

XLr8Editor

BUILT ON
eclipse.

Version Control 2

A **U2logic** PUBLICATION

© 2016, U2logic, Inc.
All Rights Reserved.

XLr8Editor Version Control 2

Doug Averch

Version 1.0
Created April 2016
Updated May 24, 2016

Printed in the United States of America
U2logic, 13963 Powhaton Road, Brighton, Colorado 80603

This product and its documentation are protected by copyright and are distributed with licensed copies of the respective software system. This document and the confidential information contained within it may in no part be distributed, reproduced, stored in or introduced into a retrieval system, or transmitted in any form or means without the express written permission of U2logic, Inc.

U2logic is not responsible for errors or omissions in this guide. U2logic reserves the right to change the information described herein at any time without notification.

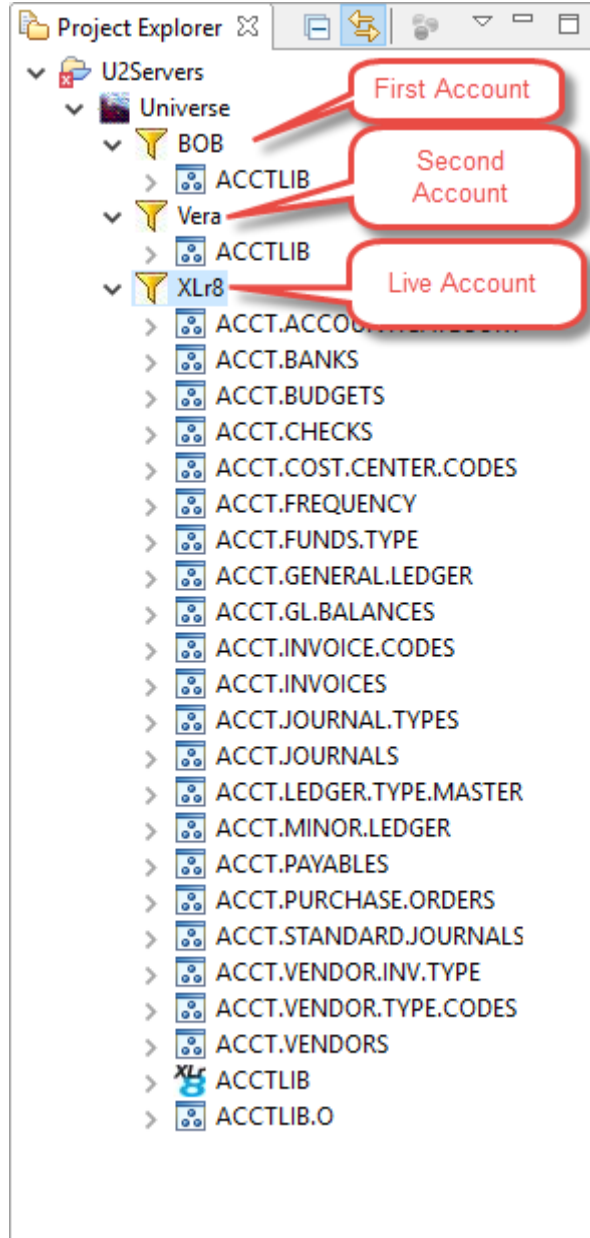
I. Executive Summary

Eclipse is fully version control cognizant. Eclipse comes built in with CVS versioning. You can add other version control system as long as there is a plug-in for them. Currently, Eclipse supports these version control systems:

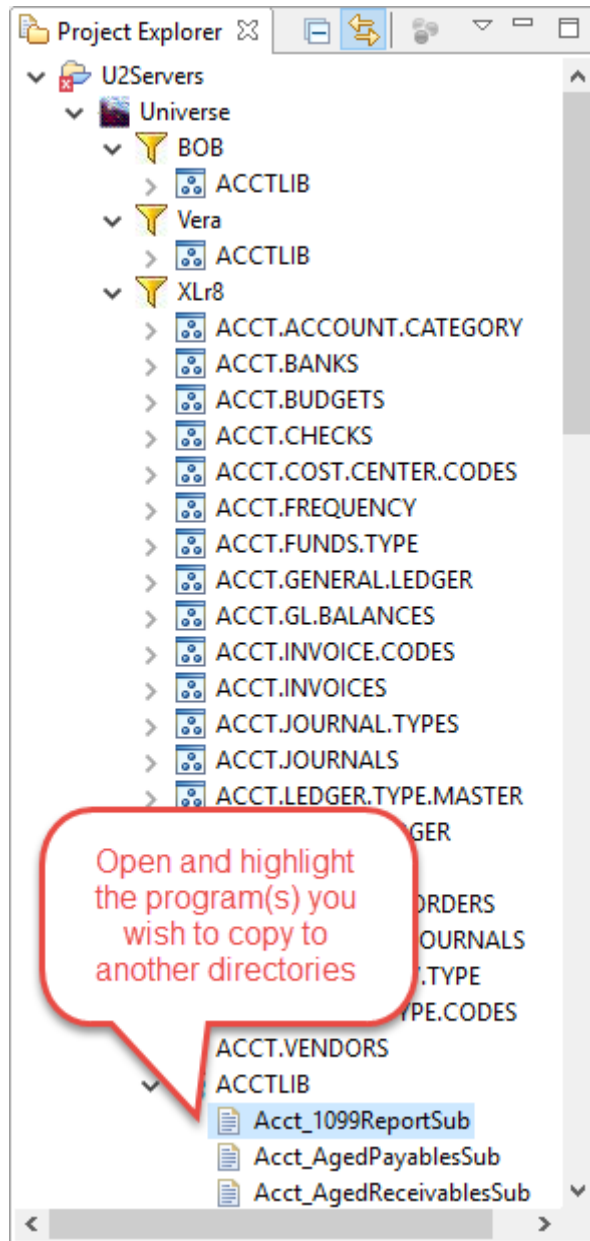
1. CVS
 - a. Built-in no extra plug-in required
 - b. Windows Server version is available at <http://www.march-hare.com/>
2. Subversion
 - a. Requires plug-in see http://subclipse.tigris.org/update_1.10.x
 - b. Windows Server version is available at <http://www.visualsvn.com/server/>
3. Perforce
 - a. Requires plug-in see <http://www.perforce.com/>
 - b. Requires server see <http://www.perforce.com/>
4. GIT
 - a. Requires plug-in see <http://download.eclipse.org/egit/updates>
 - b. For server see <http://github.com/>
5. Team Foundation Server
 - a. Requires plugin see <http://msdn.microsoft.com/en-us/library/gg413285.aspx>
 - b. Requires server see <http://msdn.microsoft.com/en-us/vstudio/ff637362>

