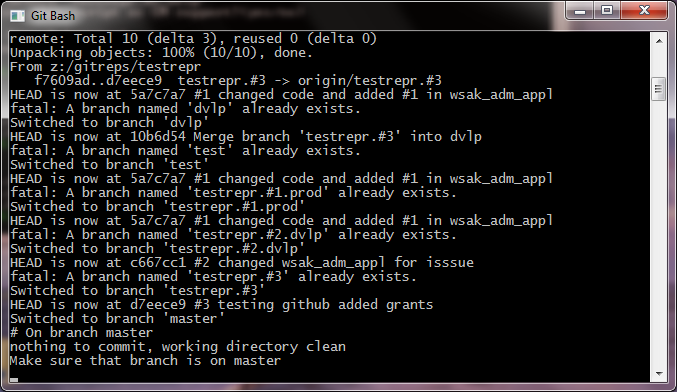
<reposit>:#<issuenumber> example: testrepr.#3



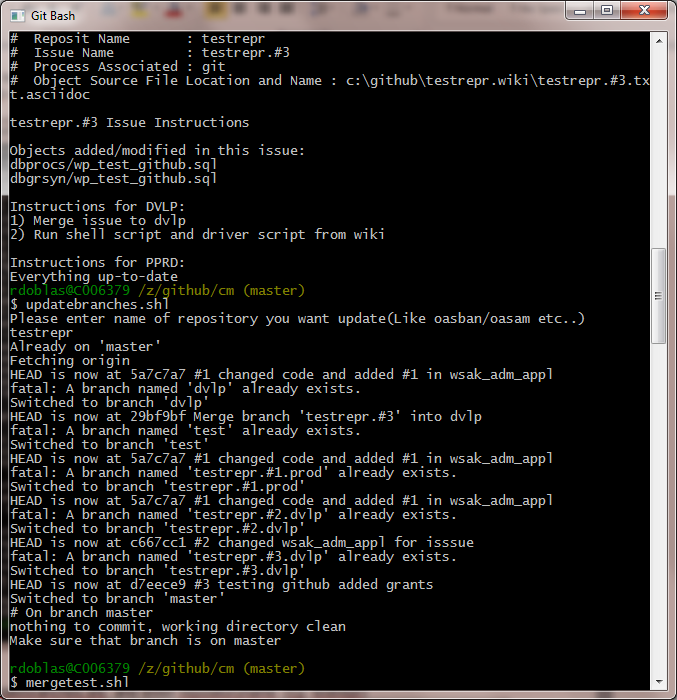
1. Run updatebranches.shl, and enter repositoryname (Eg: testrepr)

