

Paper 1192-2021

Bridging the Gap between Legacy SAS® and SAS Viya in the Cloud at the Centers for Medicare and Medicaid Services (CMS)

Rick Andrews, CMS; Manuel Figallo, SAS Institute Inc.;
Kevin Boone, SAS Institute Inc.; Prakash Subramanian, SAS Institute Inc.;
Logan Perry, SAS Institute Inc.;

ABSTRACT

SAS Viya has introduced many powerful capabilities to enrich the end user experience. From scalable infrastructure resources to easy-to-use front-end features, SAS Viya can improve a team or organization’s ability to turn data into insights. Having decided to evaluate or move to SAS Viya, end-users are often left wondering how they can “bridge the gap” between their legacy investment in SAS to the more Cloud-native SAS Viya version.

Based on the experiences of such users at CMS, this paper helps others facilitate a move to SAS Viya. Of particular interest are those end-users who:

- Currently use client-server or mainframe applications --such as SAS Enterprise Guide, SAS Stored Processes, and SAS for the IBM® mainframe-- for querying or coding and want to bridge the gap with their SAS Viya equivalents, i.e., SAS Studio and SAS Job Execution.
- Use Open Source (Python) and want to bridge the gap between their Python notebook environment and SAS Viya CAS using Python SWAT (Scripting Wrapper for Analytics Transfer).

INTRODUCTION

Like many government agencies, CMS is utilizing more Cloud-based services. SAS Viya is built to be scalable for both private and public Clouds. Complex analytical, in-memory calculations are optimized, because SAS Viya scales compute capacity to faster process increased workloads using available infrastructure resources. This lets an analyst to quickly experiment with different scenarios and apply more sophisticated approaches, such as machine learning, to increasingly large volumes of incoming data.

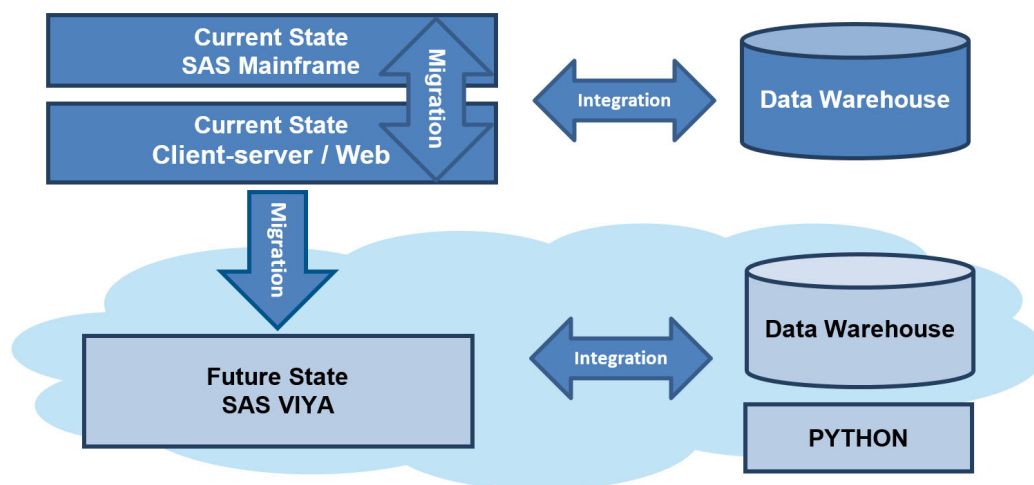


Figure 1: "Bridging the Gap" at CMS Involves Migration and Integration.

As indicated by Figure 1, CMS has used SAS throughout its evolution from mainframe technologies to client-server and web platforms. The next stage in this evolution is the Cloud, an environment for which SAS Viya was built, to process massive amounts of CMS data.