II. Setup Programmer Accounts

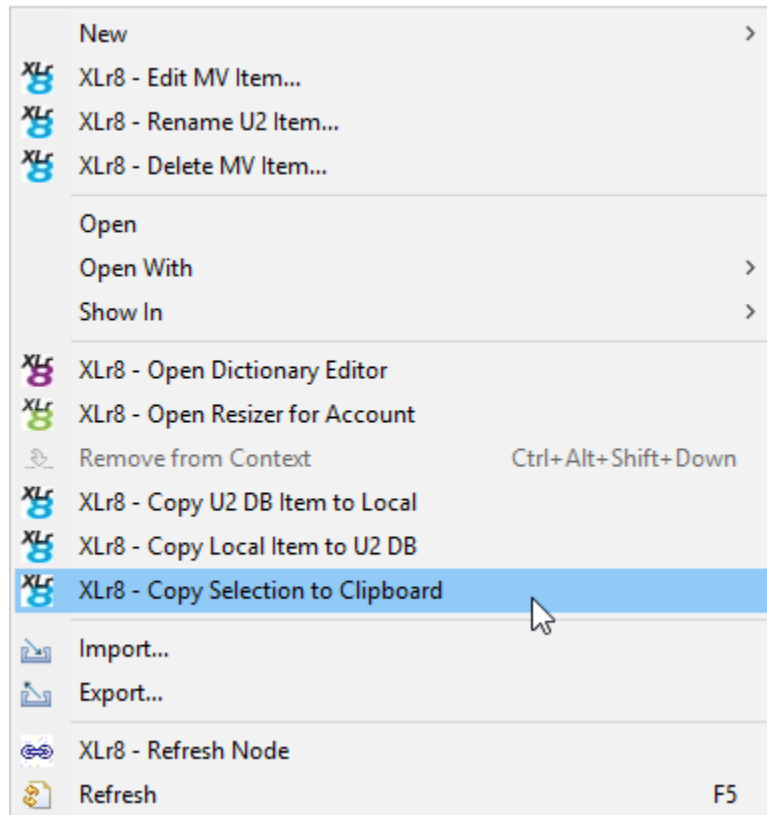
Create your source accounts and add them to the project explorer.