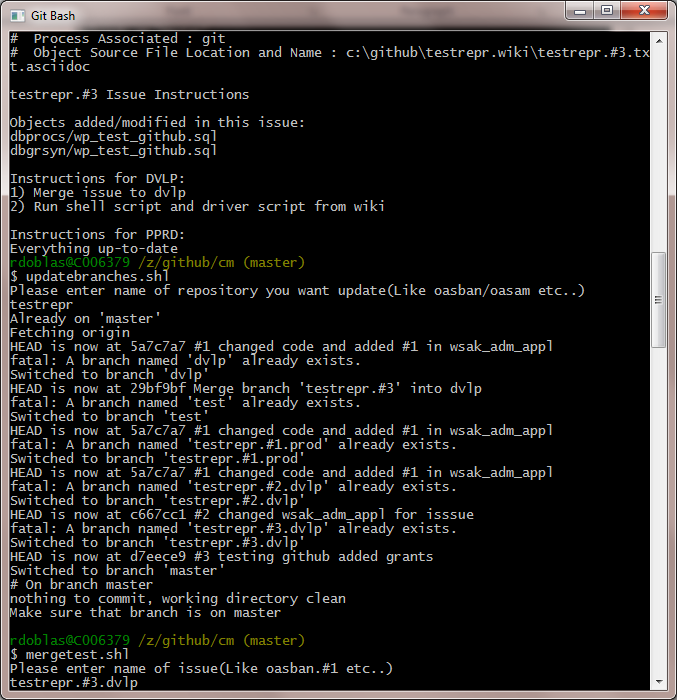


1. Check if branch is on master and just press enter if branch is on master.

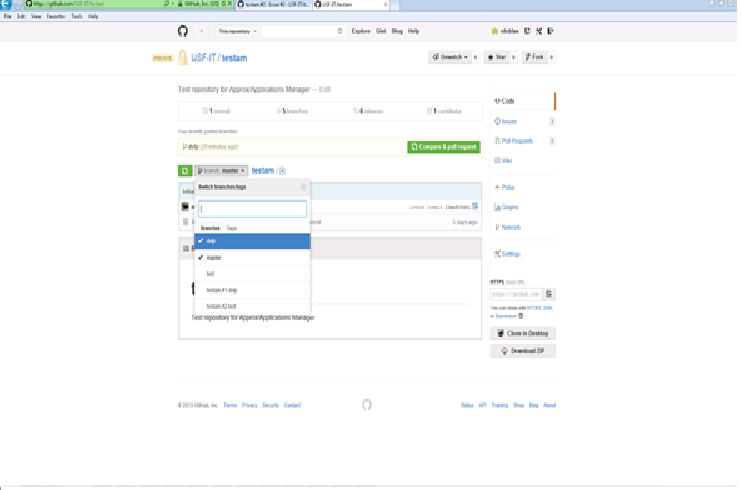


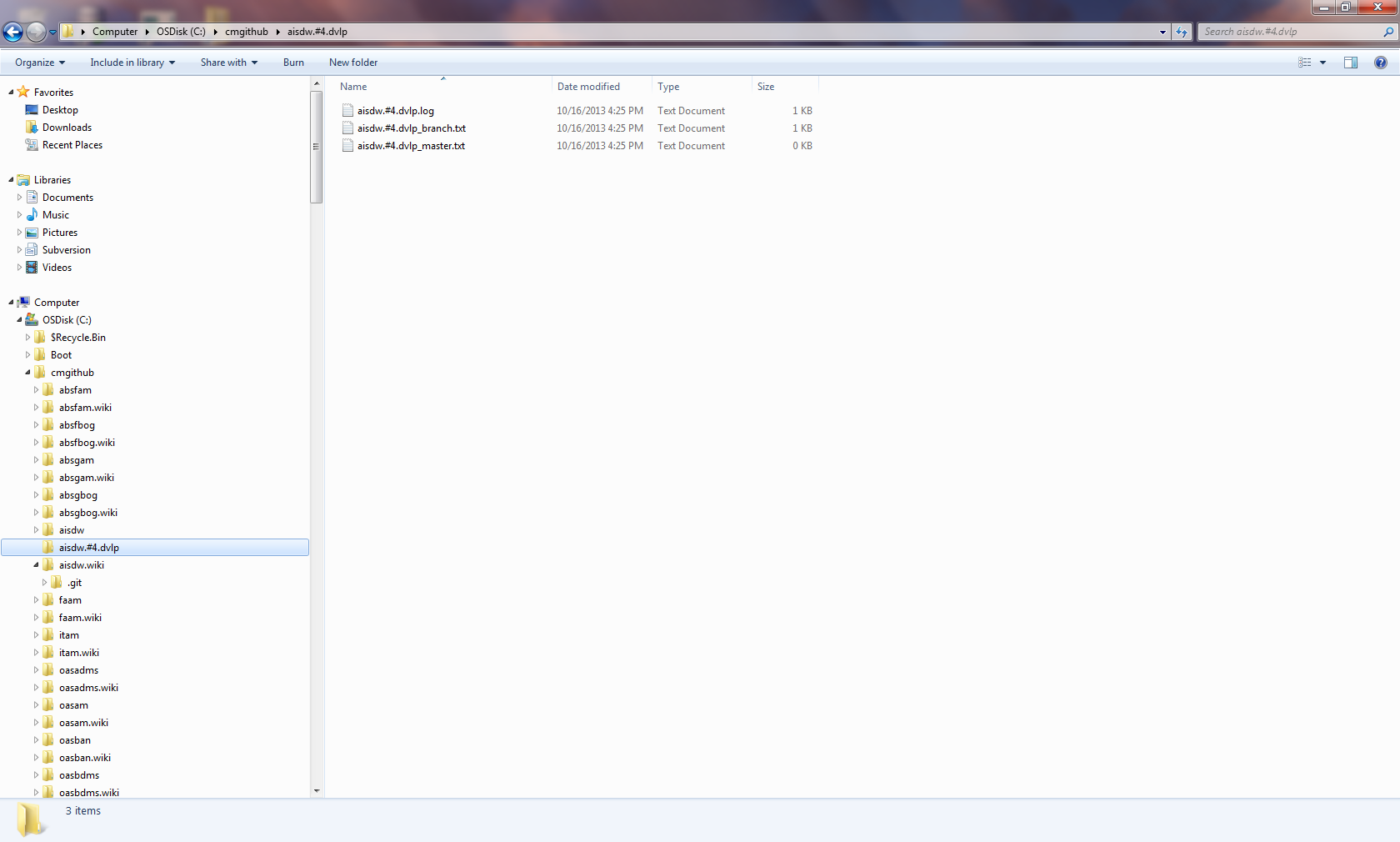
1. Run mergetest.shl, enter issuename.**dvlp** (like testrepr.#3.**dvlp**) when prompted





1. Test github for branch rename as issue.#.dvlp then renamed to issue.#.test



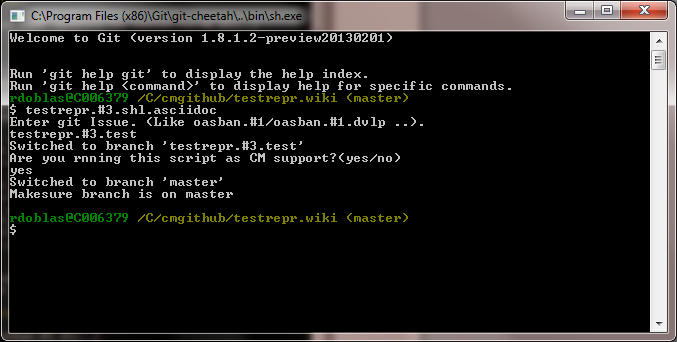


At the time the issue is merged in test it will create a folder <<issuname>>.dvlp see above aisdw.#4.dvlp. This folder can be deleted and a new <<issuname>> folder will be created as step 7 below is processed.

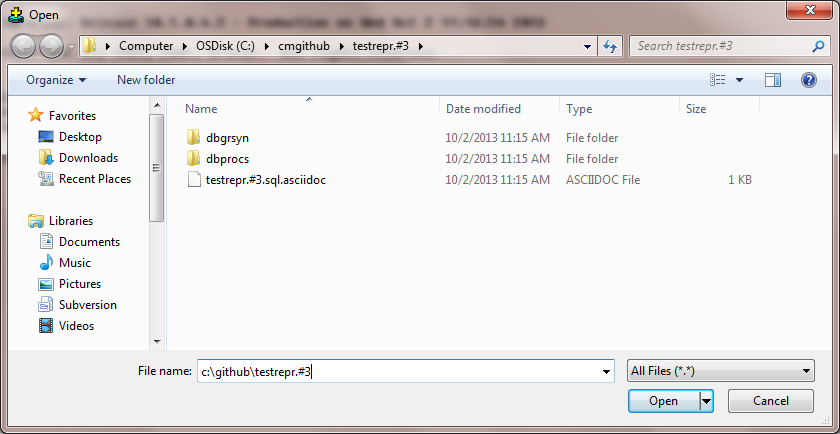
1. Open /c/cmgithub/<repository>.wiki in gitbash. (Example /c/cmgithub/testrepr.wiki)