The Cloud's benefits include the following:

- **Scalability.** As already mentioned, this is the ability to scale hardware and infrastructure resources to accommodate ever changing workload requirements. For the purposes of this paper, this results in outcomes such as improved performance and improved cost containment. This capability in the Cloud also reduces the need for extensive capacity planning.
- **Extensibility.** Once deployed in the Cloud, SAS Viya has access to several Cloud managed services that can extend the capabilities already available in SAS Viya, including data services (e.g., Snowflake), ETL/ELT services, and open source environments for Python and even R.
- **Integration.** The Cloud makes it easier to integrate to these Cloud managed services since it uses a standard set of standard protocols for integration utilizing HTTP and APIs.
- **Maintenance.** Since the services are managed, infrastructure, for example, no longer has to be maintained. Also, SAS Viya for the Cloud has been redesigned with microservices for a lighter foot-print and faster deployment. The benefit to the users is that they will experience less system downtime and automatically receive updates for SAS Viya
- **Reliability.** The Cloud also provides access to several service areas for replication and redundancy making SAS Viya highly available.
- **Costs.** Although many of the items in this bulleted list have a cost dimension, it's worth noting that Cloud-enabled platforms like SAS Viya can help CMS move their IT expenses, such as hardware, from a capital expense to an operational one—the equivalent of "renting" IT infrastructure and paying for what is needed.

These benefits provide some of the reasons why CMS is moving to the Cloud.

So what do we mean by "bridging the gap"? As in previous SAS upgrade or "modernization" efforts at CMS, we mean *migrating* code or a GUI to a newer version with as little modification as possible. But, what if we need to *integrate* with new data stores or Cloud services, such as open source programmatic environments in the Cloud-- i.e., Python?

The purpose of this paper is to tackle these questions with a set of best practices for *migrating* to the new, Cloud-ready platform, SAS Viya, or *integrating* with Cloud services for data or open source analytics. These best practices are based on interactions with CMS as they begin to consider moving to SAS Viya in the Cloud. They are meant to provide general guidance, insights and knowledge during the move to the Cloud for any SAS Viya customer, and they highlight the advantages of Infrastructure-as-a-Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS), and even Functions as a Service (FaaS) in the Cloud.

This paper will also help answer questions that normally come up when bridging the gap-- "will my SAS code still run?" and "how will SAS work with new services in the Cloud?" -- and, it is meant for SAS leaders, programmer analysts, and researchers who want and need to understand what advantages SAS Viya in the Cloud can bring to their organizations. But, first, a little bit of SAS history at CMS to give further background and context.

CURRENT STATE

CMS has been SAS customer for many years and has experienced various evolutions of the software throughout its history. By coincidence, CMS and SAS share some similarities. For instance, Medicare was signed into law in 1965 whereas the beginnings of what is now SAS began in 1966 with funding provided by the United States Department of Agriculture (USDA) and the National Institutes of Health (NIH). That said, SAS has been part of Health and Human Services (HHS) since its inception. When CMS split from the Social Security Administration (SSA) in 1976, the SAS system was already being used to analyze Medicare data. That year also happens to be when the SAS Institute was incorporated.

Mainframe

When CMS split from the SSA and instituted its own data center, SAS was one of the first applications installed for data analysis. At that time, most data were being stored within variable length flat files and COBOL was the primary language used by the information technology (IT) office to examine data. However, SAS quickly became the tool of choice.