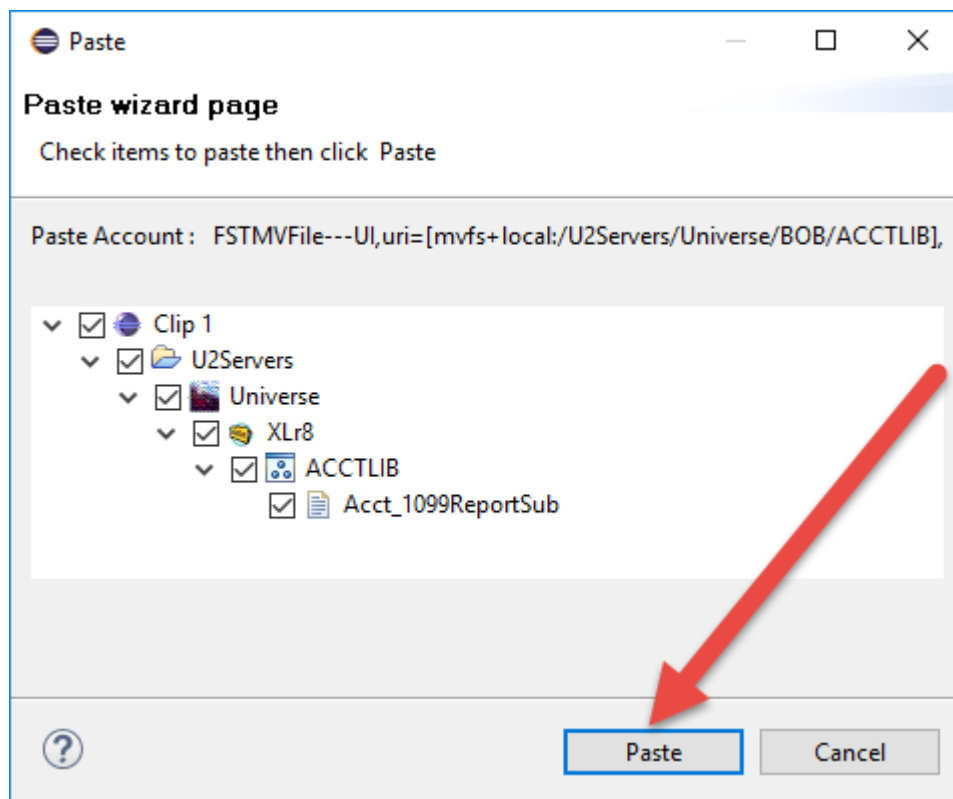
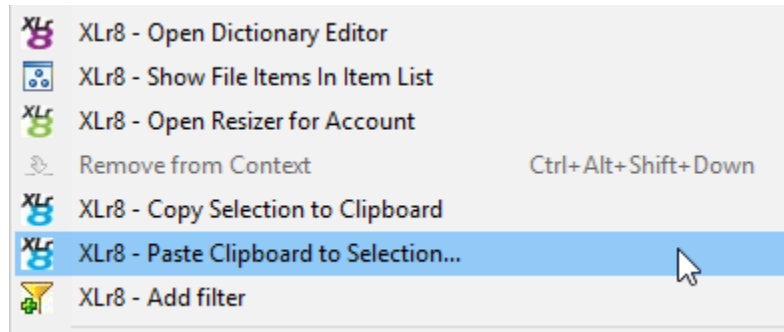
Use the project explorer to highlight the programs you would like to copy.



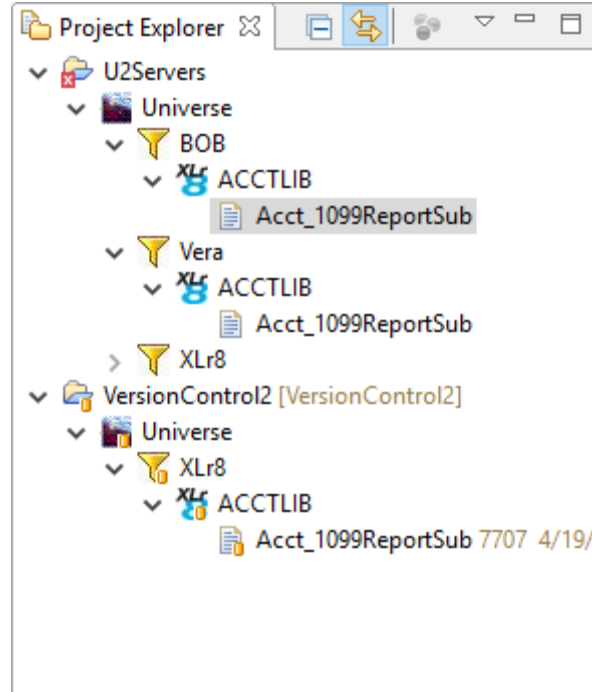
Right-click on the program and highlight the “Copy Selection to Clipboard”



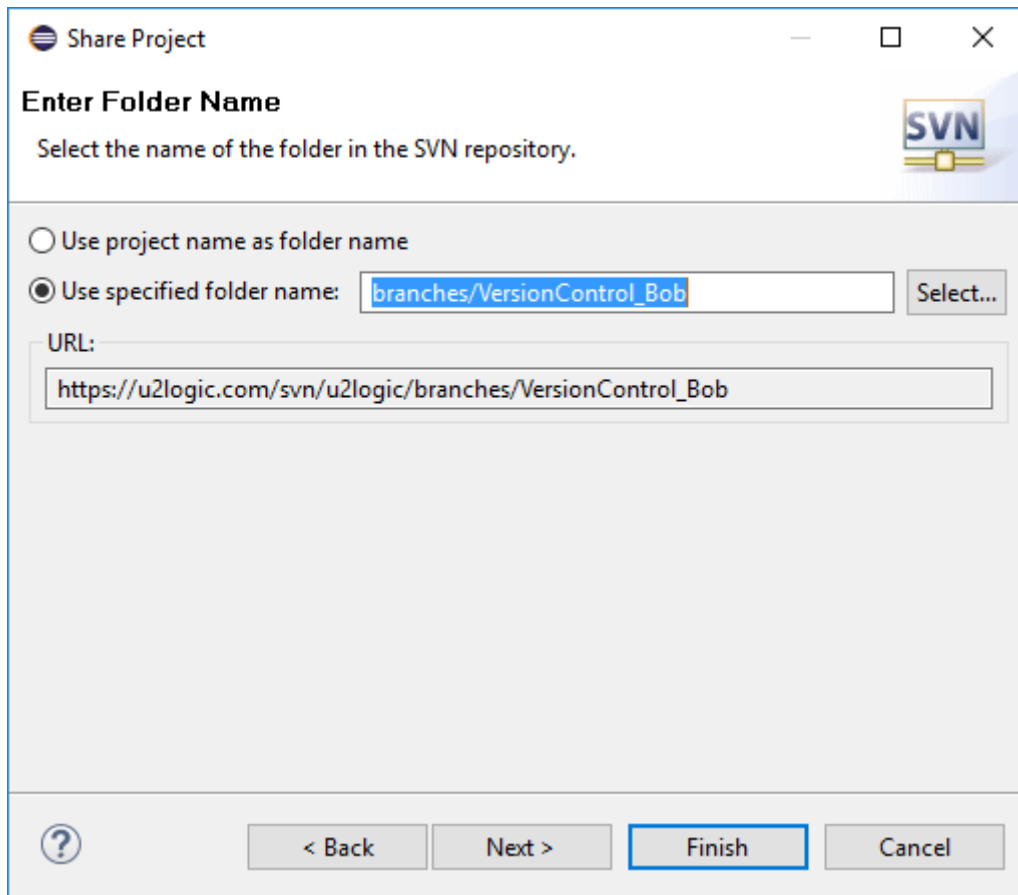
After you highlight the directory where you want to put the source, right-click and pick the “Paste Clipboard to Selections...”. With the paste wizard page press the “Paste” button to copy the source code to this directory.