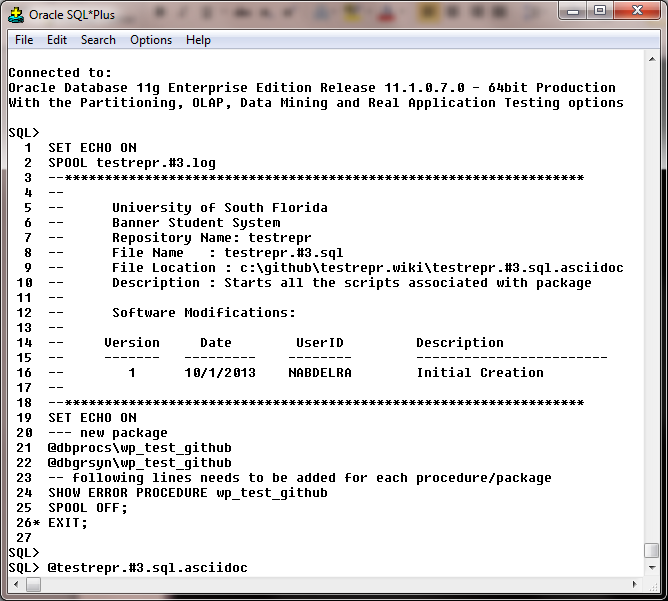


1. Run issuename.shl.asciidoc (Example testrepr.#3.shl.asciidoc) and enter issuename.test. Example (testrepr.#3.test) when prompted.



1. Open sqlcommand in PPRD and open c:\cmgithub\<<issue>> folder and load <<issue.sql.asciidoc>> file. Example (c:\cmgithub\testrepr.#3 and load testrepr.#3.sql.asciidoc) (See notes for applying a database issue using the dbcommit.shl process).



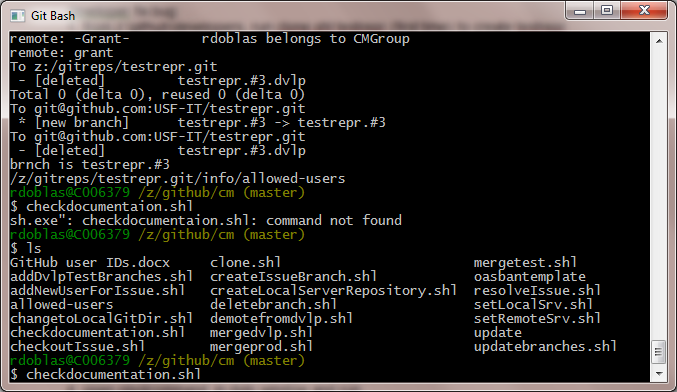


1. Send log file from c:\github\<<issuefolder>>\<<issuename.log>> to developer.

Move Issue folder from c:\github to O:\IT-SIT\CMSupport\GitIssues folder after renaming folder name to issue>.test like testrepr.#3\_test

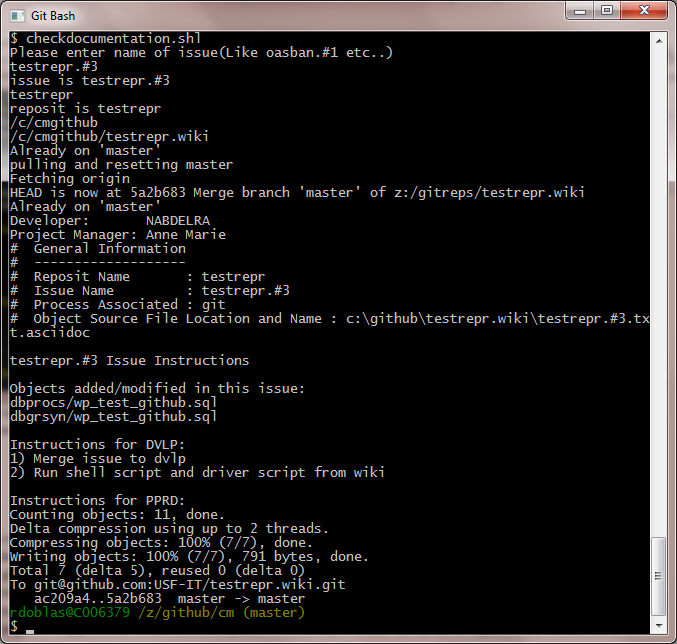
**Process and Issue in Prod:**

1. Open /z/github/cm, run checkdocumentation.shl. Enter issuname when prompted.

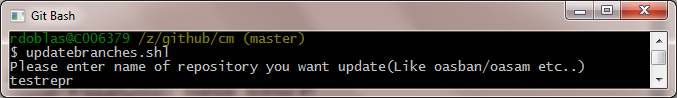


1. Enter issue name

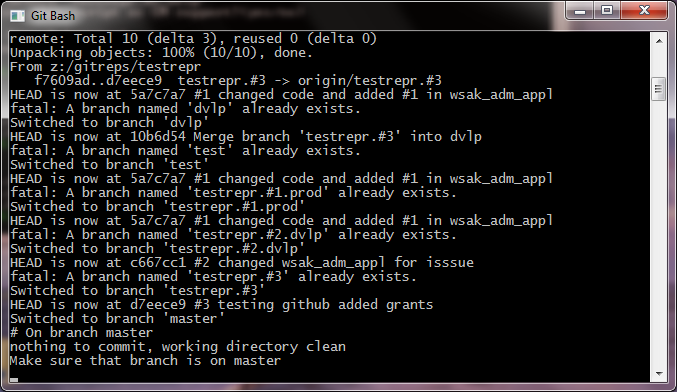
<reposit>:#<issuenumber> example: testrepr.#1