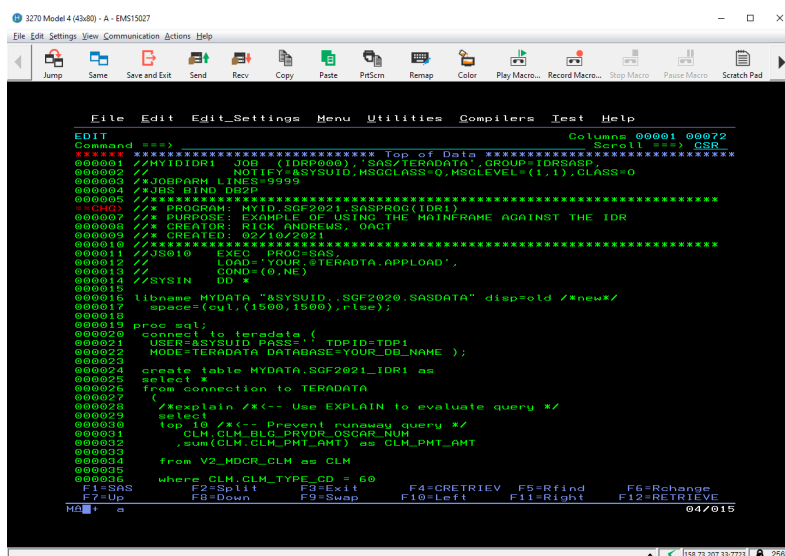
SAS/Access

As CMS data sources have evolved, so has the SAS software. A product called SAS Access has been used against various databases to summarize information from Relational Database Management Systems (RDBMS) like Oracle®, DB2®, Sybase®, and Teradata®. These systems, known as Massively Parallel Processing (MPP) systems vastly improve the speed at which data can be analyzed. The example shown in Display 1: Mainframe SAS Example depicts the use of SAS Access to Teradata to summarize information from the Integrated Data Repository (IDR).

The IDR was implemented under Section 101 of the Medicare Prescription Drug, Improvement, and Modernization Act of 2003 (MMA) (Pub. L. 108-173), to house Part D drug information. The system now includes Parts A, B, C and D, and DME claims, beneficiary and provider data sources, along with ancillary data such as contract information, risk scores, and many others. Access to this robust integrated data supports much needed analytics across CMS.

It should be noted that other database systems are also utilized at CMS for various purposes. For example, the Chronic Conditions Warehouse (CCW) was implemented under Section 723 of the MMA to provide less complicated access to analysts at CMS and other government agencies, as well as external researchers, such as those based in universities. For reference, the underlying database used for this environment is Oracle. Thus, SAS Access to Oracle is used to access those data.

In the future, CMS intends to utilize databases housed within the Cloud. This allows CMS to rent space from entities like Amazon® Web Services (AWS), Microsoft® Azure (Azure) and Google® Cloud Platform (GCP) in lieu of purchasing hardware to be housed within the Baltimore Data Center (BDC.) The evolution to Cloud based systems is cheaper and allows for easier expansion. Cloud-based systems such as Hadoop®, Postgres®, and Redshift® are



```
3270 Model 4 (4368) - A - EM51507
File Edit Settings View Communication Actions Help
Jump Same Save and Exit Send Recv Copy Paste PrtScn Remap Color Play Macro... Record Macro... Stop Macro... Pause Macro Scratch Pad

File Edit Edit_Settings Menu Utilities Compilers Test Help
EDIT Command ===> Column# 00001 00022
***** Top of Data *****
000001 //NY10IDR1 JOB (IDR000), SAS(TERADATA), GROUP=DRSSSP
000002 //          NOTIFY=ASYSUID,MSGCLASS=Q,MSGLEVEL=(1,1),CLASS=O
000003 //JOBPARM LINES=9999
000004 //JBS BIND DB2P
000005 //*****
000006 //          PROGRAM: NY10.SGF2021.SASPROC(1DR1)
000007 //          PURPOSE: EXAMPLE OF USING THE MAINFRAME AGAINST THE IDR
000008 //          CREATOR: RICK ANDRES, GACT
000009 //          CREATED: 02/10/2021
000010 //*****
000011 //JS010 EXEC PROC=SAS,
000012 //          LMOD=YOUR.TERADATA.APPL0AD*,
000013 //          COND=(0,NE)
000014 //SYSIN DD *
000015
000016 Libname MYDATA "ASYSUID.SGF2020.SASDATA" disp=old //newk/
000017 spool=fcut.(1500,1500),rlse);
000018
000019
000020 proc sql;
000021 connect to teradata (
000022 user=ASYSUID PDS= TOPID=TOP1
000023 MODE=TERADATA DATABASE=YOUR_DB_NAME );
000024
000025 create table MYDATA.SGF2021_IDR1 as
000026 select *
000027 from connection to TERADATA
000028
000029 /*explain /*c-- Use EXPLAIN to evaluate query */
000030 top 10 /*c-- Prevent runaway query */
000031 clm CLM_BLD_PROVIDER_OSCORE_NUM
000032 ,sum(CLH.CLH_PMT_AMT) as CLM_PMT_AMT
000033
000034 from V2_HDCR_CLM as CLH
000035
000036 where CLM.CLM_TYPE_CD = 60
000037
000038 F1=SAS F2=Split F3=Exit F4=RETRIEV F5=Rfind F6=Rchange
000039 F7=Up F8=Down F9=Swap F10=Left F11=Right F12=RETRIEVE
04/015
```

