Paper OS06

Organizational Considerations When Replacing or Adding a New Software Language

Brian Varney, Experis, Kalamazoo, Michigan USA

ABSTRACT

Replacing or adding a new software language for your organization's users can add a lot of value and productivity to your team(s). There are, however, many factors and considerations to consider avoiding chaos and hidden costs.

INTRODUCTION

This paper intends to outline factors to consider when replacing or adding new software for your organization's users. Although this could pertain to several different types of software packages, the motivation for this paper is geared towards development/programming software such as SAS®, R, Python, etc. Specifically, this paper will focus on SAS and R due to the prevalence of these two software packages in clinical trials programming today. Due to the trends and today's landscape, this paper will be written from the aspect of adding R to a SAS environment.

IT ARCHITECTURE, DEPLOYMENT / ADMINISTRATION, AND VALIDATION CONSIDERATIONS

To really evaluate the factors surrounding the implementation of a new software package, it is important to define the necessary supporting architecture and topology. This also includes resource requirements for running the software package on a server or user desktop. There are typically additional costs and talent requirements as complexity is added to an environment.

ARCHITECTURE / TOPOLOGY

SAS and R can both be deployed in multiple server architectures. In general, SAS leverages mostly I/O to and from disk during processing while R leverages mostly memory for its processing. When sizing servers, this must be considered. Although it is important for SAS to have enough memory per CPU, most traditional SAS processes rely heavily on having disks with high performant I/O. SAS Viya and SAS Visual Analytics does host more processes and data in memory and that will have to be factored into hardware requirements for those products.

SAS 9.4 architecture typically involves a SAS Metadata, Compute, and Web server.

Posit has products that can be centralized to a server and are organized under a bundle called Posit Team. Posit Team includes Workbench (web based Posit RStudio IDE and compute), Connect (publish and share results), and Package Manager (manage repositories of packages).

Both SAS and Posit have cloud versions of their software that individuals can use for educational purposes or organizations can license.

DEPLOYMENT & ADMINISTRATION

Before implementing a new software package, it is important to understand the deployment & administrative needs required for everything to run smoothly. Any training or additional talent requirements will need to be evaluated. While there are some similarities in administering SAS and R, significant training is necessary to switch from one to the other.

Both SAS and Posit have specific training programs for deployment and administration. For each software, it is important to leverage resources that already have this experience and training.

VALIDATION

If your organization is regulated, such as the case with pharmaceuticals, it is important to have validation planning and documentation requirements. The pharmaceutical industry is regulated by the Food and Drug Administration, FDA, and there are strict controls on results submitted to the FDA. Specifically, software used to produce results needs to be validated in the environment it is running in, whether it be a server or desktop.

Guidance for Industry – Computerized Systems Used in Clinical Trials
https://www.fda.gov/inspections-compliance-enforcement-and-criminal-investigations/fda-bioresearch-monitoring-information/guidance-industry-computerized-systems-used-clinical-trials

SAS software quality is governed by SAS Institute. They have a team that monitors the software development, testing, and documentation and signs off on it prior to release to customers. When SAS releases a new version or a maintenance release, they provide customers with updated SAS Software Depots that can be downloaded upon request from their SAS account manager. SAS can also release hotfixes to correct bugs in their software and those

are also all controlled centrally by SAS Institute. This encompasses SAS integrated development environments (IDEs) and all SAS products and modules.

R Software and packages are available from the Comprehensive R Archive Network (CRAN). CRAN is a repository for the R programming language and includes the source and compiled versions of R along with thousands of packages. Base R software is developed and maintained by the R core team, a non-profit organization that provides ongoing support for the R language. R packages can be developed by anyone, but CRAN scrutinizes and tests all packages that are submitted. They check the quality of the documentation, code, and test coverage that the developer has included in their package. The most widely used IDE for R programming today is RStudio. RStudio is developed and maintained by Posit.

The FDA requires that organizations have a documented plan for validation of software used for submissions. These plans should have testing that covers Installation Qualification (IQ), Operational Qualification (OQ), and Performance Qualification (PQ).

Installation Qualification (IQ) demonstrates that the critical components of the software are present and installed. Often, the sizes of the files in the software installation are checked to ensure they are present and the expected size.

Operational Qualification (OQ) tests that the software components are working as intended. For example, a SAS OQ routine may test standard procedure calls for each module, while an R OQ test would test standard functions calls within each package.

Performance Qualification (PQ) tests that the software components will meet the unique needs of your organization. If there are certain types of analyses that are commonly carried out, there should be validation programs that can be run to ensure that the actual values are equal to the expected values.

SAS includes the IQ and OQ routines along with SAS software. Since the software is developed by one organization, it is possible to bundle them in. Once SAS is installed, these routines are run manually on the server or desktop.

Validating an R environment can be challenging if there are a variety of packages being used. Your organization can develop these IQ, OQ, and PQ internally or purchase a set of packages that include these validation scripts for you to use. Another recently developed alternative is a package called thevalidatoR. This is a package that you can access and read about at: https://github.com/insightsengineering/thevalidatoR/tree/main. Using the thevalidatoR package allows you to generate a validation report. There are examples of these validation reports at the link provided earlier in this paragraph. There are other packages described on the Pharmaverse website (https://pharmaverse.org/) related to package validation such as covtracer, valtools, riskmetric, and riskassessment. These and thevalidatoR are in the Developers section of the Pharmaverse website.

Any time there are changes to your server or software environments, the validation plans need to be carried out and documented to ensure software is working as expected and to show during audits.

WORKFORCE AND DEVELOPMENT CONSIDERATIONS

While the impacts to the IT department are important, so are the impacts on the developers using the software. Evaluating and planning for these impacts can make for a much less stressful adoption.

