PharmaSUG China 2021 - Paper AD-14

Development of Integrated Statistical Computing Platform (iSCP) Based on Intelligent Apps

Zhiping Yan, Dizal

ABSTRACT

With new biotech companies are springing up around the world, only fast movers in the pharmaceutical industry can win the fierce competition and dominate the market. SAS macros have been widely used by statistical programming team to improve work efficiency and help speed products to market. However, boring repeated unproductive manual work is still needed. Not to mention that some tasks cannot be accomplished via SAS macros. To further increase work efficiency, the Dizal Integrated Statistical Computing Platform (Dizal-iSCP) which is a seamless and interactive platform is built to consolidate SASEG, Linux SAS, Python, and other software like Adobe PDF, Microsoft Word and Excel, etc.

Dizal-iSCP consists of mutually independent applications and each application is designed for a specific task such as SASEG automation, batch run, CRF annotation, and generation of SAS program. These applications are classified and placed into five tabs as per clinical data analysis flow. With good flexibility and extensibility, as many applications as possible can be developed and added to this platform at any time. This allows us to use the platform while developing and perfecting it. In summary, it extremely maximizes the efficiency for statistical programmers. This paper will present how to design and construct such a kind platform.

INTRODUCTION

With the competition getting more and more fierce in the pharmaceutical industry, drug companies are striving to bring drugs to market in the shortest possible time. As a key role player in clinical development, the statistical programming team should try to do more with less to improve work efficiency and assure a constant high level of quality.

Repeated use of validated SAS macros for similar and routine tasks can significantly increase programming efficiencies and improve programming qualities. However, it is not easy to fulfil tasks like SDTM CRF annotation and text mining using SAS. Additionally, switching between different tools and study folders is time-wasting. Therefore, it is urgent to build an integrated Statistical Computing Platform that can maximize efficiency by integrating all types of software.

Developing a platform is time-consuming, resource-consuming, and will cost a lot of time and money. It is hard for a new biotech company to complete and put a platform into service within a short time. However, these difficulties can be overcome by breaking the whole platform into mutually independent units. At Dizal, an Integrated Statistical Computing Platform (hereinafter referred to as Dzial-iSCP) allowing users to access standalone applications is built to provide a scalable, portable, and extensible operating environment. Plus, it can significantly reduce compliance risk and avoid audit findings by meeting the below requirements:

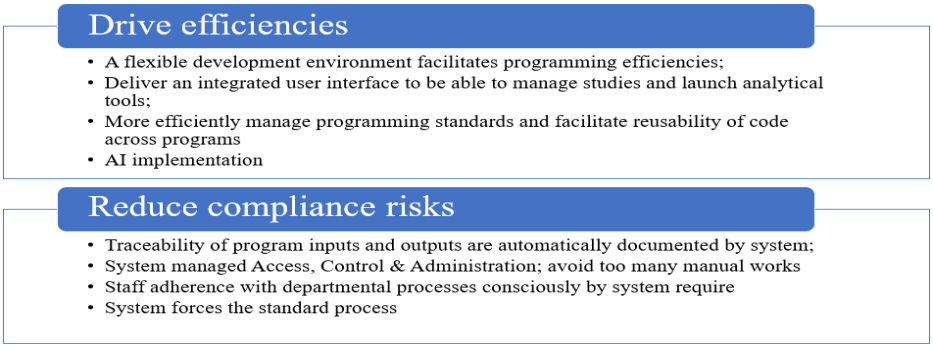


Figure 1. Core functional requirements of Dizal-iSCP

STRUCTURE OF DIZAL-ISCP

Dzial-iSCP developed using Python is essentially an ecosystem which consists of three levels of contents. The lowest or third level is the foundation of whole platform and includes the IT architecture like SAS server, storage environment and file system. The second level contains metadata, data, programs, output, tools developed using Python or SAS, software programs and various types of documents. The highest or first level is the view level through which users can get access to all kinds of components by interacting with the elements in user interface.

