Getting Started with the WIT-Projects Stream in RTC on CS/Next

November 11, 2011
Revised January 17, 2013

by Bob Wittrock

The BAMS department has several software projects relating to WIT: WIT, SCE, WIT-J, etc. The source code for these projects is stored in an RTC stream on CS/Next. The stream is called “WIT-Projects”. This document explains what you need to do to get started working the WIT-Projects stream. (This document assumes some familiarity with the structure of the source code files for the WIT-related projects.)

Please note that the URLs in this document are set up as clickable links, but are also given as full URLs, in case you need to cut and paste.

Joining the BAMS Applications Community

First, if you haven't already done so for other reasons, you will need to join the BAMS Applications Community on CS/Next. To do this, go to the following site and follow its instructions:

`http://w3.ibm.com/connections/wikis/home/wiki/Wd9eaeceee6b2_47ef_b68f_3c745dab8e92/page/How%20to%20join%20BAMS%20Applications%20Community%20Source%20Next?lang=en_US`

Documentation

In the process of joining the community, you'll receive an e-mail with various links to documentation about RTC, so I'll assume that you've had a look at that. A couple helpful links are:

Comparing Concepts Between Subversion, CVS, and RTC:

`http://w3.ibm.com/connections/wikis/home/wiki/Wc4ff7836119b_49d8_9d84_b711c0610d16/page/Comparing%20concepts%20between%20Subversion%2C%20CVS%2C%20and%20RTC`

The RTC Glossary:

`http://publib.boulder.ibm.com/infocenter/rtc/v2r0m0/topic/com.ibm.team.concert.doc/topics/glossary.html`

Note that “Jazz Source Control” and “Jazz SCM” are older names for RTC.

Installing an RTC Client

Next, you'll need to install some version of the RTC client on your machine. RTC is available on Linux and on Windows. It does not appear to be available on AIX 6.1. There are two kinds of RTC client that I have experience with:

The Command Line Interface: an executable to be invoked from the command line with arguments.
The Eclipse Client: a plug-in to Eclipse

I think there's also a web client, but I don't really know anything about it.

Which client to use? I have successfully used the Command Line Interface on Linux, the Eclipse Client on Linux, and the Eclipse Client on Windows. I mostly use the Eclipse Client (on Linux) now : It's more comprehensive, I find it easier to use, and you have much more visibility to what you're doing. Note that using the Eclipse Client doesn't require using Eclipse as your development environment: It doesn't put the files in an Eclipse workspace (unless you want it to); it puts them wherever you want. So you can just use Eclipse for RTC and then use Xemacs, Make, etc. Also, I know of one significant action (deleting a repository workspace) that simply can't be done with the Command Line Interface. (There are work-arounds; more on this later.) Some advantages of the Command Line Interface are that you don't have to learn how to use Eclipse, you don't have to have Eclipse running to use it, and you can put the commands into a script.

To get the RTC Command Line Interface, go to the following CS/Next web page and follow its instructions:

```
https://csnext.ibm.com/ram/search/_rlvid.jsp.faces?_rap=!
assetDetails&_rvip=/search/index.jsp&guid=%7B29F26A2C-AF16-18A6-CA27-
804ACAD63A12%7D&v=2.0.0.2%20iFix%205&submission=false
```

The user name and password that you will need will be your IBM Intranet user name and PW. Note that the package that you get includes both an executable for Linux and one for Windows. User documentation for it can be found at:

```
http://publib.boulder.ibm.com/infocenter/rtc/v2r0m0/topic/com.ibm.team.scm.doc/topi
cs/c_scm_cli.html
```

To get the RTC Eclipse Client, go to the following CS/Next web page and follow its instructions:

```
https://csnext.ibm.com/ram/search/_rlvid.jsp.faces?_rap=!
assetDetails&_rvip=/search/index.jsp&guid=F122FAB6-0BE1-738C-9225-
FFA68840E264&v=2.0.0.2%20iFix6&submission=false
```

Being Java based, the Eclipse Client works on both Linux and Windows. User documentation for it can be found at:

```
http://publib.boulder.ibm.com/infocenter/rtc/v2r0m0/topic/com.ibm.team.scm.doc/topi
cs/c_scm_eclipse.html
```

The Eclipse Client seems to require Eclipse 3.5 at least. I'm using it with Eclipse 3.6. I did have some trouble installing the Eclipse Client on Linux, but Yingjie found the solution and documented it here:

```
http://w3.ibm.com/connections/wikis/home?
lang=en_US#/wiki/Wd9eaeceee6b2_47ef_b68f_3c745dab8e92/page/Problems%20encountered
%20during%20RTC%20installation
```

Using the RTC Command Line Interface

Here's a scenario of how I might use the RTC Command Line Interface on Linux to work with the WIT-Projects stream:

The Command Line Interface executable is called “scm”. To begin, I added the directory that contains the scm executable to my PATH.