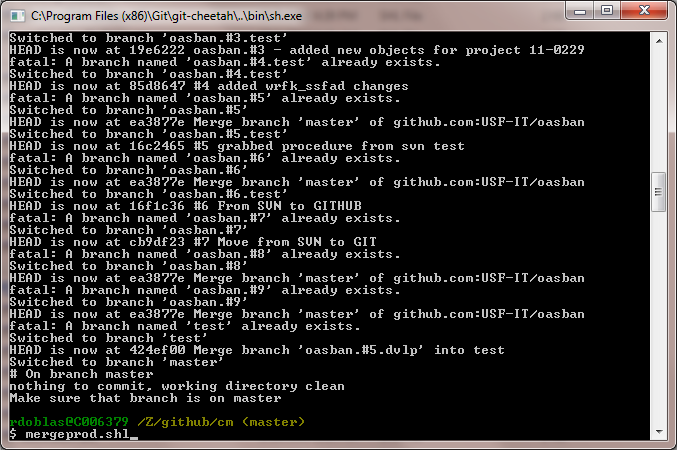
1. Run updatebranches.shl, and enter repositoryname (Eg: testrepr)

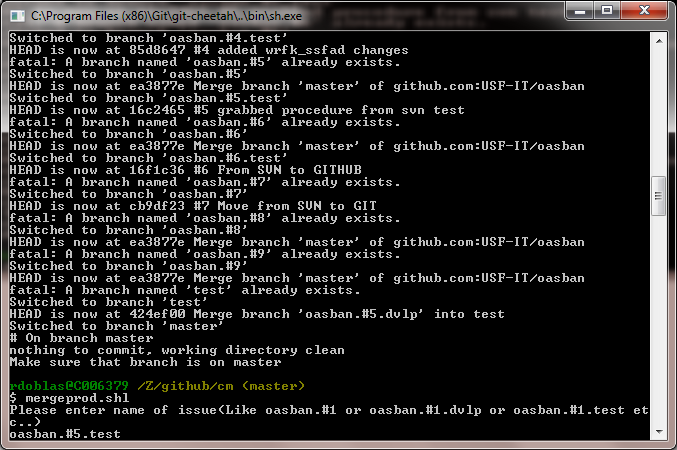


1. Check if branch is on master and just press enter if branch is on master.

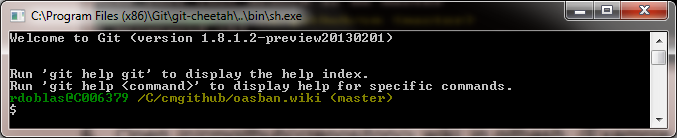


1. Run mergeprod.shl, enter issuename.test (like testrepr.#1.test) when prompted.

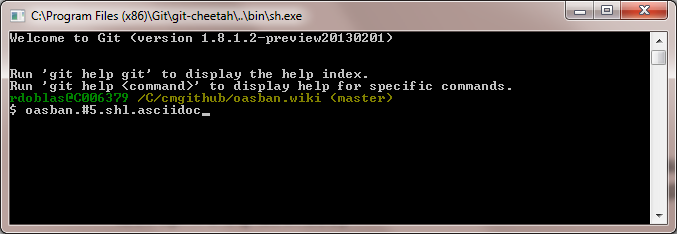


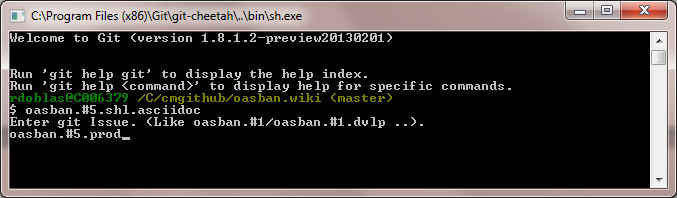


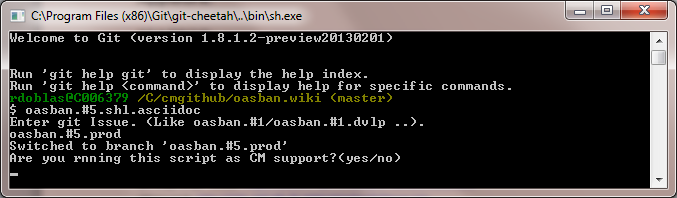
1. Open /c/cmgithub/<repository>.wiki in gitbash. (Example /c/cmgithub/oasban.wiki)