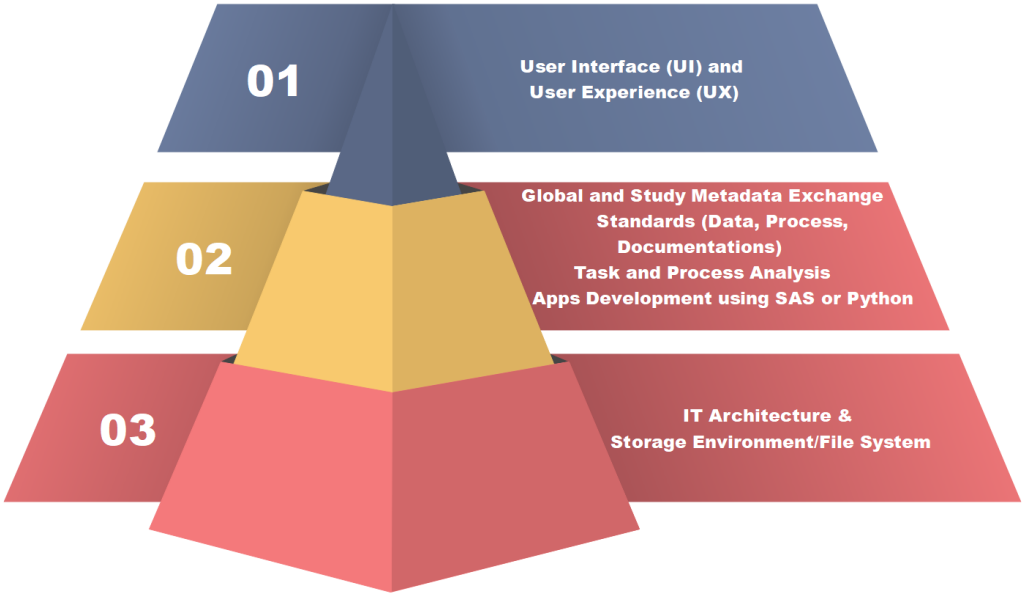


Figure 2. Structure of Dizal-iSCP

USER INTERFACE OF DIZAL-ISCP

The user interface of Dizal-iSCP consists of a home tab and five task tabs — Basic, EDC, SDTM, ADaM, and Reporting — organized per clinical data analysis flow.

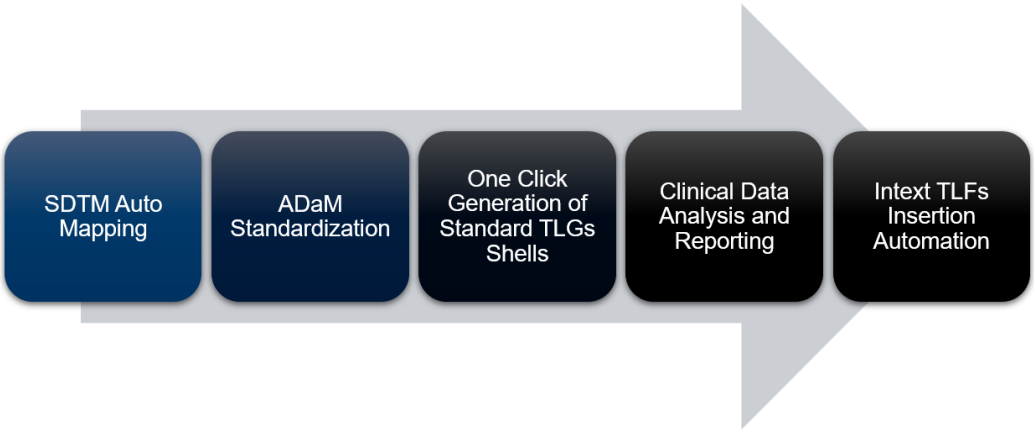


Figure 3. Clinical data analysis flow

Besides shortcuts providing links to task tabs or ‘SASEG Autoexec’ application, the home tab also provides four combo boxes for users to choose their desired study. Only folders to which users have access will be displayed in combo boxes. The retrieving folder list is time-consuming especially when a user has access to dozens of or hundreds of study folders. To reduce startup time, the button 'Refresh Access' is added to extract and put the folder list in an external text file. The button is expected to be clicked only in the situation where the user's access permissions are modified.