Next, note that in addition to the scm executable, there's a script called “lscm” which has exactly the same subcommands and syntax. The difference is that lscm starts a daemon the first time and then runs faster on subsequent times. That's what I use. I had to alter the last line of this script to make it find the "java" executable in the right place.

There is also a a Windows batch file “lscm.bat” that does the same thing on Windows that the “lscm” script does on Linux. To use it, you may have to alter a line (that begins with “SET JAVA=”) to make it find the "java" executable in the right place.

The first command that I would use is “login”, which I would enter as follows:

```
lscm login -r https://csnext.ibm.com:8003/jazz/ -n csnext -u wittrock@us.ibm.com -c
```

This asks me for a password, and I enter my IBM intranet PW.

This command logs me into the CS/Next RTC repository that the BAMS Community uses.

- The -r argument specifies the CS/Next RTC repository that the BAMS Community uses.
- The -n argument defines a nickname for the CS/Next repository that can be used in subsequent commands.
- The -u argument is my RTC user ID, which is my IBM intranet ID.
- The -c flag tells it to cache my password encrypted on my file system, so I don't have to re-enter it in subsequent commands.

Next, I might use the “create workspace” command, as follows:

```
lscm create workspace -r csnext -s WIT-Projects WIT-Projects-RW
```

This command creates a “repository workspace” for me. This is the first of two things that I have to do in order to do the equivalent of a CVS checkout.

- The -r argument specifies the nickname for the CS/Next repository that I defined in the login command.
- The -s argument specifies the “stream” to which the repository workspace will be attached. In our case, the stream is called “WIT-Projects”, and it's the central place where the source code for WIT-related projects (WIT, SCE, etc.) is stored and updated.
- The final argument (`WIT-Projects-RW`) is the name that I am giving to the newly created repository workspace. This repository workspace is my personal copy of the WIT-Projects stream. It is stored on the server.

Unfortunately the Command Line Interface does not provide a way to delete a repository workspace when you're done with it! The only way to delete it is to use the Eclipse Client: You can use the Eclipse Client to delete a repository workspace that you created even if you created it from the Command Line Interface and even if you created it from a different machine. But no other user (not even an administrator) can delete it for you. So if you're working with the Command Line Interface and need to delete a repository workspace, I can think of three ways to deal with it:

- Install the Eclipse Client and learn how to delete a repository workspace from it.
- Find a local colleague who has the Eclipse Client and knows how to use it (in Yorktown, that's me), have

them log into the Eclipse Client as you (with you entering your PW), and have them delete it.
- Just leave it there. Somebody who was suggesting this alternative on an RTC forum wrote: “The workspaces do not take up much space on the server...”. Ugh.

Anyway, after creating the repository workspace, I would then create a directory in my file system to be used by RTC as my “local workspace”, the place in my file system where the source code for WIT-related projects will be stored. I might enter:

```
mkdir /home/rjw/main/wit/wit-projects
```

This directory plays the role of “.../u/wit/rjw” in our traditional directory structure. You should be able to use any name and put it wherever you want.

Next, I would CD to this directory:

```
cd /home/rjw/main/wit/wit-projects
```

Next, I would use the “load” command, as follows:

```
lscm load -r csnext WIT-Projects-RW WIT-Projects-Component/config
```

This command (the way I've specified it) makes the current directory into a local workspace and then loads a selected portion of a repository workspace into it. This is the second of the two things that I have to do in order to do the equivalent of a CVS checkout.

- The -r argument specifies the nickname for the CS/Next repository that I defined in the login command.
- The `WIT-Projects-RW` argument specifies the repository workspace to be loaded.
- The `WIT-Projects-Component/config` argument specifies a “remote path” of the item to be loaded. Specifically, `WIT-Projects-Component` is the RTC “component” for the source code files of WIT-related projects and `config` is our familiar “config” directory, whose contents are used by the makefiles of WIT-related projects . Naturally, it's the first thing that should be loaded.

When this command executes, it displays many “Downloading” lines, e.g.:

```
Downloading /config/p_power64v5.mk  (2.4 KB)
Downloading /config/p_linux.mk  (4.8 KB)
```

If I now issue the following:

```
ls -a
```

The response is:

```
.  ..  config  .jazz5
```

The .jazz5 directory contains files that allow the current directory to function as an RTC local workspace.

