Paper 2415-2018

# Finding Best Tools for SAS Programs Change Management within a Regulated Environment

Sy Truong, Pharmacyclics, Inc.; Pradheep Raman, Pharmacyclics, Inc.

## ABSTRACT

Change management within regulated environments among a multi-functional team, over different projects can pose many challenges. Recent enhancements to SAS Enterprise Guide have added new debugging features along with tools to monitor and maintain audit trails of SAS programs. These features are easy to use and powerful. There are also other tools apart from SAS such as GitHub that can also be used to manage changes to SAS programs. This paper will evaluate all the tools mentioned to find the best solution.  Some of the criteria included in this evaluation include:

1. Ease of Use – How easy is it for users to gain program change management capabilities.
2. SAS Environments – Apart from an EG project file, can it adapt to SAS files in batch mode or stored processes.
3. Configuration and Administration – Ease of setup and does it require additional administration efforts beyond those of a SAS programmer.
4. Features Rich – How many different useful functions does the tool have for SAS programs and related outputs.

Data integrity is of paramount importance in a regulated environment which involves tracking changes and maintaining an audit trail. Uniqueness of a statistical computing environment is due to system configuration and the end user’s usage of the application. Thus, the selection of an optimal change management tool that does not significantly impact user’s productivity is imperative.

## EVALUTATION OVERVIEW

The objective of this evaluation is to identify a tool or a set of tools to help manage the changes in SAS programs during the development of an analysis and reporting within a small team.  This will allow the team members to collaborate on the SAS programs, while maintaining greater integrity.  The evaluation will be from the perspective of a programmer.  However, other members such as: statisticians, data mangers, IT support staff or other related team members will contribute to the evaluation.  Although each tool has different strengths and weaknesses; the goal is to identify the best tool that is most efficient for a small SAS programming groups.

Every computing environment and end user behaviors are unique. Therefore, this paper will describe the pros and cons of tools for one example group, rather than to identify a solution that is universal. This is in hopes that the results can help others identify the best solutions for themselves.  Some of the attributes of the current group and computing environment where this evaluation is performed include:

- **Group Size –** The projects that include SAS programmers, validation testers, statisticians, data management, and reviewers can range from 5 to 15 members.  There are multiple projects running simultaneously which can lead to team members switching among teams.  The larger group that works on all the projects combined range between 60 to 120 members.

- **Server Operating System –** Users log onto a Windows Server using remote desktop to run SAS 9.4. The desktop or laptops run Windows 7 while the server runs Windows 2008 R2.
- **Access Permissions –** Users access are controlled by active directory domain groups. An IT administrator works in collaboration with a business administrator to set groups which then are assigned the specific folders and related files with the desired permissions.
- **Manual Change Control –** Prior to utilizing the tools being evaluated, users would manually document changes by updating the program header that includes the user's name and date pertaining to major changes performed to the program.

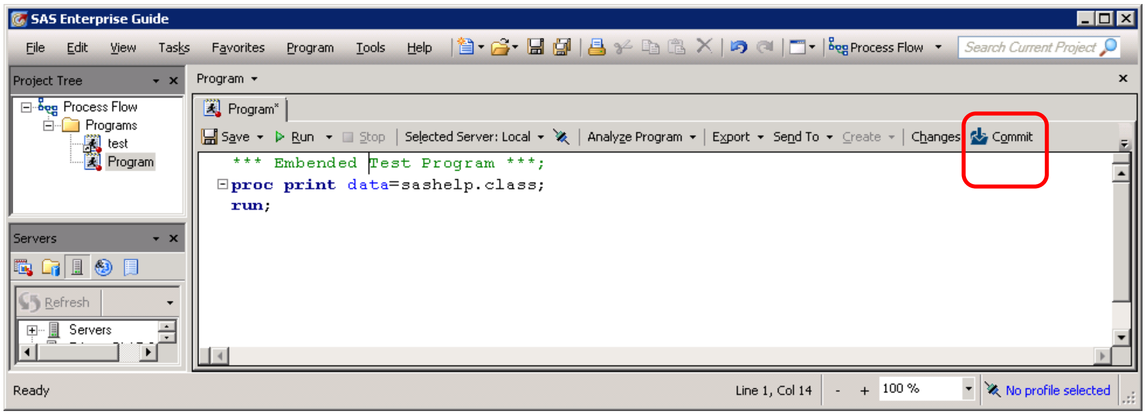
The implementation of a new tool requires major changes to the current computing environment. An evaluation needs to be performed to correlate the tool benefits as opposed to the users work flow changes.

## EASE OF USE

One of the main criteria used in this evaluation is to measure the ease of use of the tool for a SAS programmer to gain the benefits of SAS programs change management. This section elaborates upon the user's requirements as it relates to usability of the tools and how the tools meets these needs. This can span from a general user experience to a more specific feature culminate to aiding the user in managing changes to their programs.

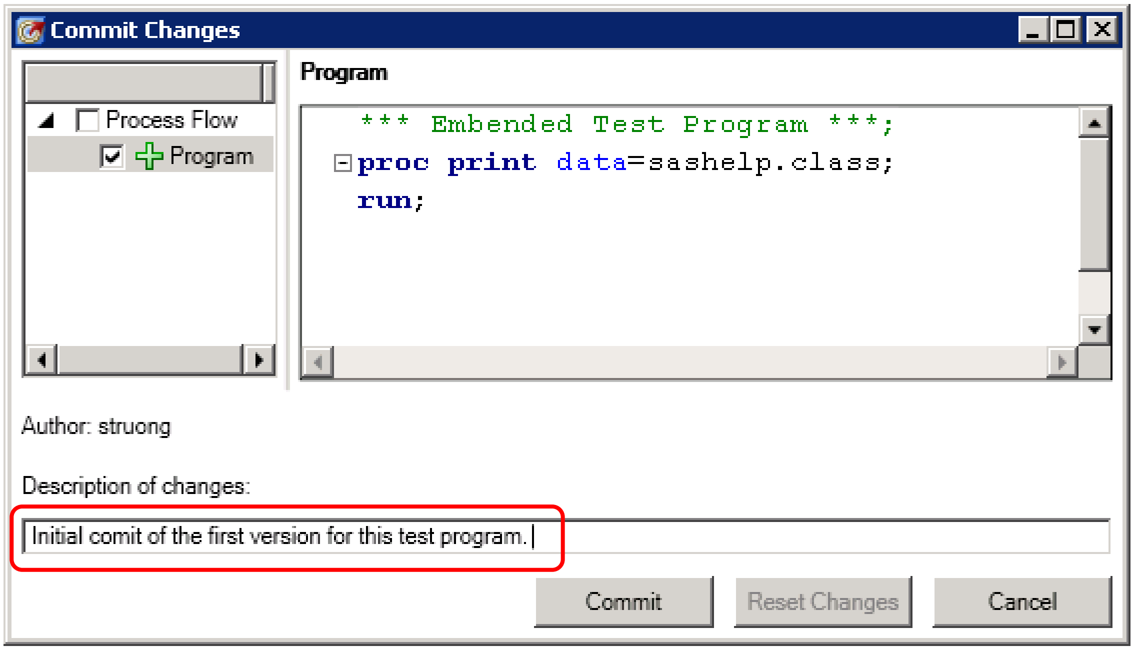
SAS Enterprise Guide has one of the most user-friendly interfaces for SAS programmers to manage change control. For SAS programs embedded to EG projects, the user can perform the following steps to start to track changes.

1. **Commit –** From the program editor within SAS EG, click on the "commit" button on the upper right.



```
  *** Embended Test Program ***;
proc print data=sashelp.class;
run;
```

2. **Change Description –** There is an option to type a short description capturing the meaning of this change.