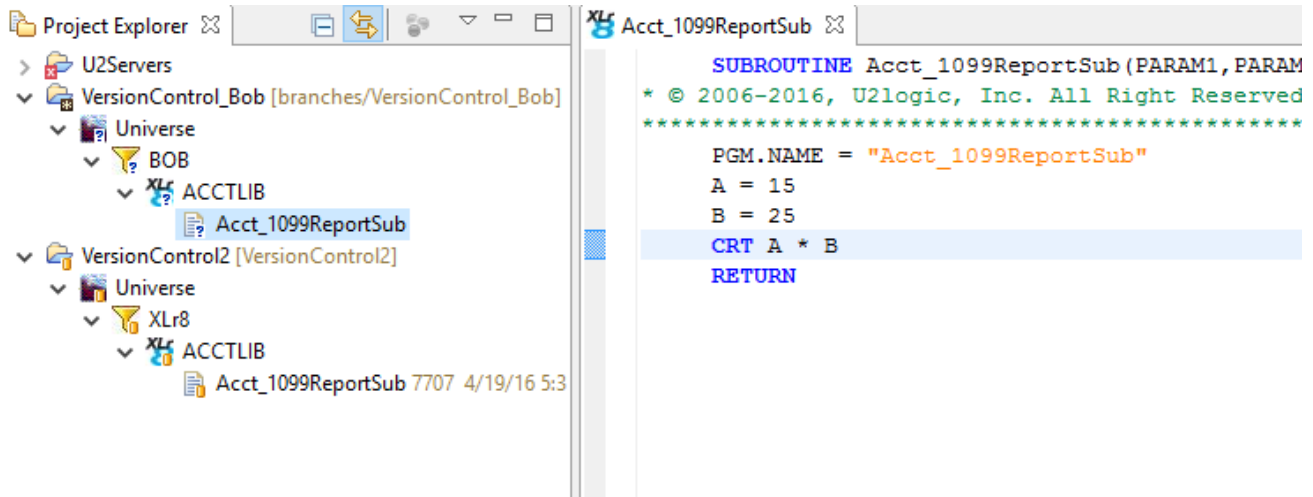
Create a linked project and share it using your version control. Link your source code, procs, and dictionaries to this project. Then check in your code.