Next, I would use the “load” command again, as follows:

```
lscm load -r csnext WIT-Projects-RW WIT-Projects-Component/mcl
lscm load -r csnext WIT-Projects-RW WIT-Projects-Component/wit
```

These commands load the mcl and wit directories from the WIT-Projects-RW repository workspace into the current directory.

If I now issue the following:

```
ls -a
```

The response is:

```
.  ..  config  .jazz5  mcl  wit
```

I can now build and run MCL and WIT from under the wit-projects directory.

Eventually, I might want to “unload” the local workspace, i.e., remove the local files and RTC's connection between them and the repository workspace. How is this done? Once again, there is no explicit command in the Command Line Interface to do it, i.e., to unload a local workspace. But in this case, it's really no problem. All you have to do is remove the directory where the local workspace is located. According to a forum post that I read, this is the correct way to do it. In the case of my example, I would enter:

```
rm -r /home/rjw/main/wit/wit-projects
```

I've done this several times now and haven't had any trouble with it. Note that it removes the .jazz5 directory, so that's probably why it works cleanly.

Now suppose that I have not unloaded the local workspace. How would I do the equivalent of a CVS update, i.e., download updates from the repository and merge them into my local workspace? To do this, I would use the “accept” command:

```
cd /home/rjw/main/wit/wit-projects

lscm accept
```

This command updates both the repository workspace and the local workspace with any changes that are in the repository.

If you are going to use the source code for WIT-related projects in “read only” mode, just receiving code and not making changes, then I think these may be the only commands that you need. If you are going to be making changes (especially SCE), you'll need other commands for that, e.g., checkin and deliver. For this, I refer you to the Command Line documentation link given above, but I'm available for questions.

Using the RTC Eclipse Client

Here's a scenario of how I might use the RTC Eclipse Client on Linux to work with the WIT-Projects stream:

First, in the process of joining the BAM Applications community, I received an e-mail that included the following lines at the bottom:

Use File->Accept Team Invitation within the Rational Team Concert client to open the Accept Team Invitation dialog. Copy the invitation below and paste into the Accept Invitation text field.

```
teamRepository=https://csnext.ibm.com:8003/jazz
userId=wittrock@us.ibm.com
userName=Robert J. Wittrock
projectAreaName=BAMS Applications
```

I followed this instruction and (as I recall) it set up my Eclipse environment for working with the BAMS Applications repository on CS/Next.

Next, I would log in:

- I go to the Team Artifacts view.
- I double-click Repository Connections to expand it.
- I right-click wittrock@us.ibm.com@csnext.ibm.com.
- I click Log in.
- etc.

Next, I would create a repository workspace:

- I stay in the Team Artifacts view.
- I double-click BAMS Applications to expand it.
- I double-click Source Control to expand it.
- I right-click WIT-Projects.
- I select New.
- I click Repository Workspace.
- In the next window, for Repository Workspace name, I type “WIT-Projects-RW”.
- I click Next.
- In the next window, I select one of the Read Access Permissions.
- I click Next.
- In the next window, I remove the check mark on “Load repository workspace after creation”.
- I click Finish.

This creates the repository workspace WIT-Projects-RW.

Later, if I want to remove this repository workspace, I do the following:

- I double-click My Repository Workspaces to expand it.
- I right-click WIT-Projects-RW.
- I click Delete.
- In the Confirm Delete window, if I have created a local workspace for this repository workspace, it will ask what to do with it. I select “Delete the local content associated with the repository workspace”.
- In the Confirm Delete window, I then click Yes.

This deletes the repository workspace and if I had created a local workspace, it deletes that as well.

Anyway, assuming that I have not deleted the repository workspace, I would then load a selected portion of the repository workspace into a “local workspace”, the place in my file system where the source code for WIT-related projects will be stored:

- I double-click My Repository Workspaces to expand it.
- I right-click WIT-Projects-RW.
- I click Load.
- In the next window, I select “Browse the components to select the folders to be loaded”.
- I click Next.
- In the next window, I see a list of all the directories of the WIT-related projects that are in the RTC repository: config, mcl, samples, sce, scenario, wit, witj, witm, and witutil. Each is selected with a check mark.
- I click Deselect All. The check marks are removed.
- I click the check mark boxes for config, mcl, and wit.
- I click on Advanced Options to expand this section.
- This is where I specify the local workspace, which is also called the “Sandbox”.
- On the right side, I click Browse.
- On the next window, I navigate to `/home/rjw/main/wit`.
- I click Create Folder.
- I type “wit-projects”.
- I click OK.
- I select “Load the selected folders, but do not create Eclipse projects”.
- I click Finish.

This creates the directory `/home/rjw/main/wit/wit-projects` and loads the config, mcl and wit directories from the repository workspace into it.