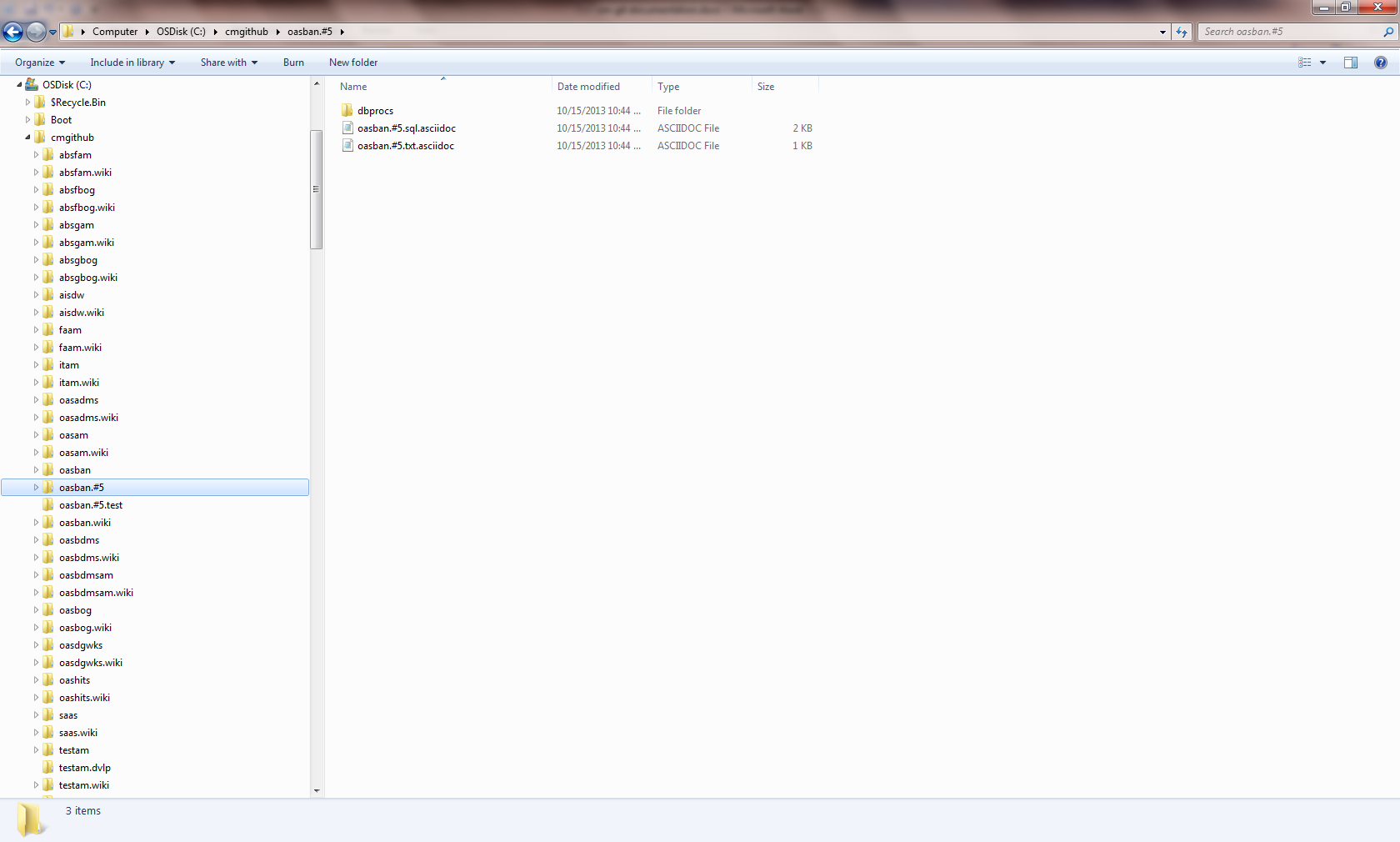
1. Run<<issuename>>.shl.asciidoc (Example testrepr.#3.shl.asciidoc) and enter issuename.prod. Example (oasban.#5.prod) when prompted.



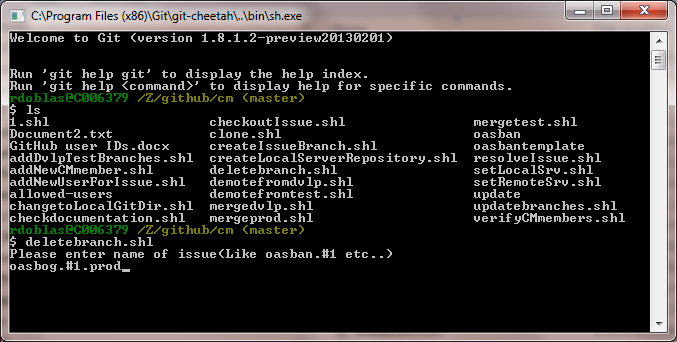




1. Create folder for DBAs by copying oasban.#5 (see example below) to z:\versioncontrol\<<repositoryname>>



1. Make sure that there is no asciidoc extension for the DBAs
2. Have DBAs apply issue in the production databases (i.e., PROD, DWHOUSE, REPO, IMGPROD, etc.)
3. Create a ServiceNow ticket for DBAs to process the issue
4. Send log file to the developer after DBAs are done
5. Copy folder to O:\IT-SIT\CMSupport\GitIssues folder after renaming folder name to <issue\_prod> (oasban.#5\_prod folder)
6. Run deletebranch.shl and enter the issuename.prod you want to delete the branch for

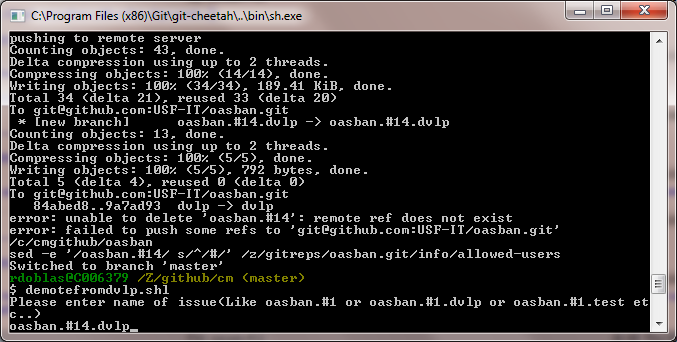


1. Apply to upgrade instances for OASBAN issues to DVLPUPG and PPRDUPG. Verify with developer for other upgrade instances if it is not an OASBAN issue