SOFTWARE ADOPTION RESISTANCE

There may be a significant portion of your workforce that does not want to learn a new development language. This could result in team members leaving for another company or retiring. Whatever the case, adopting another software can cause resistance and chaos. To minimize this resistance, it is best to have a well communicated plan for training and ensure that you will allow users time to take this training as part of their work week.

LEARNING CURVE

In the example of a SAS user learning R, it will require a change in thinking. SAS is a 4th generation language, and many consider R to be a 3rd generation language. If a user is used to solving modularity and automation using the SAS Macro language, it will take some time to get used to writing R functions. SAS does have the ability to allow users to develop user defined functions, however, SAS Macro is the standard approach used for code modularization in the pharmaceutical industry. SAS Macro is just text substitution prior to compile time and there is not any real equivalent in R. SAS users will have to learn and get used to the more stringent scope of objects used in R functions.

TRAINING

How will the developers using the software learn a new language while keeping up with their normal work? Which

method of training delivery is best for your user community?

- Online self-guided training
- In person instructor led training
- Online instructor led training.
- Train the trainer

Different users excel with different learning styles. A major challenge is for users to find enough time for training while keeping up with their normal full-time job. It is important for management to make it possible for the users to carve out a certain number of hours each week for training.

SUPPORT

The software users and administrators will both need support to have a healthy development environment. Below is a list of things to consider when evaluating support for a software product.

- What is the quality and completeness of the documentation?
- Is there support available and is it centralized or decentralized?
- If a fix or enhancement is needed, is there a process to request it and get it done?
- How much internal talent does your organization have with the software?

SAS has a centralized support desk. Regardless of which module you are using, you can call one number and get routed to the appropriate support resource. If you have a production problem after hours, you can get routed to a support desk in another part of the world.

If you are purchasing products from another company such as Posit, there will be support available for you to use. Otherwise, you must either rely on interaction with the author of the package or using support sites such as Posit, Hadley Wickham's sites, and Stack Overflow.

PUBLISHING

The results that are needed from a software package need to be considered. For example, if your deliverables require bookmarked PDF documents, you will need to make sure that additional development is not required to achieve those results.

SAS offers the Output Delivery System (ODS) which handles outputting tables, listings, and figures in a variety of formats such as HTML, PDF, RTF, Word, Excel, etc. SAS ODS works with SAS reporting procedures and has been in place and constantly refined for over 20 years.

R is catching up to SAS in reporting and publishing. R packages are currently being developed to address the reporting needs of the pharmaceutical industry. There is an initiative called the "Pharmaverse" which involves developers from different organizations collaborating to build packages and products to address these reporting needs, CDISC data set creation, and other areas necessary for clinical trials submissions. For more information on the Pharmaverse, please visit https://pharmaverse.org/.

R also has a package called Shiny which allows developers to create custom interactive applications in R much more easily than SAS.

VERSION CONTROL FOR CODE

For results to be reproducible, version control software will need to be considered for project code. SAS and R both work with version control software. Open-source development projects often require collaboration by people from different organizations, therefore, use of Git and Github is becoming more prevalent. SAS and R both have Git integrated into their IDEs.

Traditionally, clinical trials programmers have worked with centralized version control packages such as Apache Subversion (SVN). These are best to use when one person is working on a program at a time. They can essentially check out a program from the centralized code base, make their changes, and then check it back in.

Open-source development projects can often require multiple programmers working in parallel. This is where distributed version control systems such as Git are best.

There is a significant learning curve in switching from using a centralized version control system to a distributed version control system. I feel it is best to use centralized version control systems for clinical trials programming and

distributed version control systems for projects such as package development or projects that require many programmers working in parallel on the same set of files.

CURRENT LEGACY PROGRAMS AND PROCESSES

If you need to match results from legacy studies, there could be rounding and/or computational differences to code around or explain. Typically, neither software is incorrect, they just use different options or default computational algorithms. There are groups such as CAMIS (Comparing Analysis Method Implementations in Software), that are researching and documenting these differences. Please see https://psiaims.github.io/CAMIS/about.html for more information.

If you have completed studies for which you need to regenerate results, it is good to keep at least one desktop license around for this purpose.

AGILITY

How easy is it to obtain and install additional modules for the software. For some software packages, it is a cumbersome and lengthy process. For others, one just needs to download and install.

SAS has a very controlled and cumbersome process for adding modules to an existing license so users can try them out or just begin using them.

With R, a user can just download a package to their desktop R installation and begin using it. However, to use it for production work, IT will have to add it to the R server environment production package repository and ensure it is validated.

COST

LICENCING

The licensing costs for software packages are important to evaluate. Some software packages are free to install and use on the desktop but once they are installed on a server or in the cloud for a larger user base, they require licensing fees.

SAS Software licenses are primarily based on the SAS products involved and # of CPU cores. The more powerful the servers that are running SAS, the higher the annual license fees. There are also license fees to run SAS on the desktop.

R and RStudio can be downloaded and used for free. Products that support an R deployment from the Posit Team bundle require license fees and are primarily based on the products involved and the number of users.

There are companies that offer a set of validated R packages for a fee. Instead of an organization needing to create validation routines for the packages they are using for submission work, they can just pay to acquire them. This would be an additional cost on top of the Posit Team products software bundle.

CONCLUSION

Replacing or adding a new software package in your environment is not a decision to take lightly. There are many costs that are not obvious and can impact your users and organizations significantly. Details that we addressed earlier in this paper such as administration, validation, support, training time away from day-to-day work, all factor into the total cost of ownership. Just because you acquire a new software package does not mean you can just turn off the old one right away. It often takes years to fully transition and there will be increased costs while you are supporting both software packages.

CONTACT INFORMATION

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

Brian Varney
Experis
269-365-1755:
Brian.varney@experis.com:
www.experis.com