If I now issue the following:

```
ls -a /home/rjw/main/wit/wit-projects
```

The response is:

```
.  ..  config  .jazz5  mcl  wit
```

I can now build and run MCL and WIT from under the wit-projects directory.

Later, if I want to unload this local workspace, I would do the following:

- I double-click My Repository Workspaces to expand it.
- I right-click WIT-Projects-RW (loaded).
- I click unload.
- In the next window, I select “Delete the local content associated with the repository workspace”.
- I click Yes.

This removes the directory `/home/rjw/main/wit/wit-projects` and returns the repository workspace to an unloaded state.

Now suppose that I have not unloaded the local workspace. How would I do the equivalent of a CVS update from the Eclipse Client, i.e., download updates from the WIT-projects stream and merge them into my local workspace? To accomplish this, I would do the following:

- I go to the Pending Changes view.
- I click on the icon with a left arrow and a right arrow and with a bubble label “Refresh Remote Changes”.
- The top of this view shows the number of incoming change sets.
- I right-click WIT-Projects-RW.
- I click Accept.

This downloads the updates to my local workspace.

Now suppose I have made changes in my local workspace and I want to upload the changes to the WIT-Projects stream. Here's how I would do that:

- I go to the Pending Changes view.
- I click on the tiny black triangle that's just to the right of the icon with a left arrow and a right arrow.
- This brings up a menu.
- I click on “Refresh Sandboxes and Remote Changes”.
- I wait awhile for it to update. At the bottom of the window, there's a place where it says “Refreshing Sandboxes” and gives a percentage, indicating progress. When that disappears, it's done.
- To the left of the left and right arrows, it shows the number of local changes that it found, e.g., “2 unresolved local”.
- There are various ways to look at the changes:
    - The up and down arrows would march me through the changes.
    - If I right click on the name of my repository workspace (WIT-Projects-RW), it brings up a menu. If I then click on “Expand Children”, it expands to a hierarchical list of all the files that are changed in my local workspace.
        - I can click on any of the file names to see the differences.
        - To the left of each file name is a small arrow icon. If the array has a plus sign on it, the file is newly created; if it has a minus sign, the file has been deleted; otherwise the file is just altered.
- When I'm ready to check in the changes, I right click on Unresolved.
- This brings up a menu.
- I move the mouse to the “Check-In” item.
- This brings up a (short) menu.
- I click on “New Change set”.
- I wait awhile.
- Eventually, the “Unresolved” item disappears and a new item “Outgoing” appears.
- At this point, my changes have been checked into my repository workspace, but they're not in the stream yet.
- If I right-click on “Outgoing”, it brings up a menu.
- If I click on “Expand Children”, it expands to a hierarchical list of all the files that are changed in this newly created “change set”, and I can examine the changes in the same way as above.

- Immediately under “Outgoing”, there is an item, “<Enter a Comment>”.
  (I may have to click on “Outgoing” to see this item.)
- I click on the item.
- I push F2.
- I now type my comment, a label for this change set.
- I push Enter to add the comment to the change set.
- I right click on the comment.
- This beings up a menu.
- I click Deliver.
- It “delivers” the new change set into the WIT-Projects stream.
- That's it.

Next, suppose that I've made some local changes and I decide that I want to undo some or all of them. Here's what I would do:

- I would go through the “Refresh Sandboxes and Remote Changes” procedure ()see above), but without checking in.
- To undo all changes, I would click on “Unresolved”.
- To undo the changes in some files and not others, I would ”Expand Children” (see above) then control-click each file to be undone, thereby selecting it.
- I then right-click on something selected.
- This brings up a menu.
- I click “Undo”.
- This brings up an “Are you sure?” window.
- I click OK.
- I wait awhile.
- The changes disappear from Eclipse.
- If I now look at the files, they are in the state that they were in before I changed them.

Finally, suppose I want to look at an old version of a file, i.e., I want to see a file as it was on a particular date. For concreteness, let's say the file is wit/src/wit.h and I want to see it as it was on July 6, 2012. Here's what I would do:

- I double-click My Repository Workspaces to expand it.
- I right-click WIT-Projects-RW (loaded).
- I click Show.
- I click Repository Files.
- This brings up a pane in which I can navigate through the WIT-Projects files.
- I navigate to the file wit/src/wit.h.
- I right-click wit.h
- I click Show History.
- This brings up a table of all the change sets that have been applied to the file wit..h.
- I maximize this pane.
- I can now see a column in the table that shows the date and time of each change.
- I right click on the row with the last date before the date that I'm interested in, in this case, June 18, 2012.
- I click Open File.
- This opens an editor (in my case, Xemacs) with a temporary copy of the file as it was just after the change.
- I can now do whatever I want with that temporary copy.