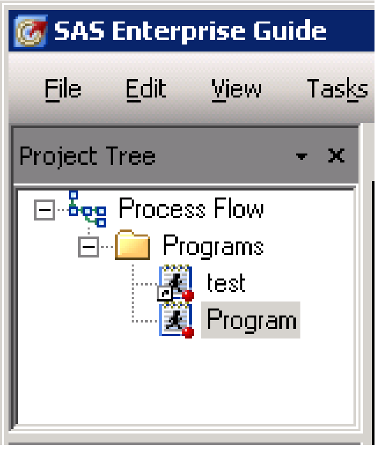
```
*** Embended Test Program ***;
proc print data=sashelp.class;
run;
```

Author: struong

Description of changes:

Initial comit of the first version for this test program.

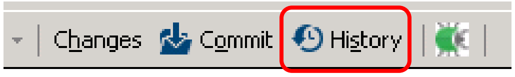
There is no arduous configuration or setup for this, if the SAS program is embedded in an EG project. For external SAS programs on the file system, there are additional configuration requirements. The difference can be seen on the project tree on the left panel within EG.



The embedded program is shown as just the running man icon. However, an external SAS program file will have a “shortcut” arrow on the lower left of the icon. If the SAS environment were to use embedded SAS programs and programs are stored within SAS EG project files, this is the most user-friendly interface for committing changes.

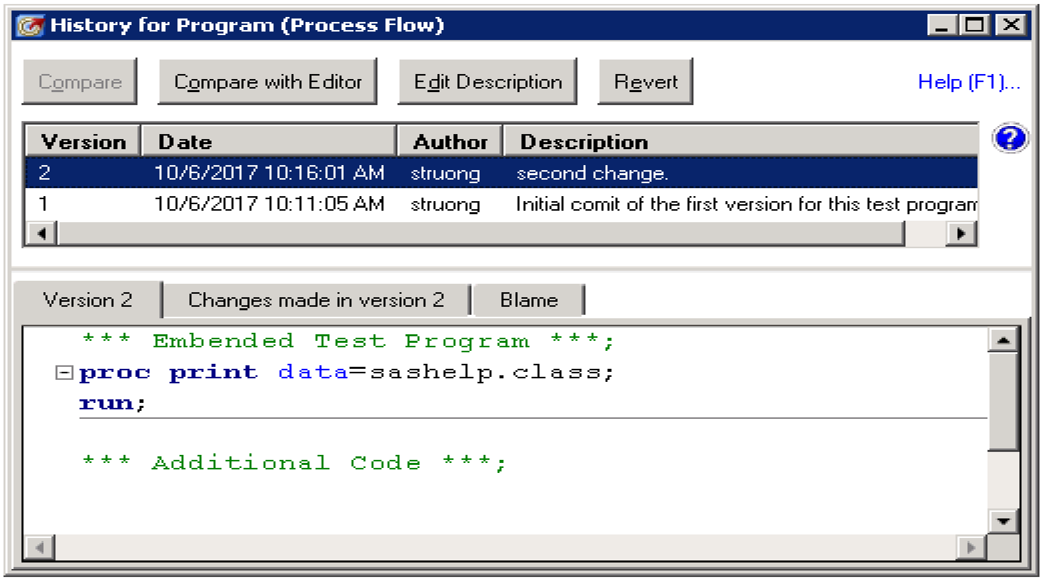
In addition to committing changes, reviewing a program history to see what changes were made and rolling back to a previous version is just as easy. This is accomplished through the following steps.

1. **Initiate History –** Click on the “History” button that appears on the upper right next to the “Commit” button.



2. **Review Old Programs –** This brings up a history view listing out all the previous versions that have been committed. By clicking on the specific version on the top panel list, the

corresponding SAS program code is displayed on the lower panel.



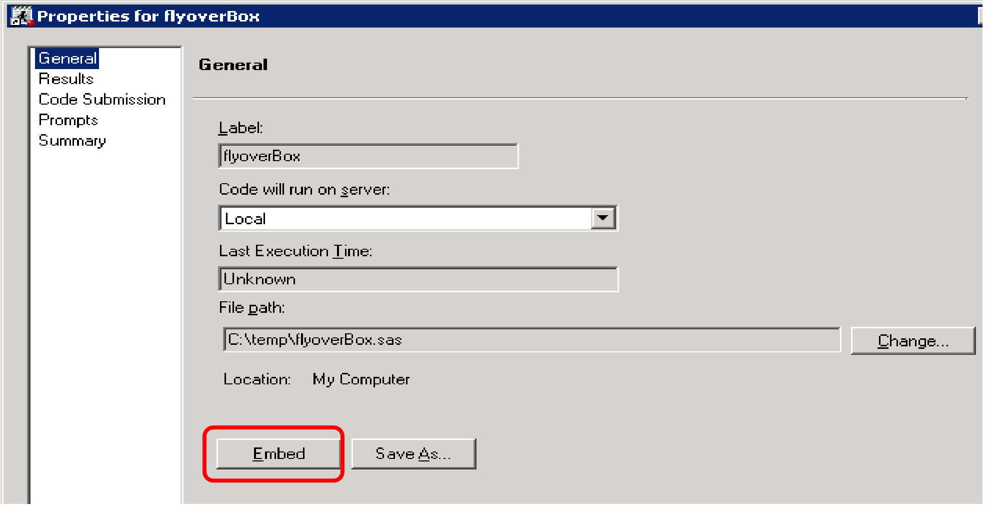
| Version | Date | Author | Description |
|---|---|---|---|
| 2 | 10/6/2017 10:16:01 AM | struong | second change. |
| 1 | 10/6/2017 10:11:05 AM | struong | Initial comit of the first version for this test program |

```
*** Embended Test Program ***;
proc print data=sashelp.class;
run;

*** Additional Code ***;
```

3. **Reverting –** If an error is found in the current version of a program, the user can view an older version by clicking on the "Revert" button in order to roll back to the selected previous version.

These user-friendly steps allow users to perform the most common tasks in managing changes to their SAS programs. This makes for a user experience that is both intuitive to use and integrates well into the SAS Enterprise Guide and SAS interface.

An important distinction with the above features is that it is available for embedded SAS programs which are added to an EG project. To access SAS programs that reside on the file server directly, users will need to add them into an EG project and "embed" the files by navigating through to the properties menu within the process flow as shown below:



Label: flyoverBox

Code will run on server: Local

Last Execution Time: Unknown

File path: C:\temp\flyoverBox.sas

Location: My Computer

This step will need to be applied one at a time for each "external" SAS programs. The advantage is that once embedded, all the features of change control describe above is available. The disadvantage is that users can no longer use their “external” SAS programs residing on the file system since those files are not synchronized with the corresponding programs embedded into the EG project. This may impact the workflow and should be evaluated accordingly.

On a related observation about EG project files; each project file is designed to be edited by one user at one time. Other tools evaluated in this paper provided an option for multiple users to be able to edit the SAS programs at the same time. The current version of EG does not allow for this since it will lock out a second user attempting to edit the same project file. EG works well for a single user, however, if a multi-user environment is required, this can be a limiting factor.

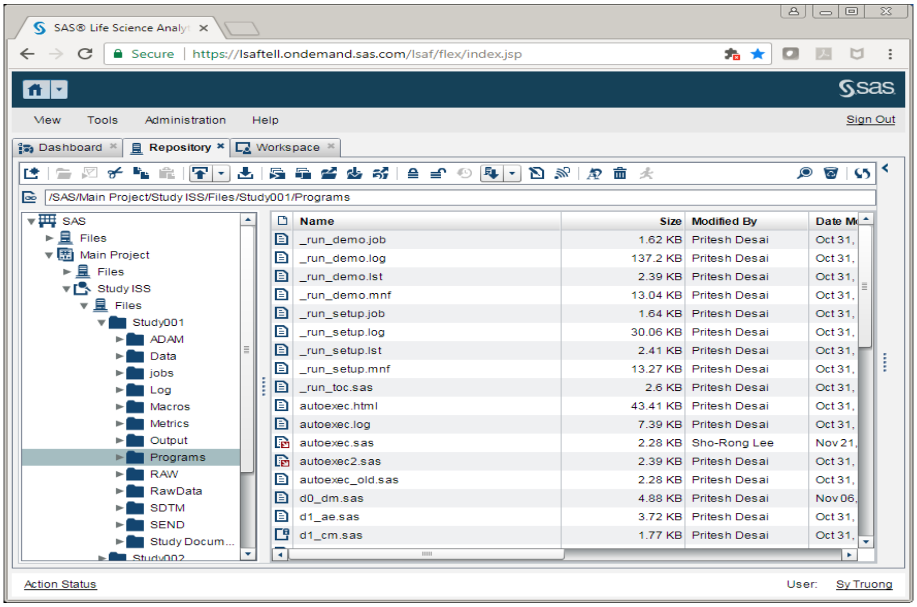
## SAS ENVIRONMENT

The main tools being evaluated in this paper are implemented on a Window server with users logging onto to the server using remote desktop. All the files such as SAS programs and related SAS datasets and output files are stored within folders controlled through active directory. The general computing environment is primarily using Microsoft operating systems and tools. If the evaluation were to be performed within a different operating system such as Unix or if everything needs to be applied through a thin client web browser, the evaluation criteria and end results will be very different.