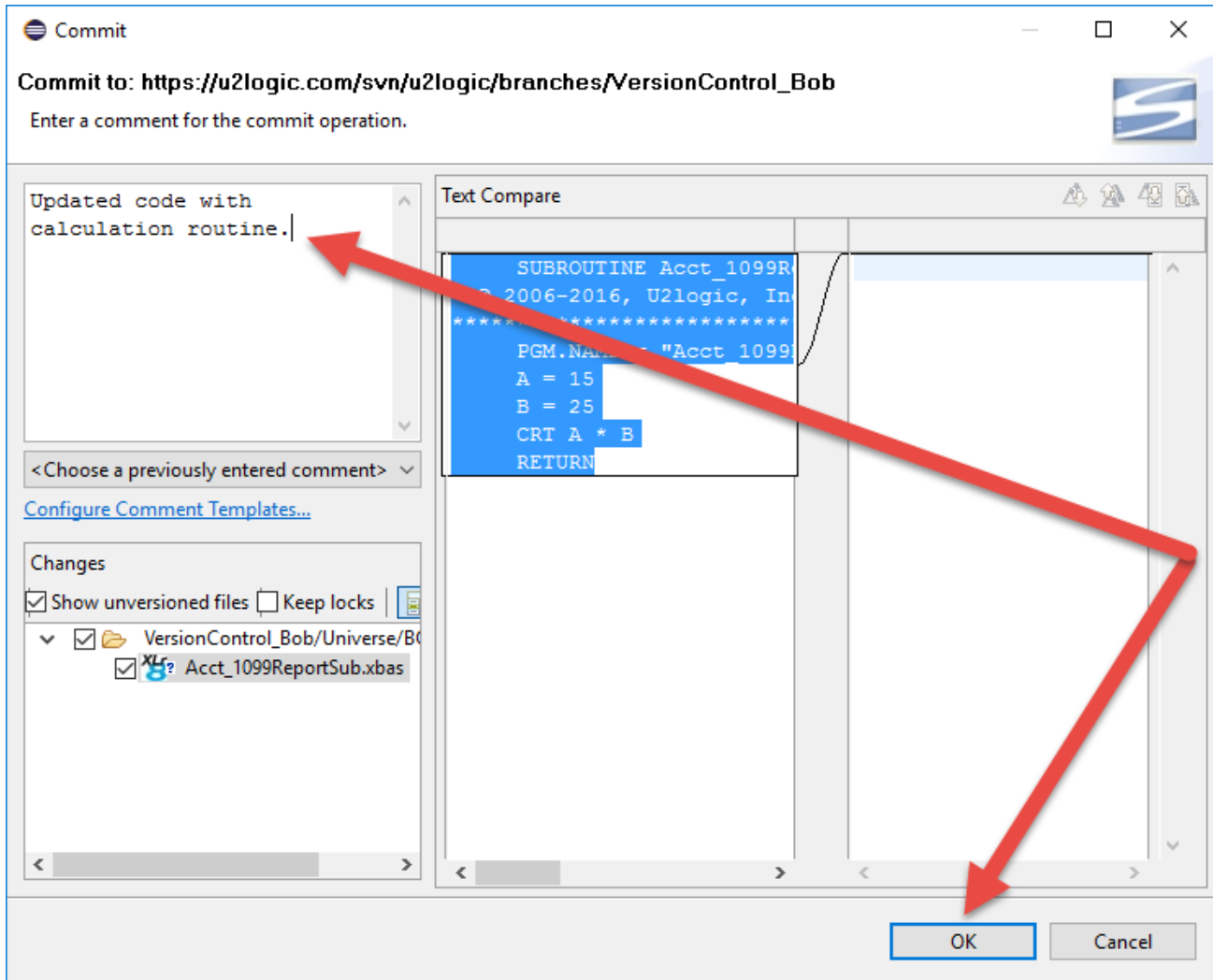
After you have created a linked project called VersionControl_Bob, attached the source code, you share your project as a branch in version control.



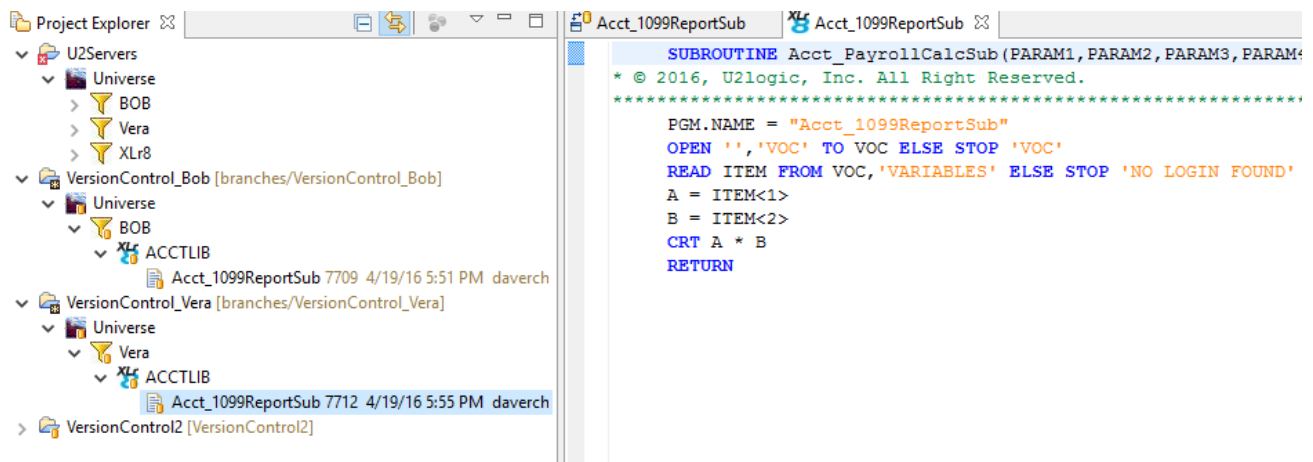
Bob has now changed the source code in his account and in his local linked project.