Figure 4. Home tab of user interface of Dizal-iSCP

The other five task tabs are made up of shortcut icons to standalone applications. Figure 5 shows how the applications are organized in task tabs. By double-clicking on icons like the one under 'Quick Access to Open SAS EG' in Figure 4, corresponding applications can be launched customizing for the study as selected through combo boxes (see Figure 6 for demo). A warning message will be prompted for any operations on shortcut icons if the study is not chosen. This kind of error and warning check exists throughout the platform to make it easier for users to use the platform correctly.

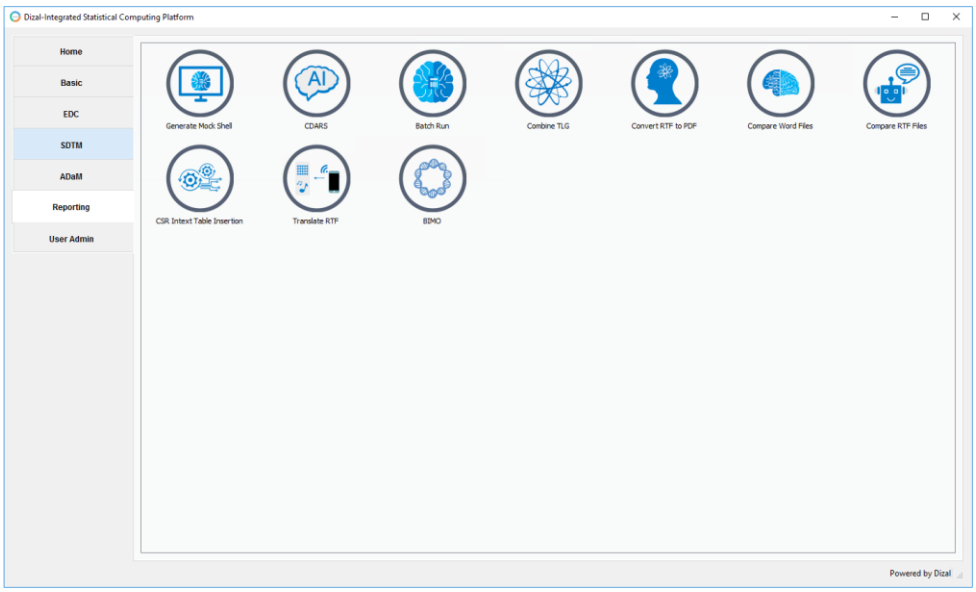


Figure 5. How applications are organized in task tabs

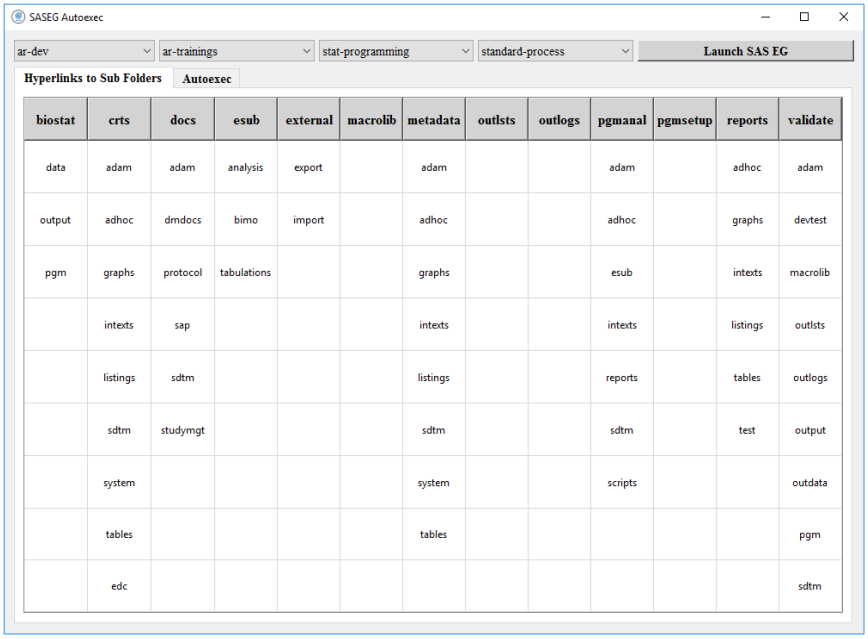
Through SASPy, Paramiko, Subprocess and os python packages, software like SASEG, Linux SAS, Adobe PDF, Microsoft Office and Internet explorer can be started by manipulating applications inserted into above five task tabs.