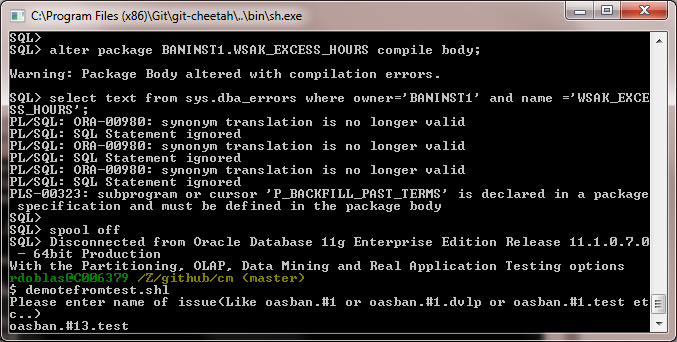
**Demote from Develpoment or Test:**

If the developer needs to make changes to the code but not to the driver script, text file or shell scipt, CMsupport needs to be notified to demote the issue ieither from Development or from Test. The two comands will be used depending if the issue has been merged to dvlp or test.

1. Demote from Development:



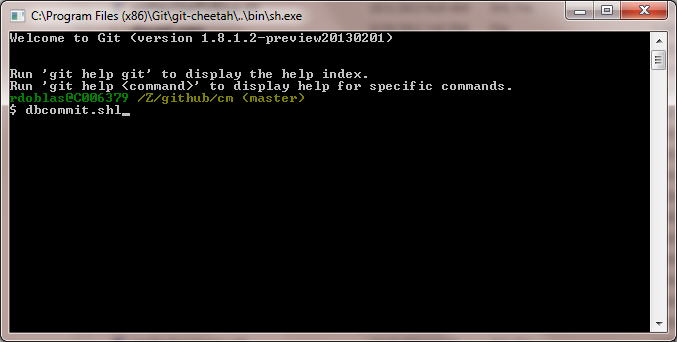
1. Demote from Test:



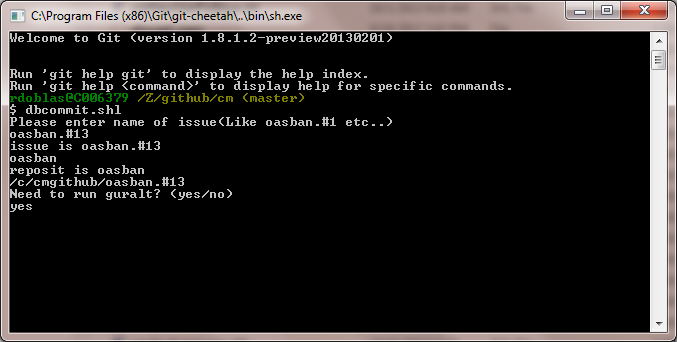
**Applying a Database Issue Using the dbcommit.shl Process:**

In addition to applying a database issue in the way described in previous pages where you go to SQL+ and open and run the driver script, there is there is a command line shell script that will allow CM Support to apply the issue and run the invalid objects script before and after the issue is applied for (OASBAN issues only)

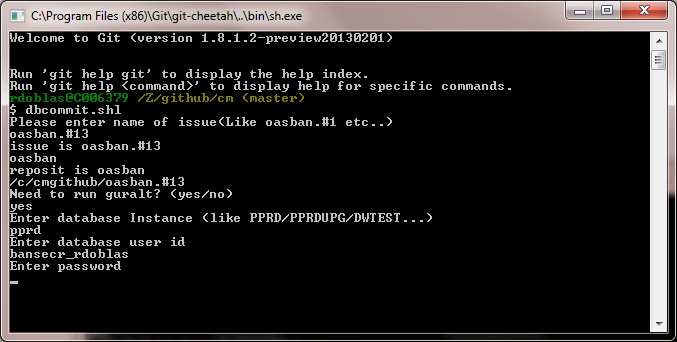
1. Go to /z/github/cm and run dbcommit.shl. Follow the prompts:



1. Choose yes for OASBAN issues only



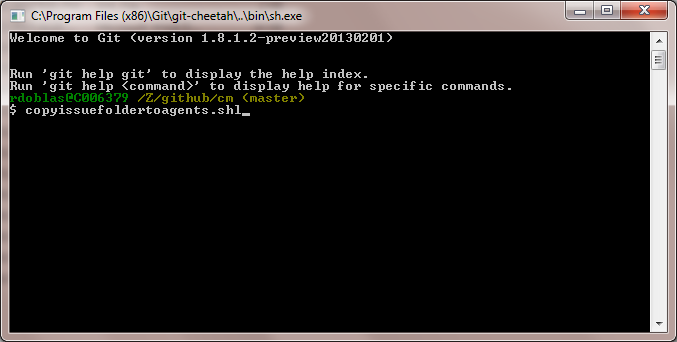
1. Enter the ID and password corresponding to the type of issue being applied



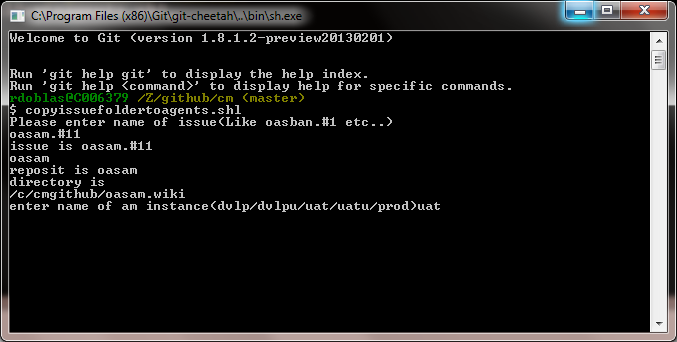
1. A log file will be generated as with the other process and will be sent to the developer to review