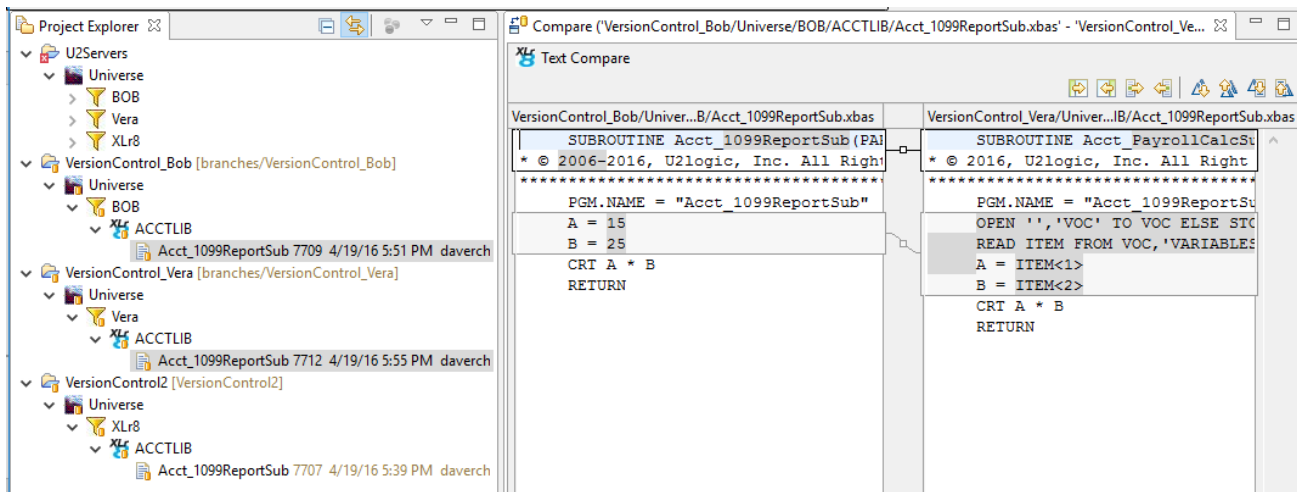
Bob's code is now being checked in to his branch



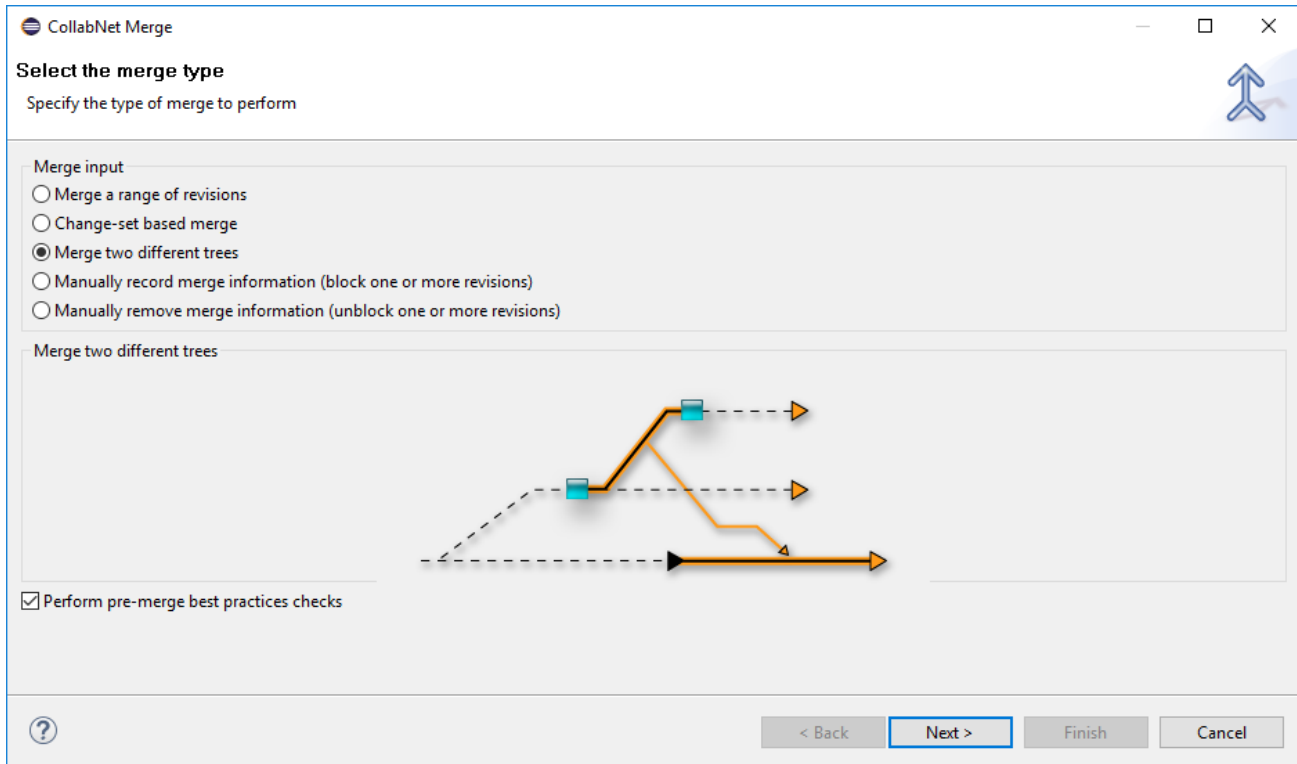
Both Bob and Vera have checked in their code into their respective branches as well as having their accounts having this different source code.



Using the right click menu option, you can compare two source codes and merge the changes you would like keep.



We are going to merge two different trees using SVN CollabNet.



Select the merge sources and revisions you are merging.

CollabNet Merge

Select the merge source and revisions

Specify the URL and revisions containing the items to merge

From: (start URL and revision of the range to merge)

/branches/VersionControl_Bob/Universe/BOB/ACCTLIB/Acct_1099ReportSub.xbas Select...