Display 1: Mainframe SAS Example

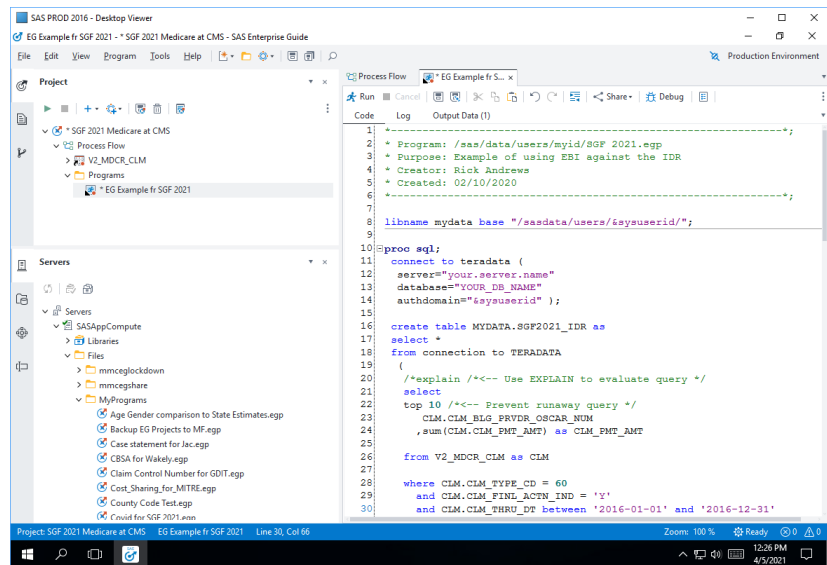
already being used at CMS with the possible expansion to others including Snowflake®. That said, SAS Access engines for these databases are already available to summarize data. Since CMS employees and contractors have been using SAS Access products for many years, conversion to these new systems will be simpler to accomplish.

Client-server (SAS Enterprise Business Intelligence)

Another feature of the current state at CMS is the use of a client-server environment known as the SAS Enterprise Business Intelligence (EBI) server. This setting allows users to migrate existing mainframe queries against the IDR to a product called Enterprise Guide (EG.) The program contained within Display 2: Client-server Example of the code shown in Display 1.