EXAMPLE APPLICATION 1 — SASEG AUTOEXEC

Folder tree in Servers window of SAS EG can allow users to access specific study folders. However, clicking and expanding folder tree layer by layer is annoying and time-wasting. Plus, autoexec.sas must be submitted before any other files can be processed after desired study folders are selected. Additionally, navigating File Expore in windows

is also a task that no one would like to do. To enable SAS EG have out-of-the-box feature and provide shortcuts to study folders, SASEG Autoexec application as showed by Figure 6 was built.

Each application will be opened initializing based on folders selected in home tab of Dizal-iSCP. Below figure shows what users will see after traning folders are selected and shortcut icon under 'Quick Access to Open SAS EG' in home tab (see Figure 4) is clicked. Folders in top-left four combo boxes are populated automatically and only child folders under /ar-dev/ar-traning/stat-programming/standard-process are extracted and arranged into table showed in Hyperlinks to Sub Folders tab. First level of sub folders (e.g. biostats, crts and docs) are displayed in table header. Child folders of these first-level-sub-folders are placed in table cells. For example, data, output and pgm are sub folders of biostat (see leftmost column). By clicking on cell data, folder /ar-dev/ar-traning/stat-programming/standard-process/biostat/data will be opened. To facilitate users check libname or filename statement, code from autoexec.sas will be presented in Autoexec tab.



biostat	crts	docs	esub	external	macrolib	metadata	outlsts	outlogs	pgmanal	pgmsetup	reports	validate
data	adam	adam	analysis	export		adam			adam		adhoc	adam
output	adhoc	dmdocs	bimo	import		adhoc			adhoc		graphs	devtest
pgm	graphs	protocol	tabulations			graphs			esub		intexts	macrolib
	intexts	sap				intexts			intexts		listings	outlsts
	listings	sdtm				listings			reports		tables	outlogs
	sdtm	studymgt				sdtm			sdtm		test	output
	system					system			scripts			outdata
	tables					tables						pgm
	edc											sdtm

Figure 6. Initialized SAS EG Autoexec application

By clicking on button 'Launch SAS EG' as showed in Figure 6, a SAS EG window will be opened and selected folders (see combo boxes in Figure 6 and red squares in Figure 7) will be clicked automatically to expand folder tree in Servers window. While at the same time, autoexec.sas will be submitted to initialize SAS session settings.