Even though it is not the focus of this paper, two other tools were also reviewed that is not applied within a Windows Server environment. This includes SAS LSAF and Entimo. These solutions provide SAS program change control but do more in providing an entire environment to manage projects for SAS programmers within the Biotech Pharmaceutical Industry.

The following section captures steps illustrating how LSAF manage change control for this evaluation.

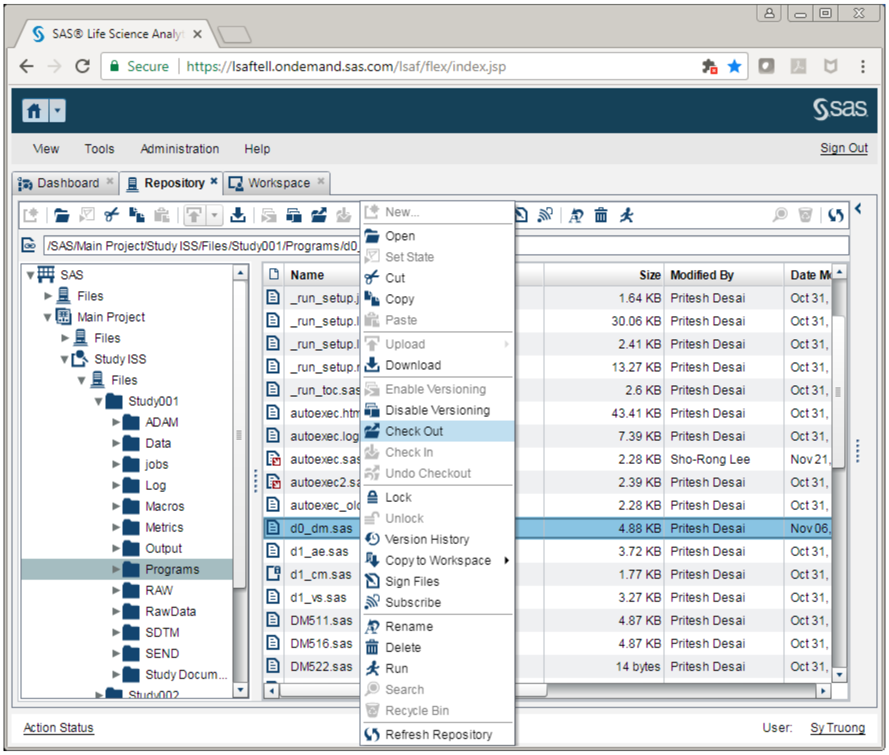
1. Login using URL: [https://lsaftell.ondemand.sas.com/lsaf/flex/index.jsp](https://lsaftell.ondemand.sas.com/lsaf/flex/index.jsp)
2. Navigate to “Repository” and then the path:



/SAS/Main Project/Study ISS/Files/Study001/Programs

| Name | Size | Modified By | Date M |
|---|---|---|---|
| _run_demo.job | 1.62 KB | Pritesh Desai | Oct 31, |
| _run_demo.log | 137.2 KB | Pritesh Desai | Oct 31, |
| _run_demo.lst | 2.39 KB | Pritesh Desai | Oct 31, |
| _run_demo.mnf | 13.04 KB | Pritesh Desai | Oct 31, |
| _run_setup.job | 1.64 KB | Pritesh Desai | Oct 31, |
| _run_setup.log | 30.06 KB | Pritesh Desai | Oct 31, |
| _run_setup.lst | 2.41 KB | Pritesh Desai | Oct 31, |
| _run_setup.mnf | 13.27 KB | Pritesh Desai | Oct 31, |
| _run_toc.sas | 2.6 KB | Pritesh Desai | Oct 31, |
| autoexec.html | 43.41 KB | Pritesh Desai | Oct 31, |
| autoexec.log | 7.39 KB | Pritesh Desai | Oct 31, |
| autoexec.sas | 2.28 KB | Sho-Rong Lee | Nov 21, |
| autoexec2.sas | 2.39 KB | Pritesh Desai | Oct 31, |
| autoexec_old.sas | 2.28 KB | Pritesh Desai | Oct 31, |
| d0_dm.sas | 4.88 KB | Pritesh Desai | Nov 06, |
| d1_ae.sas | 3.72 KB | Pritesh Desai | Oct 31, |
| d1_cm.sas | 1.77 KB | Pritesh Desai | Oct 31, |

3. Right mouse click on a program such as d0_dm.sas and select “Checkout”

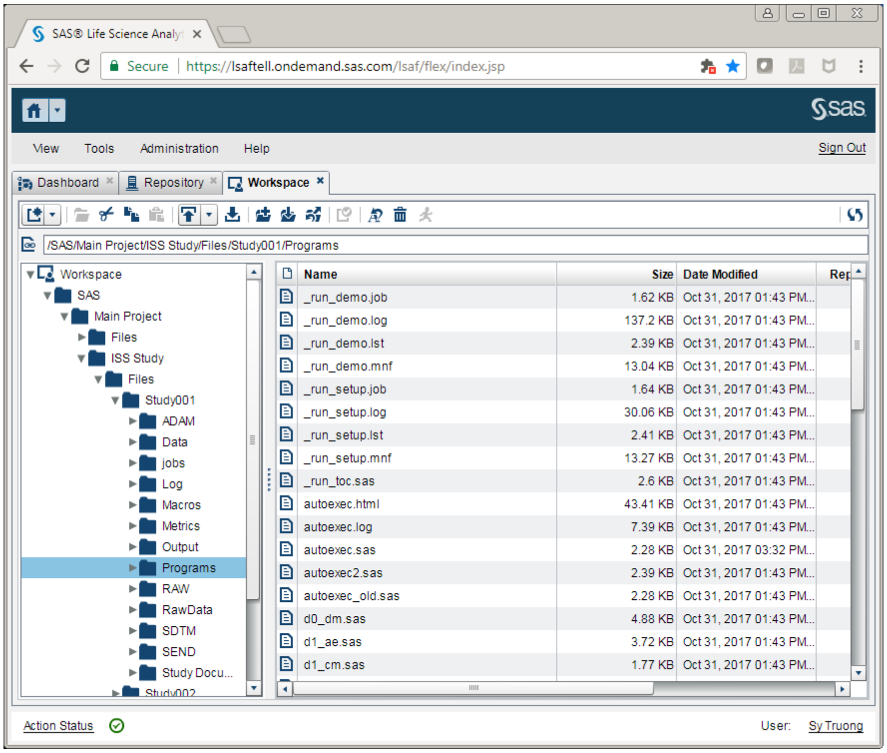


/SAS/Main Project/Study ISS/Files/Study001/Programs/d0_

| Name | Size | Modified By | Date M |
|---|---|---|---|
| _run_setup.j | 1.64 KB | Pritesh Desai | Oct 31, |
| _run_setup.l | 30.06 KB | Pritesh Desai | Oct 31, |
| _run_setup.l | 2.41 KB | Pritesh Desai | Oct 31, |
| _run_setup.r | 13.27 KB | Pritesh Desai | Oct 31, |
| _run_toc.sas | 2.6 KB | Pritesh Desai | Oct 31, |
| autoexec.htm | 43.41 KB | Pritesh Desai | Oct 31, |
| autoexec.log | 7.39 KB | Pritesh Desai | Oct 31, |
| autoexec.sas | 2.28 KB | Sho-Rong Lee | Nov 21, |
| autoexec2.sa | 2.39 KB | Pritesh Desai | Oct 31, |
| autoexec_ol | 2.28 KB | Pritesh Desai | Oct 31, |
| d0_dm.sas | 4.88 KB | Pritesh Desai | Nov 06, |
| d1_ae.sas | 3.72 KB | Pritesh Desai | Oct 31, |
| d1_cm.sas | 1.77 KB | Pritesh Desai | Oct 31, |
| d1_vs.sas | 3.27 KB | Pritesh Desai | Oct 31, |
| DM511.sas | 4.87 KB | Pritesh Desai | Oct 31, |
| DM516.sas | 4.87 KB | Pritesh Desai | Oct 31, |
| DM522.sas | 14 bytes | Pritesh Desai | Oct 31, |

Note the icon of the program will then indicate it has been checked out.