HEAD Revision Revision 7757 Select...

To: (end URL and revision of the range to merge)

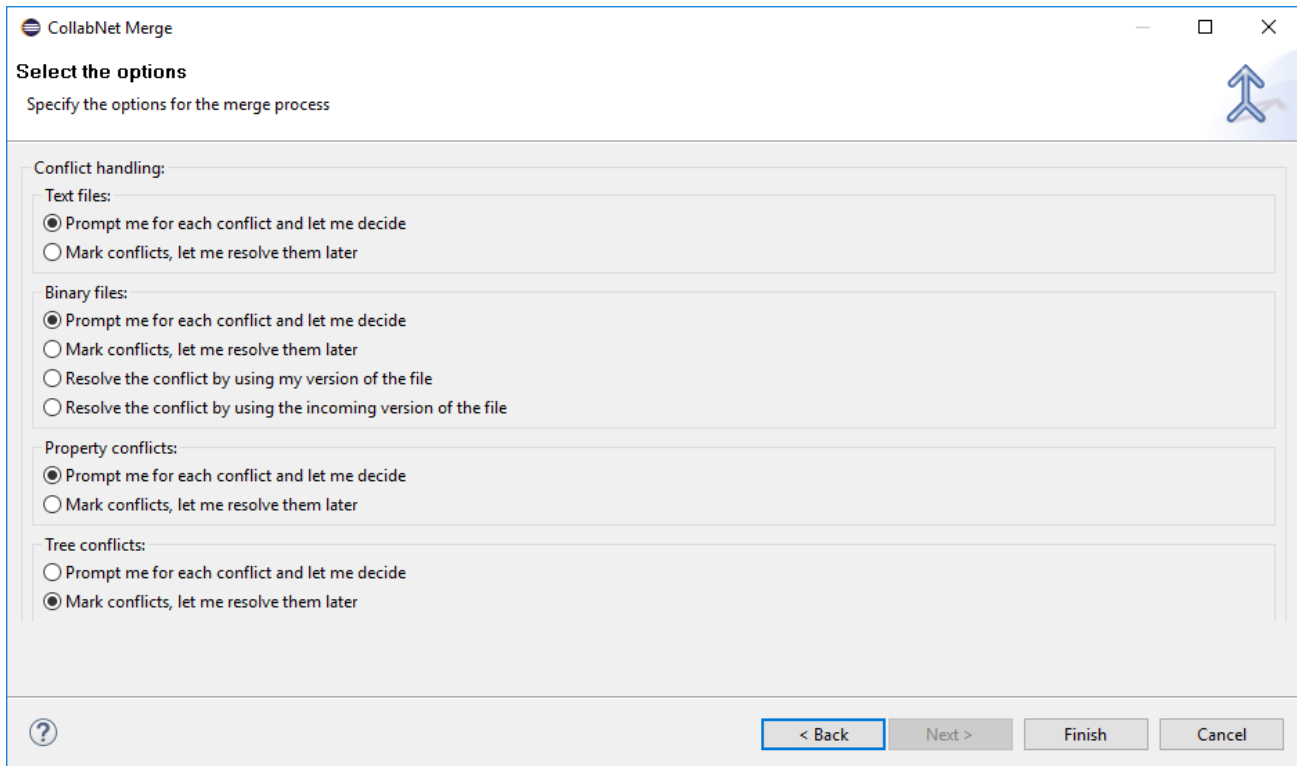
Use "From:" URL

https://u2logic.com/svn/u2logic/branches/VersionControl_Vera/Universe/Vera/ACCTLIB/Acct_1099ReportSub.xbas

HEAD Revision Revision 7712 Select...

< Back Next > Finish Cancel

Select the options you would like use on merge.