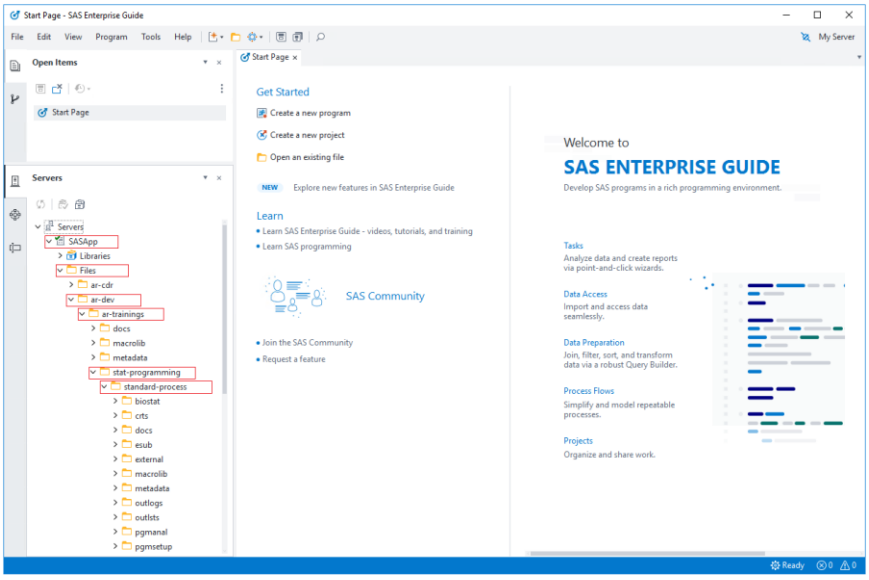


Figure 7. Automation of SAS EG

Automation of SAS EG can be accomplished with help of pywinauto[1] python package. Here is example code showing how to automate folder tree item in Servers window.

```
from pywinauto.application import Application
from pywinauto.keyboard import send_keys

app = Application(backend='uia').start(r'F:\Program
Files\SASEnterpriseGuide\8\SEGuide.exe')
app.Dialog.wait("exists ready", timeout=15, retry_interval=3)
dlg = app.Dialog.child_window(title="Servers", auto_id="Servers",
control_type="Pane")
dlg1 = dlg.child_window(title="Servers", control_type="TreeItem")

dlg = dlg1.child_window(title='SASApp', control_type="TreeItem")
dlg.ensure_visible()
dlg.click_input(button='left', double=True)
send_keys("{VK_RETURN}")
```

To disable UserWarning message, below code must be placed at the beginning before importing pywinauto.

```
import sys
import warnings
warnings.simplefilter("ignore", UserWarning)
sys.coinit_flags = 2
```

Execution of autoexce code can be done via option settings of SAS EG following below steps[2].

1. Select **Tools** -> **Options** from top menu bar to open **Options** window
2. Select **SAS Programs** and check the box for **Submit SAS code when server is connected**
3. To the right of this checkbox, click **Edit** button and enter code to include user-specific external SAS program file

Look at Figure 8, by putting study-specific autoexce file in above user-specific external SAS program file, we can achieve the goal to execute both user and study specific autoexec file. When SAS EG is launched, /root/ar-share-area/startup/startup-zhiping yan.sas will be executed. Thus another include statement in user-specific external file (startup-zhiping yan.sas) will be submitted and study specific autoexec file (/root/ar-dev/ar-trainings/stat-programming/standard-process/pgmsetup/autoexec.sas) can be executed. The problem is that user-specific external file must be modified each time SAS EG is launched. This can be completed by Python. Users only need to set the options one time. From users' perspective, SAS EG is opened to use.