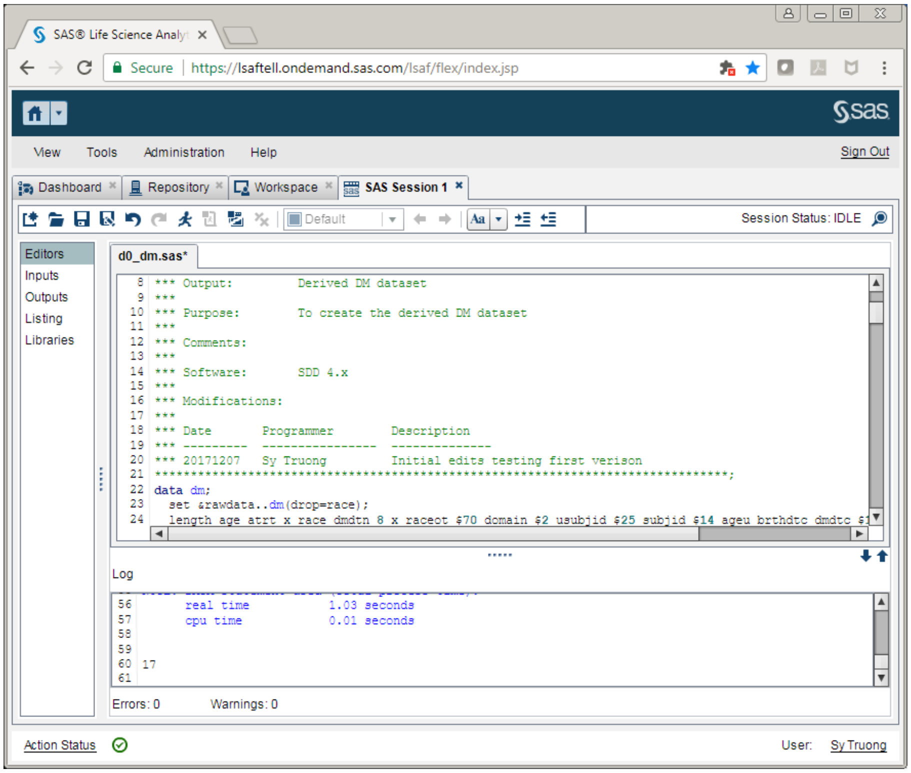
4. Open the “Workspace” and then navigate to the corresponding path:



/SAS/Main Project/ISS Study/Files/Study001/Programs

| Name | Size | Date Modified | Rep |
|---|---|---|---|
| _run_demo.job | 1.62 KB | Oct 31, 2017 01:43 PM... | |
| _run_demo.log | 137.2 KB | Oct 31, 2017 01:43 PM... | |
| _run_demo.lst | 2.39 KB | Oct 31, 2017 01:43 PM... | |
| _run_demo.mnf | 13.04 KB | Oct 31, 2017 01:43 PM... | |
| _run_setup.job | 1.64 KB | Oct 31, 2017 01:43 PM... | |
| _run_setup.log | 30.06 KB | Oct 31, 2017 01:43 PM... | |
| _run_setup.lst | 2.41 KB | Oct 31, 2017 01:43 PM... | |
| _run_setup.mnf | 13.27 KB | Oct 31, 2017 01:43 PM... | |
| _run_toc.sas | 2.6 KB | Oct 31, 2017 01:43 PM... | |
| autoexec.html | 43.41 KB | Oct 31, 2017 01:43 PM... | |
| autoexec.log | 7.39 KB | Oct 31, 2017 01:43 PM... | |
| autoexec.sas | 2.28 KB | Oct 31, 2017 03:32 PM... | |
| autoexec2.sas | 2.39 KB | Oct 31, 2017 01:43 PM... | |
| autoexec_old.sas | 2.28 KB | Oct 31, 2017 01:43 PM... | |
| d0_dm.sas | 4.88 KB | Oct 31, 2017 01:43 PM... | |
| d1_ae.sas | 3.72 KB | Oct 31, 2017 01:43 PM... | |
| d1_cm.sas | 1.77 KB | Oct 31, 2017 01:43 PM... | |

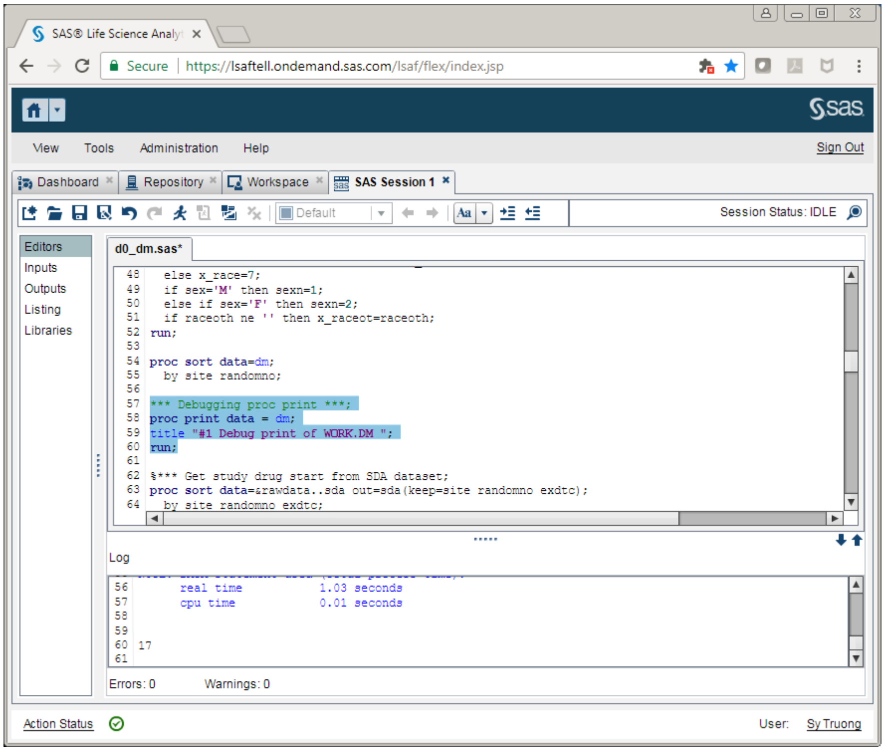
5. Double click on the program d0_dm.sas. In the program editor, edit changes in program header:



```
*** Output:         Derived DM dataset
***
*** Purpose:        To create the derived DM dataset
***
*** Comments:
***
*** Software:       SDD 4.x
***
*** Modifications:
***
*** Date       Programmer        Description
*** ---------  ----------------  --------------
*** 20171207   Sy Truong         Initial edits testing first verison
*******************************************************************************;
data dm;
  set &rawdata..dm(drop=race);
  length age atrt x race dmdtn 8 x raceot $70 domain $2 usubjid $25 subjid $14 ageu brthdtc dmdtc $1
```

```
      real time           1.03 seconds
      cpu time            0.01 seconds


 17