SAS Enterprise Guide

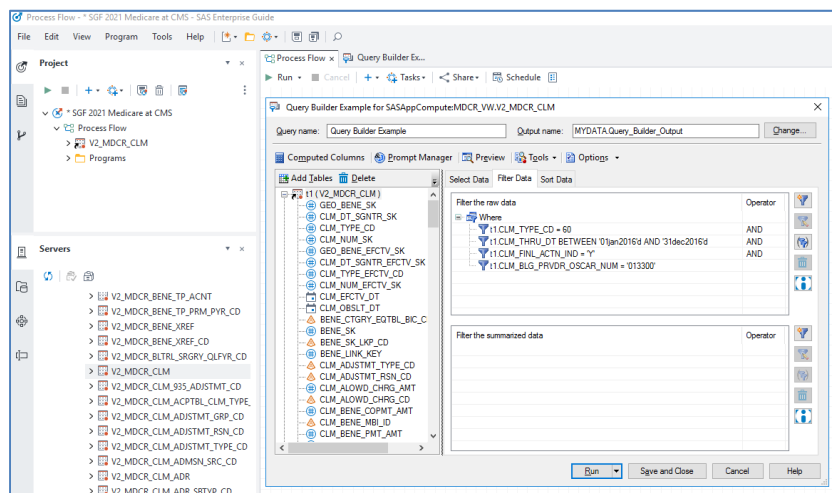
There are two main differences when converting a mainframe process to EG: 1) the LIBNAME statement and 2) the CONNECT statement. The LIBNAME statement points to the location of the SAS output on the EBI server and the CONNECT statement points, in this case, to the IDR - all of the other programming code is the same. It should be noted that using the SQL procedure in this manner is known as an “explicit” pass-through query. This means that any code after the “CONNECTION TO” statement is specific to the database being used. The IDR happens to use Teradata now, but that could change in the future. For example, if CMS migrates to a Cloud-based system such as Snowflake the CONNECT statement will need to change to access the new database.



Display 2: Client-server Example

Query Builder

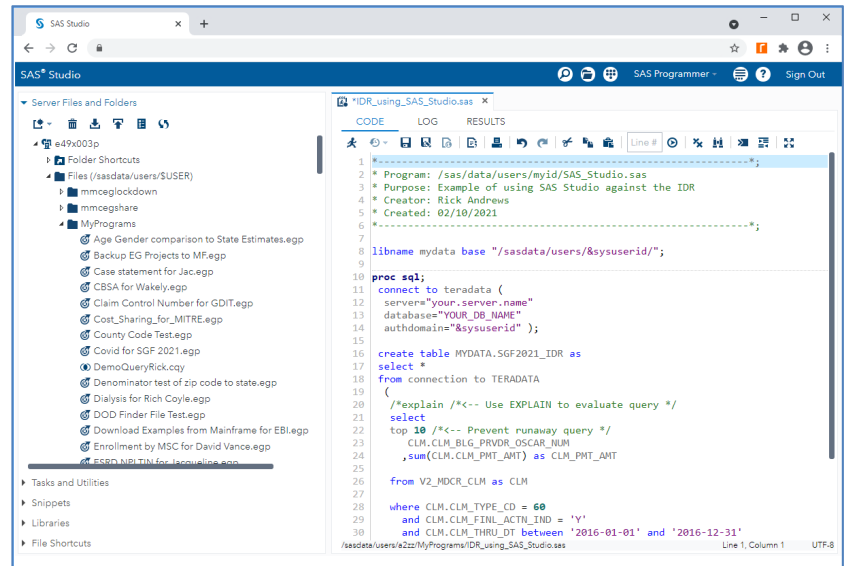
Another solution within the current environment at CMS allows users to create queries without having to know database specific syntax. This is known as an “implicit” query within SAS. Instead of coding a CONNECT statement; a LIBNAME statement is coded that points to the database. Display 3: Query Builder Example illustrates the use of the Query Builder within EG to perform the same summary as in the previous examples.



Display 3: Query Builder Example

Web Application (SAS Studio)