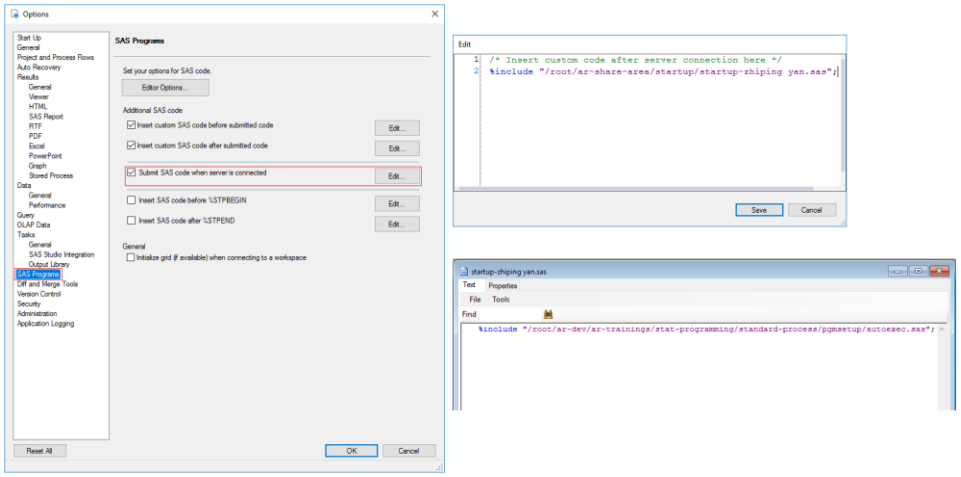


Figure 8. SAS EG option setting for automatic execution of autoexec.sas

EXAMPLE APPLICATION 2 — BATCH RUN

Common practice for batch run is to generate bash shell script using SAS macros and then excute sh file in Linux through SSH tool like MobaXterm. To input path of sh file or standalone SAS program file into MobaXterm is not an enjoyable thing. What's worse, it is not convenient to decide order of SAS program files in sh files. To simplify the process, Batch Run application was developed.

All folders containing program files will be listed in rightmost combo box when Batch Run application is launched. By selecting a folder in rightmost combo box, all SAS program files within selected folder (e.g. pgmanal/reports) will be displayed in left list box. With help of 'Add' and 'Remove' buttons, desired SAS program files can be added into middle list. 'Move Up' and 'Move Down' buttons in middle part can help change order of SAS program files. Button 'Generate sh Script' can place SAS program file from middle list in sh file. Generated sh file (dev-reports.sh) will be added into right list box.

'Move Up' and 'Move Down' buttons in right part can modify order of sh files. By clicking on button 'Batch Run', sh files will be executed one by one. If folder path in sh file is not consistent with folders from combo boxes, warning message will be prompted. After clicking on button 'Check Log', log files will be opened to check if error or warning message exist. An excel file will be generated to place log check results and opened automatically. By selecting sh files (e.g. dev-adam.sh and dev-reports.sh) in right list and clicking on button 'Generate Merged sh Script', a new sh file (e.g. dev-adam-reports.sh) will be generated.