```

6. Scroll down within the program and insert new code such as this example:

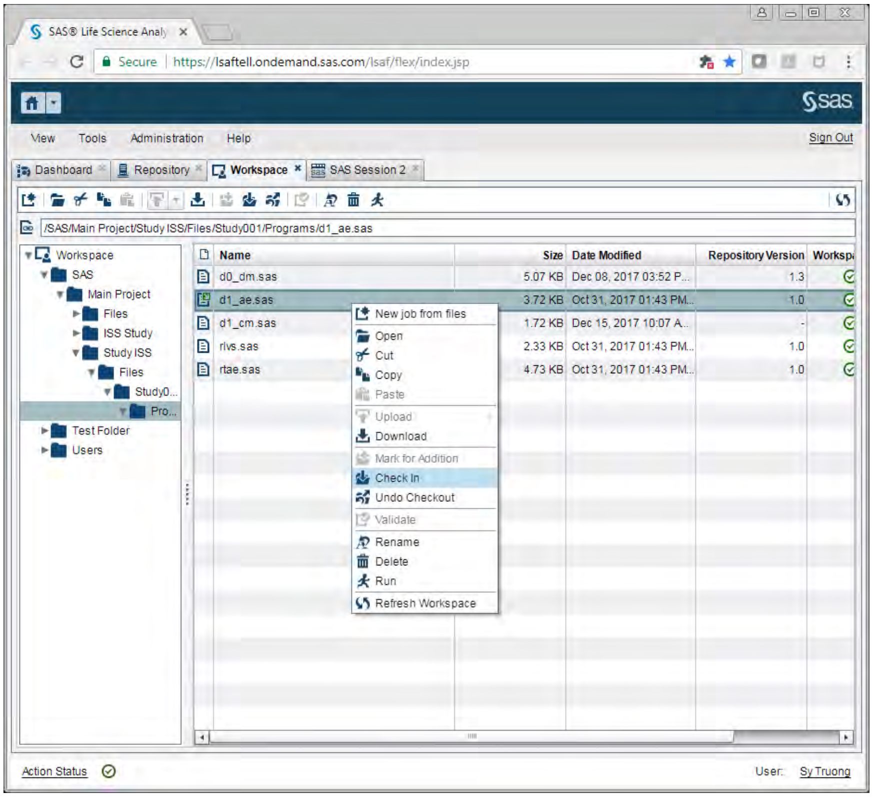


```
  else x_race=7;
  if sex='M' then sexn=1;
  else if sex='F' then sexn=2;
  if raceoth ne '' then x_raceot=raceoth;
run;

proc sort data=dm;
  by site randomno;

*** Debugging proc print ***;
proc print data = dm;
title "#1 Debug print of WORK.DM ";
run;

%*** Get study drug start from SDA dataset;
proc sort data=&rawdata..sda out=sda(keep=site randomno exdtc);
  by site randomno exdtc;
```

```
      real time           1.03 seconds
      cpu time            0.01 seconds


 17

```

7. Click on the “Save” icon to save the update.

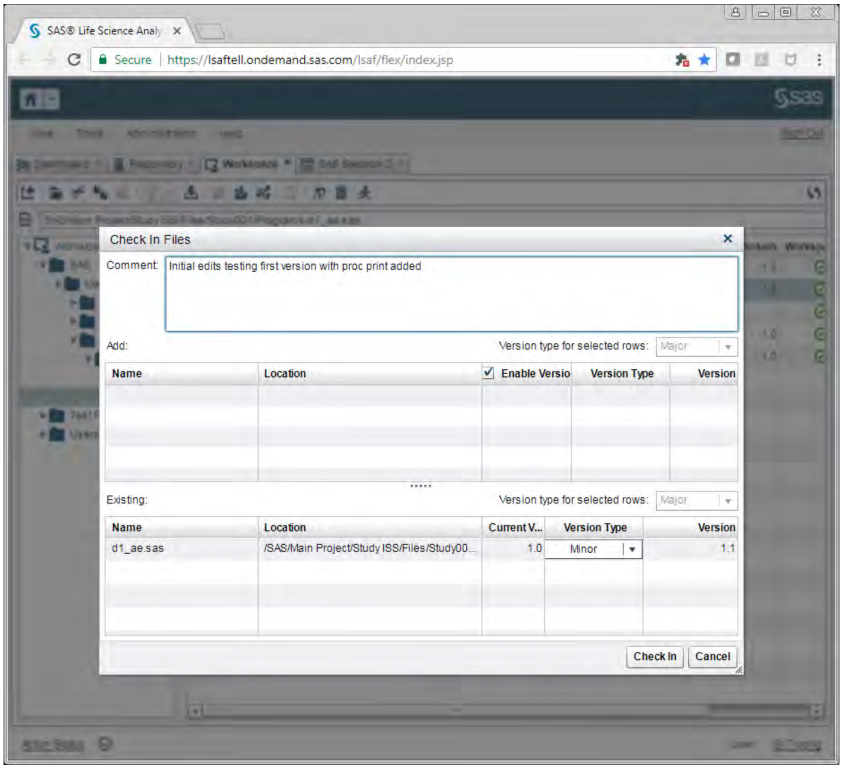
8. Right mouse click on the program in the Workspace and select "Check In"



/SAS/Main Project/Study ISS/Files/Study001/Programs/d1_ae.sas

| Name | Size | Date Modified | Repository Version | Workspa |
|---|---|---|---|---|
| d0_dm.sas | 5.07 KB | Dec 08, 2017 03:52 P... | 1.3 | ✓ |
| d1_ae.sas | 3.72 KB | Oct 31, 2017 01:43 PM... | 1.0 | ✓ |
| d1_cm.sas | 1.72 KB | Dec 15, 2017 10:07 A... | - | ✓ |
| rlvs.sas | 2.33 KB | Oct 31, 2017 01:43 PM... | 1.0 | ✓ |
| rtae.sas | 4.73 KB | Oct 31, 2017 01:43 PM... | 1.0 | ✓ |

9. Enter comments to document what has been checked in.



Comment: Initial edits testing first version with proc print added

Add:

Version type for selected rows: Major

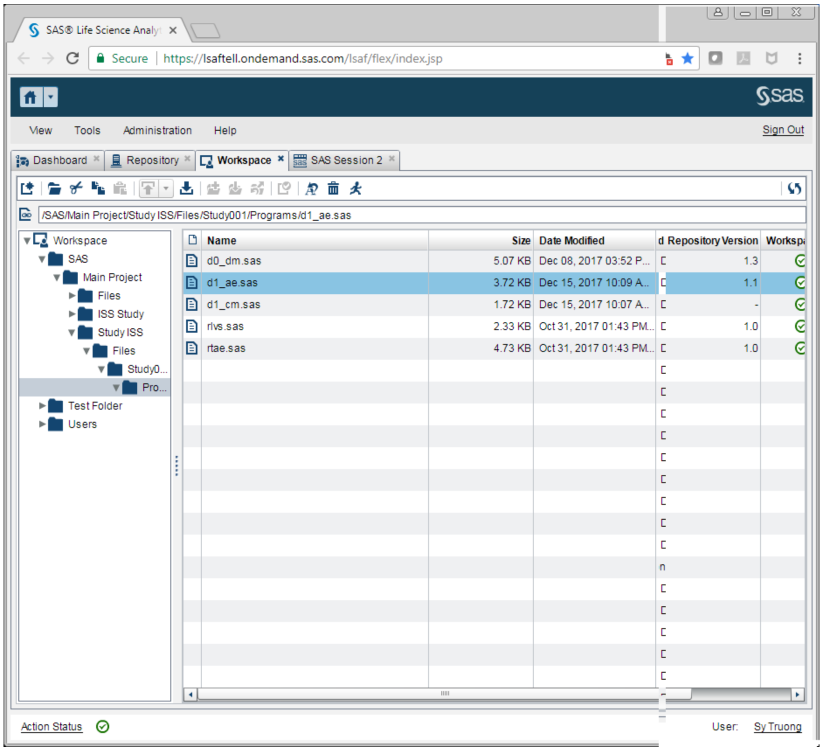
| Name | Location | ☑ Enable Versio | Version Type | Version |
|---|---|---|---|---|