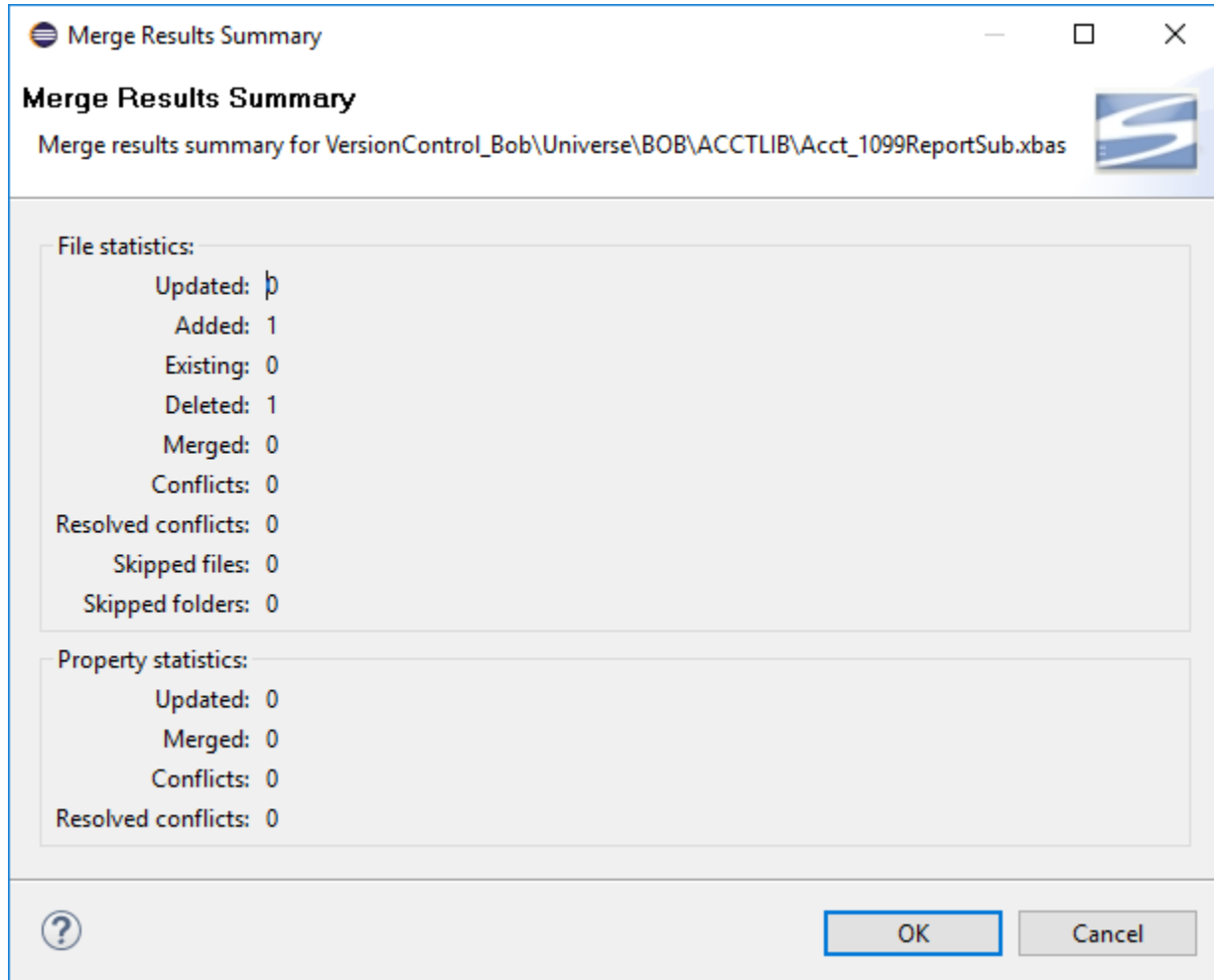


The screenshot shows a dialog box titled "CollabNet Merge" with the subtitle "Select the options" and "Specify the options for the merge process". The dialog is divided into four sections for conflict handling:

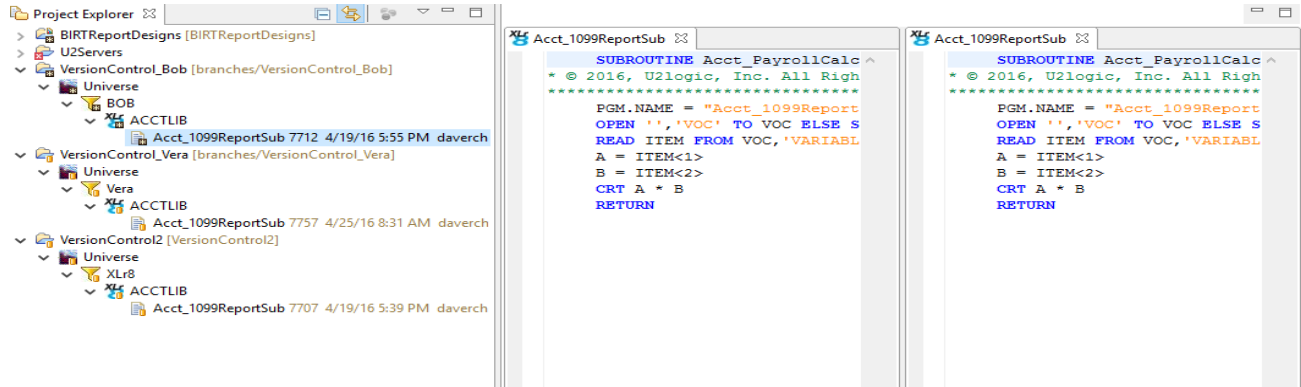
- Text files:**
 - Prompt me for each conflict and let me decide
 - Mark conflicts, let me resolve them later
- Binary files:**
 - Prompt me for each conflict and let me decide
 - Mark conflicts, let me resolve them later
 - Resolve the conflict by using my version of the file
 - Resolve the conflict by using the incoming version of the file
- Property conflicts:**
 - Prompt me for each conflict and let me decide
 - Mark conflicts, let me resolve them later
- Tree conflicts:**
 - Prompt me for each conflict and let me decide
 - Mark conflicts, let me resolve them later

At the bottom of the dialog, there is a help icon (question mark) on the left and four buttons: "< Back", "Next >", "Finish", and "Cancel".

This is the merge results summary based on our prior actions.



Bob's code is now merged from Vera's. Although it is highly not recommended to use this method because the issues with global scope of variables in UniBasic, the version control system did merge the code correctly.



The last step would be to merge the code Bob's or Vera's code to our Version Control 2 source code control which would be the same as the Bob to Vera merge.