**Processing an Appworx Issue using the automated method:**

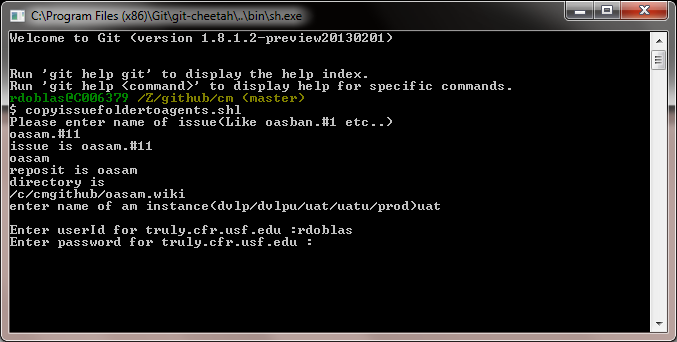
1. Go to /z/github/cm and run copyissuefoldertoagents.shl and follow the prompts. This replaces the cmsftp process in Jira:



1. Enter the issue number
2. Enter instance you are processing the issue under



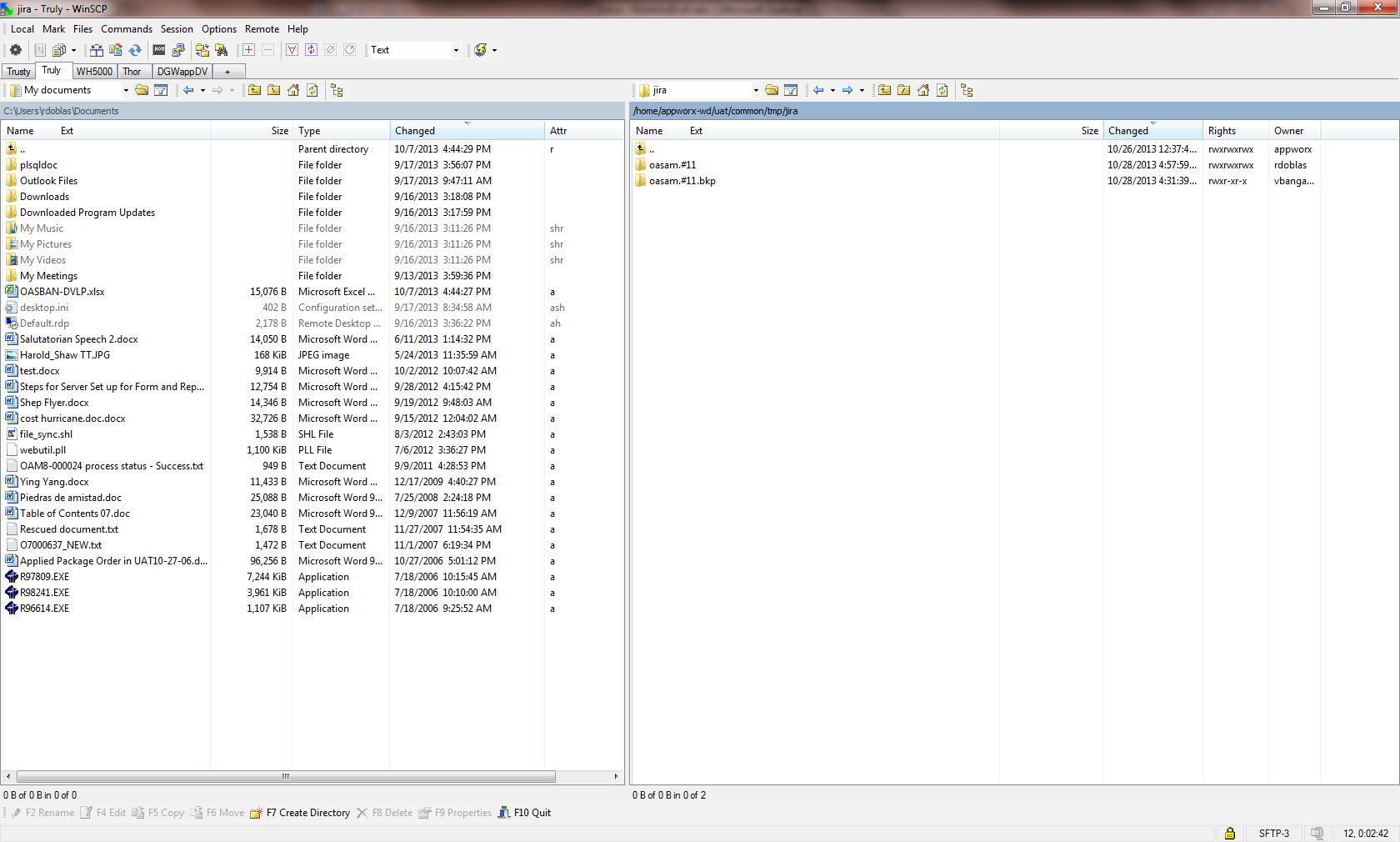
1. Enter ID and password for the Unix server you are logging into.