Existing:

Version type for selected rows: Major

| Name | Location | Current V... | Version Type | Version |
|---|---|---|---|---|
| d1_ae.sas | /SAS/Main Project/Study ISS/Files/Study00... | 1.0 | Minor | 1.1 |

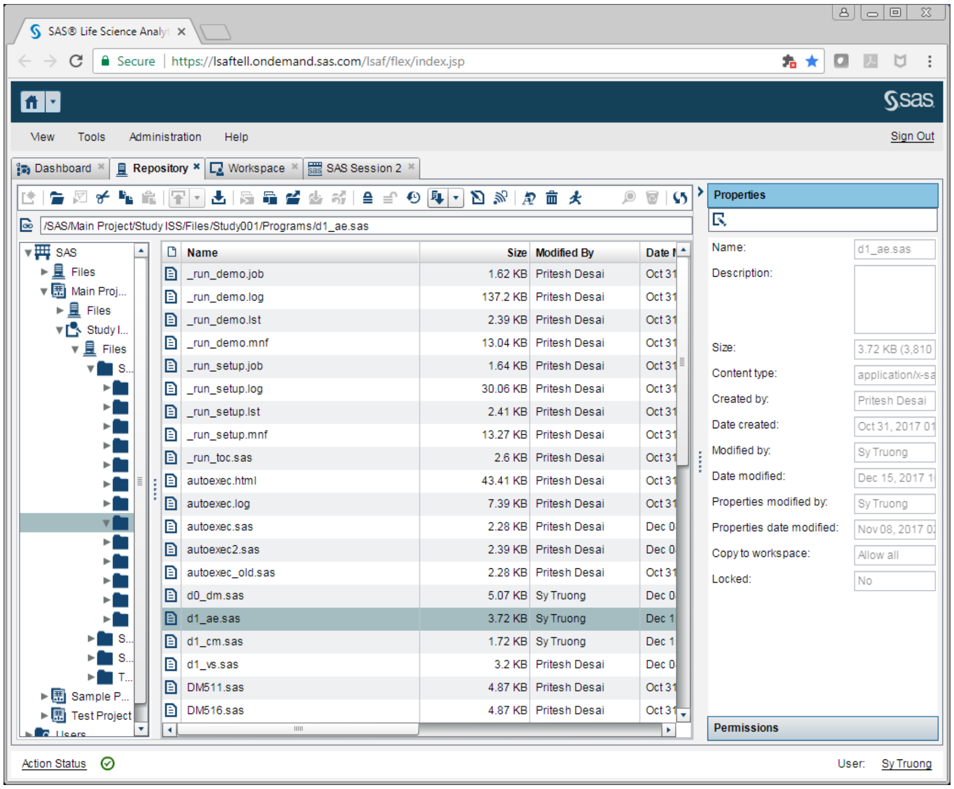
10. Verify that the icon no longer has the checked-out arrow.



/SAS/Main Project/Study ISS/Files/Study001/Programs/d1_ae.sas

| Name | Size | Date Modified | d | Repository Version | Workspa |
|---|---|---|---|---|---|
| d0_dm.sas | 5.07 KB | Dec 08, 2017 03:52 P... | C | 1.3 | ✓ |
| d1_ae.sas | 3.72 KB | Dec 15, 2017 10:09 A... | C | 1.1 | ✓ |
| d1_cm.sas | 1.72 KB | Dec 15, 2017 10:07 A... | C | - | ✓ |
| rlvs.sas | 2.33 KB | Oct 31, 2017 01:43 PM... | C | 1.0 | ✓ |
| rtae.sas | 4.73 KB | Oct 31, 2017 01:43 PM... | C | 1.0 | ✓ |

11. Navigate back to Repository to verify on properties that a new version has been modified.

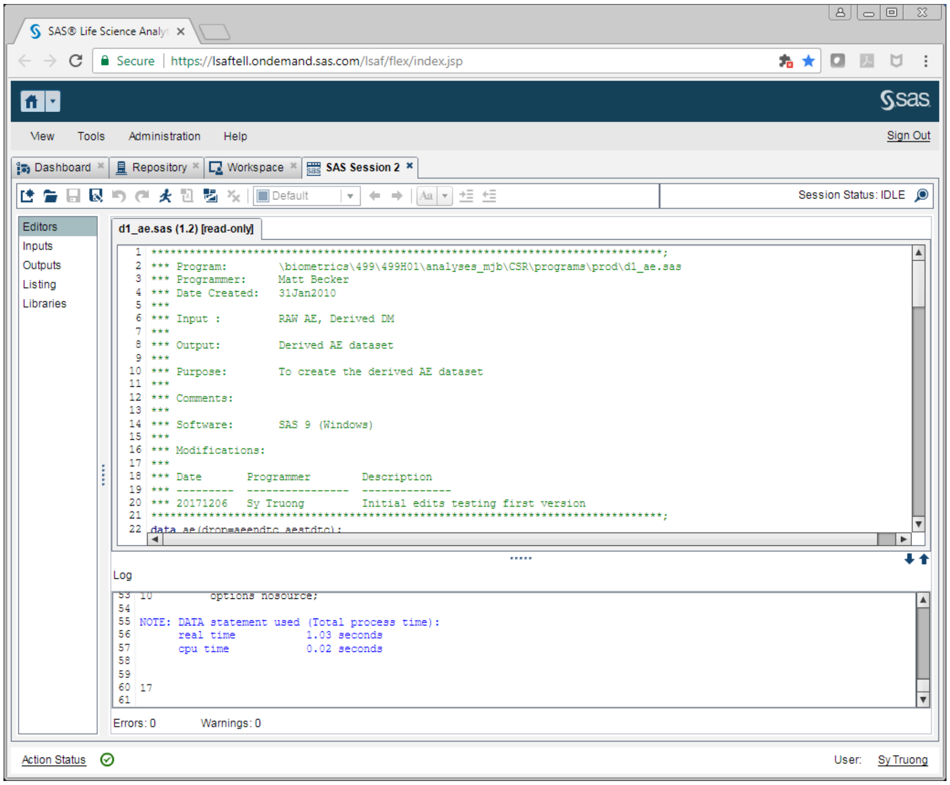


/SAS/Main Project/Study ISS/Files/Study001/Programs/d1_ae.sas

| Name | Size | Modified By | Date I |
|---|---|---|---|
| _run_demo.job | 1.62 KB | Pritesh Desai | Oct 31 |
| _run_demo.log | 137.2 KB | Pritesh Desai | Oct 31 |
| _run_demo.lst | 2.39 KB | Pritesh Desai | Oct 31 |
| _run_demo.mnf | 13.04 KB | Pritesh Desai | Oct 31 |
| _run_setup.job | 1.64 KB | Pritesh Desai | Oct 31 |
| _run_setup.log | 30.06 KB | Pritesh Desai | Oct 31 |
| _run_setup.lst | 2.41 KB | Pritesh Desai | Oct 31 |
| _run_setup.mnf | 13.27 KB | Pritesh Desai | Oct 31 |
| _run_toc.sas | 2.6 KB | Pritesh Desai | Oct 31 |
| autoexec.html | 43.41 KB | Pritesh Desai | Oct 31 |
| autoexec.log | 7.39 KB | Pritesh Desai | Oct 31 |
| autoexec.sas | 2.28 KB | Pritesh Desai | Dec 0 |
| autoexec2.sas | 2.39 KB | Pritesh Desai | Dec 0 |
| autoexec_old.sas | 2.28 KB | Pritesh Desai | Oct 31 |
| d0_dm.sas | 5.07 KB | Sy Truong | Dec 0 |
| d1_ae.sas | 3.72 KB | Sy Truong | Dec 1 |
| d1_cm.sas | 1.72 KB | Sy Truong | Dec 1 |
| d1_vs.sas | 3.2 KB | Pritesh Desai | Dec 0 |
| DM511.sas | 4.87 KB | Pritesh Desai | Oct 31 |
| DM516.sas | 4.87 KB | Pritesh Desai | Oct 31 |