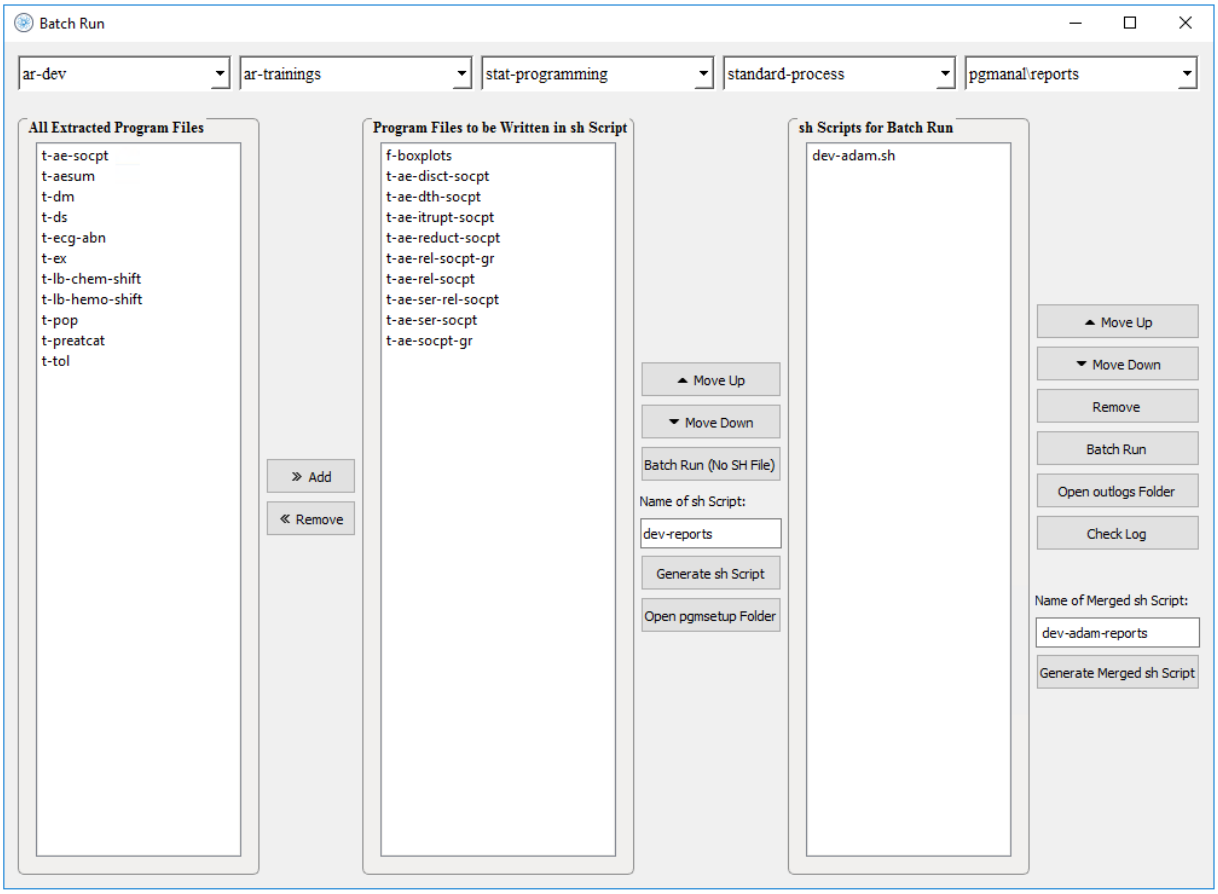


Figure 9. Batch Run application

Besides sh files, the standalone SAS program files can also be executed with help of button 'Batch Run (No SH File)'. After execution, SAS program files for which log files containing error or warning message will be highlighted in red.

Paramiko[3] package can be used to execute Linux Command line.

```
import paramiko
ssh = paramiko.SSHClient()
ssh.connect("linux sas", port=portnum, username=usrname, password=pwd)
cmd = '/root/ar-dev/ar-trainings/stat-programming/standard-
```

```
process/pgmanal/test.sas'
ssh.exec_command('chmod 755 ' + cmd)
stdin, stdout, stderr = ssh.exec_command(cmd)
ssh.close()
```

While SASPy[4] package can help submit SAS code if needed.

```
import saspy
sas = saspy.SASsession(cfgname='iomcom')
autoexec = '/root/ar-dev/ar-trainings/stat-programming/standard-
process/pgmsetup/test.sas'
ps = sas.submit('%include "' + autoexec + '"; ')
sas.endsas()
```

To successfully connect Paramiko or SASPy with the platform, configuration support from IT team is required. Those highlighted in yellow in above sample code should be updated per real computer setting.

CONCLUSION

Besides above two examples, we also have a dozen other applications for tasks like combination of TLG, CRF annotation, Comparing RTF files and Converting RTF into PDF. More applications will be developed and added into Dizal-iSCP to further increase effectiveness.

The platform allows users to access all software and manipulate almost all types of files. Repeated time-wasting manual work can be done by clicking serval buttons. It also allows users to use the platform while designing and developing new applications. All these properties demonstrate that this kind of platform is worth popularizing and utilizing, especially for start-up companies which have limited resources and many urgent tasks.

REFERENCES

[1] pywinauto package https://pywinauto.readthedocs.io/en/latest/getting_started.html
[2] Leonid Batkhan, Autoexecs—the SAS Enterprise Guide advantage
[3] Paramiko package http://www.paramiko.org/
[4] SASPy package https://sassoftware.github.io/saspy/

ACKNOWLEDGMENTS

I would like to thank Zongming Guo and Wei Zhang for their dedicated efforts in installing and setting different software. Without their excellent work, we cannot establish such kind of platform.

Thanks to all statistical programming team members at Dizal for their advice and feedback so that our platform can be improved continuously.

CONTACT INFORMATION

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

Name: Zhiping Yan
Enterprise: Dizal
E-mail: zhiping.yan@dizalpharma.com