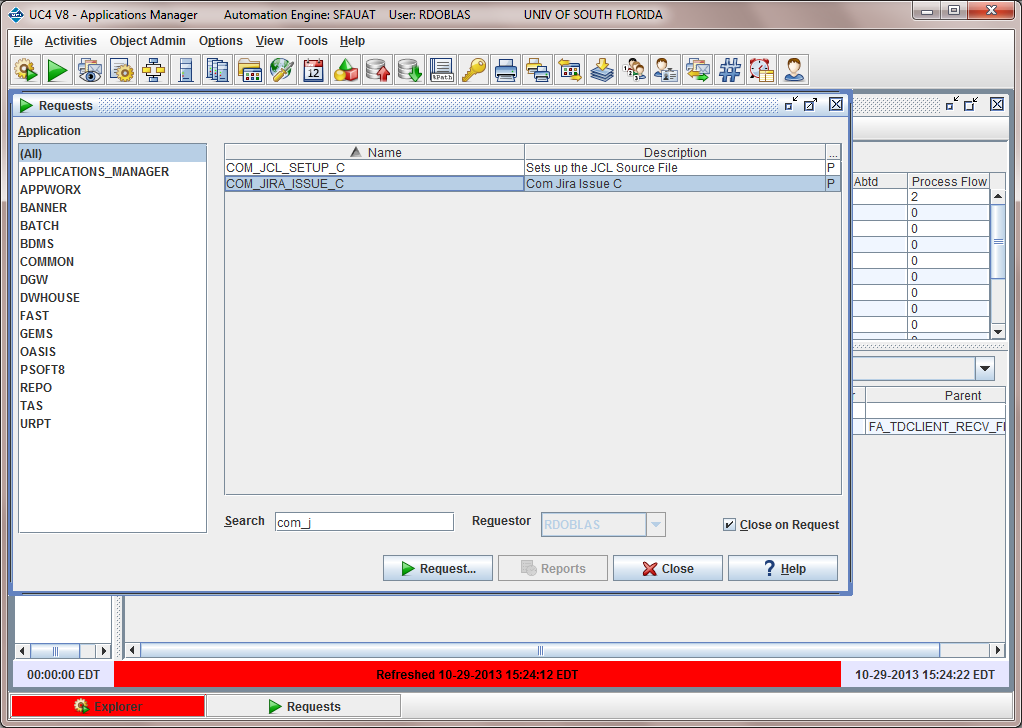
\*Ask Venu when/why to run ssh^unixserver. Example: **ssh truly.cfr.usf.edu**

**Add info here**

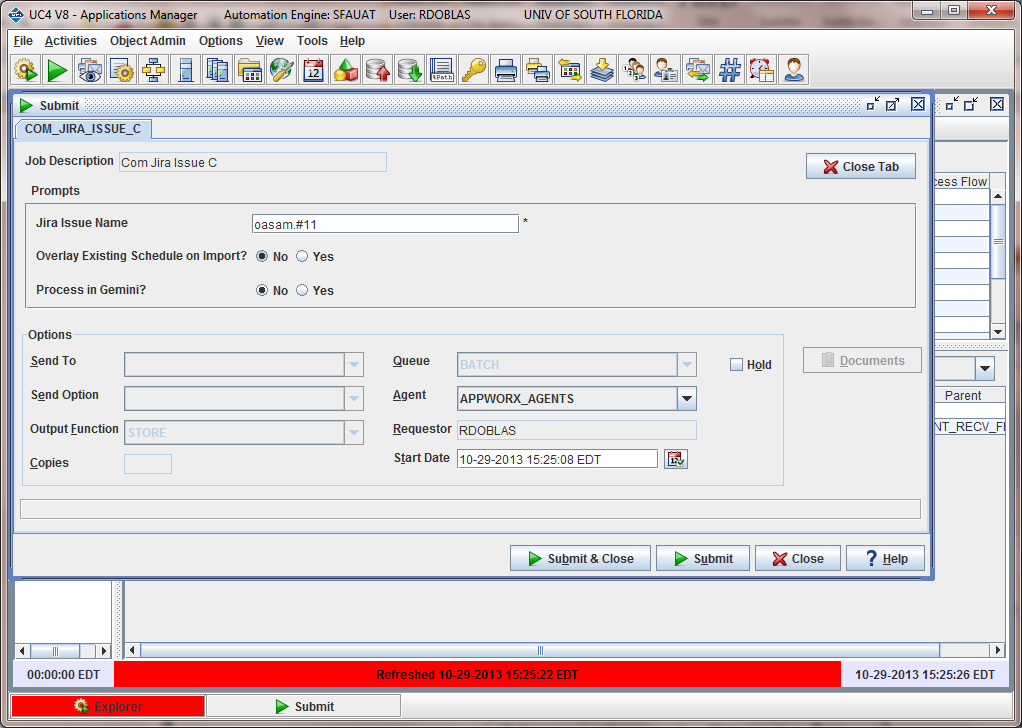
1. This script will create a temporary issue folder in Trusty, Truly or Thor depending on the instance you choose. After this step runs successfully go to the Appworx instance you are processing the issue and request COM\_JIRA\_ISSUE\_C.



1. Click on Request



1. Enter issue number you want to process. Now still says Jira issue, but it is Git issue number like oasam.#11



1. An automatic email like with Jira will be generated indicating if the process was successful or not and listing all the objects that have been imported into AppWorx (.exp) and those that have been FTPed to Unix (i.e., .sql, .rdf, .shl). Always review the text file to make sure all the objects got processed.

-------------------------------------------

From: Appworx Test[<SMTP:APPWORX@TRULY.CFR.USF.EDU>]

Sent: Monday, October 28, 2013 4:59:53 PM

To: IT CMSupport

Subject: oasam.#11 process status - Success Auto forwarded by a Rule

JIRA Status....

Starting to copy files....

ascii translation of sql...

adm\_vzbdms\_swrvdoc\_cleanup\_m.sql copied to /home/appworx-wd/uat/oasis/sql

Starting to import....

timewithsec ....... FINISHED

Imported successfully....

adm\_vzbdms\_swrvdoc\_cleanup\_m ....... FINISHED

Imported successfully....

adm\_vz\_docs\_to\_bdms\_c ....... FINISHED

Imported successfully....

it\_process\_docs\_to\_bdms\_c ....... FINISHED

Imported successfully....

adm\_bisk\_docs\_to\_bdms\_c ....... FINISHED

Imported successfully....