Name: d1_ae.sas
Description:
Size: 3.72 KB (3,810
Content type: application/x-sa
Created by: Pritesh Desai
Date created: Oct 31, 2017 01
Modified by: Sy Truong
Date modified: Dec 15, 2017 1
Properties modified by: Sy Truong
Properties date modified: Nov 08, 2017 0
Copy to workspace: Allow all
Locked: No

12. Double click to confirm that the new version code is showing in the repository.



d1_ae.sas (1.2) [read-only]

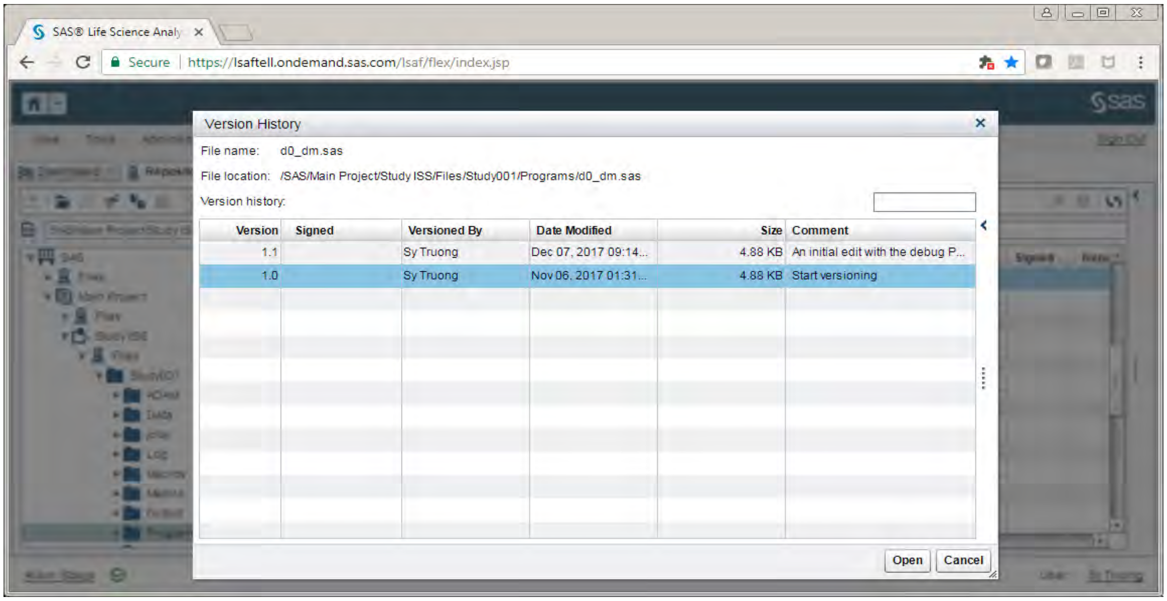
```
********************************************************************************;
*** Program:        \biometrics\499\499H01\analyses_mjb\CSR\programs\prod\d1_ae.sas
*** Programmer:     Matt Becker
*** Date Created:   31Jan2010
***
*** Input :         RAW AE, Derived DM
***
*** Output:         Derived AE dataset
***
*** Purpose:        To create the derived AE dataset
***
*** Comments:
***
*** Software:       SAS 9 (Windows)
***
*** Modifications:
***
*** Date       Programmer       Description
*** ---------  ---------------  -------------
*** 20171206   Sy Truong        Initial edits testing first version
********************************************************************************;
data ae(drop=aeendtc aestdtc);
```

Log

```
10         options nosource;

NOTE: DATA statement used (Total process time):
      real time           1.03 seconds
      cpu time            0.02 seconds


17

```

Errors: 0    Warnings: 0

13. Within the Work Space, scroll to the right and verify that the version is “1.1” and status is green.
14. Right mouse click on the program and select “Version History”.
15. From the history, click on version 1.0 and view the old version by clicking on the “Open” button.


SAS® Life Science Analy
Secure | https://lsaftell.ondemand.sas.com/lsaf/flex/index.jsp
sas
Version History
Open
Cancel

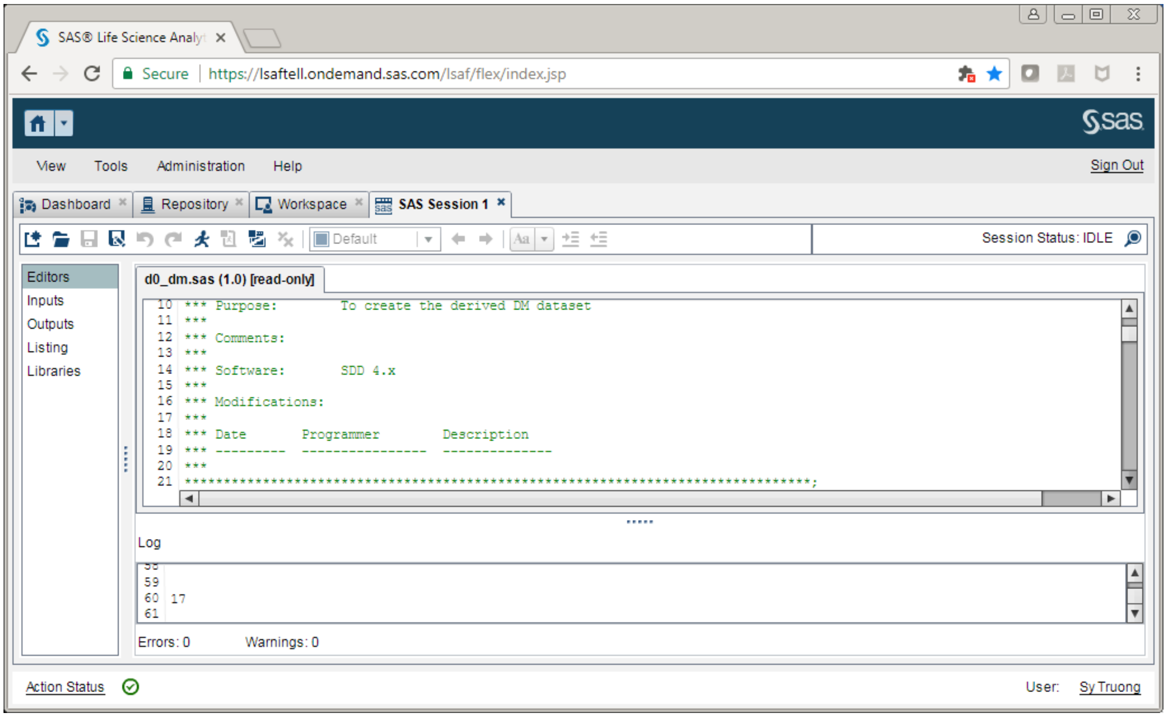
File name: d0_dm.sas

File location: /SAS/Main Project/Study ISS/Files/Study001/Programs/d0_dm.sas

Version history:

| Version | Signed | Versioned By | Date Modified | Size | Comment |
|---|---|---|---|---|---|
| 1.1 | | Sy Truong | Dec 07, 2017 09:14... | 4.88 KB | An initial edit with the debug P... |
| 1.0 | | Sy Truong | Nov 06, 2017 01:31... | 4.88 KB | Start versioning |

16. Verify in the read-only view that the program code is the old version.



Session Status: IDLE

d0_dm.sas (1.0) [read-only]

```
*** Purpose:        To create the derived DM dataset
***
*** Comments:
***
*** Software:       SDD 4.x
***
*** Modifications:
***
*** Date       Programmer        Description
*** ---------  ----------------  --------------
***
*********************************************************************************;
```

```
17
```

Errors: 0 Warnings: 0

User: Sy Truong

17. Repeat this for the new version.

SAS LSAF is a web based software which works well within a multi-user environment. However, it does not allow two different users from making updates to the same program at the same time. This is referred to as "branching" and "merging" in other tools. SAS LSAF will lock the program when it is checked out, so subsequent users will not be able to check it out at the same time. This can allow for a straight forward method of managing changes among different team members. Depending on the work flow, it can be a limitation since it does not support simultaneous editing of a program by multiple users.

## CONFIGURATION AND ADMINISTRATION

The selection of the change control tool can be affected by how the tools are configured. This will require clear communication and collaboration between the Business administrator or SAS programmer and the IT system administrator. The SAS programmer provides clear and specific business requirements pertaining to the SAS program change control tool implementation. These requirements will help the IT system administrator formulate the recommended configuration options. Some key requirements that will contribute to this decision include:

1. **Folder Level –** At the lowest level, the folder used for the change control repository can be applied to the folder that stores only SAS programs for that project. A higher level can include the SAS programs and other folders for input data and output reports. Even a higher level than that can include all global macro and utilities which the programs will use. The repository can be applied at these different levels, but a decision is made to select one that works best for the SAS programming team.

2. **Repository Deletion –** The database that the software uses to store all the versions of related files can be stored locally in a folder that the user can easily view and potentially delete. If it is required that the SAS programmer cannot view or delete this repository database, then this will influence how the tool will be configured.
3. **Local vs Web Server –** The tool can be configured to upload the repository over a web server so that the user does not have direct access to the repository database files. The other option would be that all repository related files be on the same file system that the user has direct access to. Another related consideration is if the user needs to download the files to their local laptop as compared to viewing it on the file system where it is originally stored.
4. **Branch and Merge –** A simplistic approach is to have a single version with all updates needing to merge back to the final version. The terminology used is that the "trunk" is the final version and updates are made into "branches" of the "tree". The SAS programmer must then decide that there can only one branch at a time, or can there be multiple branches that creates challenges when merging these multiple branches back to the final "trunk".

Once the above requirements are clearly communicated between the SAS programmer to the IT system administrator, this will inform the type of configuration and tool selection.

### FEATURE RICH

It very unlikely that one tool can match perfectly with all the specific requirements of any working group. The goal then for this evaluation is to identify distinct functional requirements that would add value and then try to grade each tool upon these key areas. In our evaluation, these are some of those features:

| Feature | Description |
|---|---|
| **Setup Repository** | Creating a new Repository of SAS programs |
| **Checkout** | Checking a program out repository |
| **Commit** | Commit a new program change |
| **Change Description** | Capture a description of the changes to a program |
| **History** | Review history of changes to a program |
| **Difference** | Compare current with previous version for differences |
| **Open Editors** | Open programs into editors for easy edits |
| **Roll Back** | Review and roll back to older versions |
| **Locking** | Multiple users locking |
| **Merging** | Multiple user merging changes |
| **SAS Integration** | How well it integrates current SAS process |

Each one of these features will be tested with sample SAS programs and graded by a few SAS programmers. The scoring is between 0-4; with a value of "4" being the highest score, going down to "0" for failing or not having the feature. The grades will then be summarized among testers to identify the highest score for the best tool. The list is not comprehensive and the method may not be completely scientific. However, it is sufficient for the purpose of identify the tool that best fits our requirements.

The following figure summarizes the findings in the various categories. The lighter shades indicate the average of the multiple evaluations.

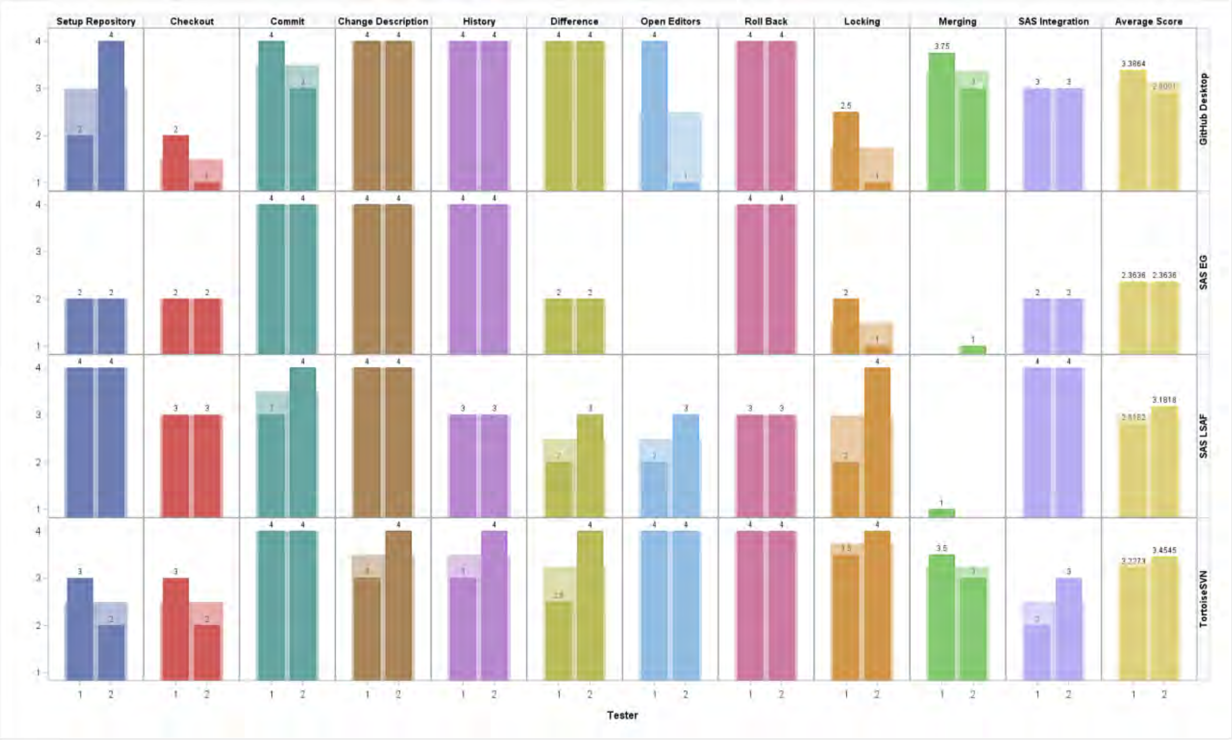


*Figure 1 – Evaluation Summary*

The average score for all the criteria is summarized on the right. Although SAS EG had many user-friendly interfaces for SAS programmers, the change control features are relatively new and is not as feature rich as compared to GitHub Desktop and TurtoiseSVN. These non-SAS tools have been around for many years and its maturity in this arena shines. SAS LSAF is more sophisticated and scored well. It has many other features outside of change management which was not included in this evaluation.

## CONCLUSION

The ability to be able to manage SAS programs as they are being updated during its development, especially in a multi user environment with hundreds of SAS scripts is essential. Since SAS programs contain the main intellectual property of any data analysis; this capability becomes a business necessity. In some regulated industries, managing change control can also be a legal requirement. At a minimum, all the tools evaluated in this paper contain sufficient capabilities in retaining old versions of a SAS program and can be used to maintain changes to the code along with capturing comments explaining the updates as the programs as they are being developed. The areas of distinctions which sets the tools apart are how it integrates with SAS programming and supporting multi-users branching and merging capabilities. The user experience and user friendliness can be personal preference but are also very different among the tools.  In summary, if the environment does not require multi-user branching and a SAS program locking can be applied, then SAS Enterprise is a very cost-effective way of deploying a user-friendly tool for users to use.  This is with the caveat of adhering the workflow to using EG project files as compared to SAS programs stored as scripts on a file system.  If the SAS programming is performed in a regulated environment such as the pharmaceutical industry, then SAS LSAF has advantages in the features it provides that are industry specific.  Otherwise, the GitHub Desktop and TortoiseSVN both provide a good solution for multi-user support including capabilities of managing multiple branches and merge features for managing change control for SAS programs as compared to locking, which SAS EG and SAS LSAF is designed for.   The final decision for selecting a tool is driven by your specific requirements and work flow within a specific environment.  With the options presented in the paper, there is a good option to meet most SAS programming groups needing to manage changes to their SAS programs.

## CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

**Authors**: Sy Truong and Pradheep Raman
**Location**: Pharmacyclics, Inc.