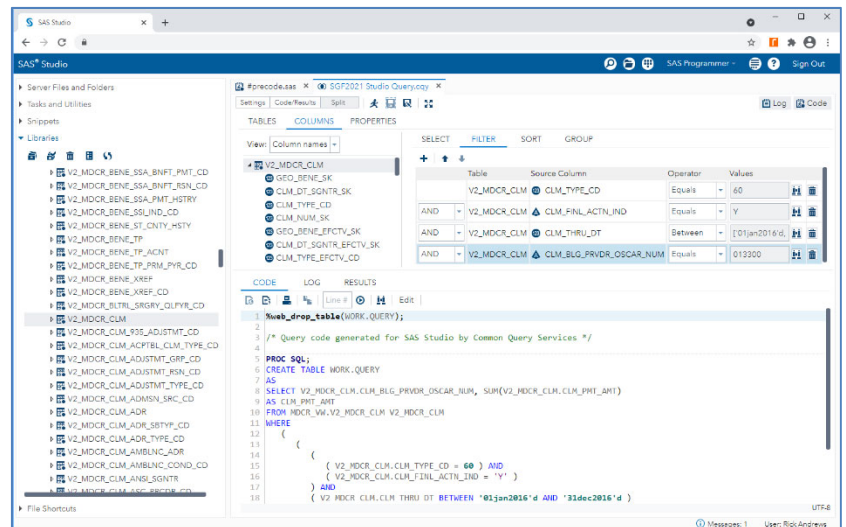
There is yet another mechanism within the current state at CMS to summarize data from the IDR known as SAS Studio. This is a web-based application that does not require a user to run dedicated client software, as the client is a web application. All that is required is an appropriate web browser such as Microsoft Edge, Mozilla Firefox, or Google Chrome. Display 4: Web Programming Example depicts the exact same code as shown above and yields the same results.



Display 4: Web Programming Example

Visual Query Tool

There is a visual query tool within SAS Studio that is very similar to the Query Builder within EG. One of the advantages of learning SAS Studio is that it is the programming arm of SAS Viya. Almost all of the SAS programming code that users write today can be copied into SAS Studio. One current limitation is that SAS Viya does not rely on SAS Metadata services and therefore process flows developed in EG are not easily copied into the new environment. SAS is currently working on a mechanism to allow for this functionality.



Display 5: Visual Query Tool Example

FUTURE STATE

The future state consists of a web browser, which allows end users to access SAS Viya applications in the Cloud from a desktop computer. SAS Viya is SAS' next generation high-performance in-memory analytics platform designed to run in the Cloud.

SAS Viya includes applications, such as SAS Studio for programming and flows, SAS Visual Analytics for dashboards and analytics, and Python Notebooks to run data or machine learning pipelines that integrate with SAS Viya.

So, if you're comfortable with a web browser, then you'll be able comfortable with a lot of the capabilities in SAS Viya.

Understanding SAS Viya

There are several important things to understand about SAS Viya:

- Many SAS Viya clients (e.g., SAS Visual Analytics) do not submit SAS code but call Cloud Analytic Services (CAS) directly. CAS is one of two in-memory processing engines available in SAS Viya, and it is discussed in detail later on.
- SAS Viya clients all use a common HTML5 web interface. This common interface allows for all functionality to be merged into a single user experience.
- There is no requirement to learn SAS code to use SAS Viya. The drag-and-drop interface opens analytics to all types of personas (programmer analysts, researchers, executives, etc.).
- SAS Viya contains the latest capabilities such as machine learning so that your organization can get the most out of your analytics.

A web browser also enables you to integrate Python with SAS Viya using the SWAT package. SWAT (SAS Scripting Wrapper for Analytics Transfer) enables open-source software such as Python and R to integrate with the dashboard and model governance capabilities in SAS Viya. SAS Viya is also more than just a User Interface (UI) and has an architecture especially designed for the Cloud.

Building Blocks

SAS Viya requires a web browser to interact with its Cloud-based architecture. The architecture is comprised of:

1. Microservices,
2. CAS (Cloud Analytic Services), and
3. SPRE (SAS Programming Runtime Environment).

For now, you can think of CAS as the new in-memory engine and SPRE as an instance of SAS' legacy workspace server (without a Metadata Server). This section describes all three components in detail and provides a graphic, see Figure 2, that provides a good summary.

Pay particular note that SAS expects that it will take time for customers to realize the full replacement value in SAS Viya, and adoption may take a while, so SAS has architected SAS Viya to include two compute engines – SAS SPRE, the "legacy" engine, and SAS CAS. These two compute engines are technologically bridged.

The three main components of SAS Viya are easy to remember:

- Microservices, CAS, and SPRE.

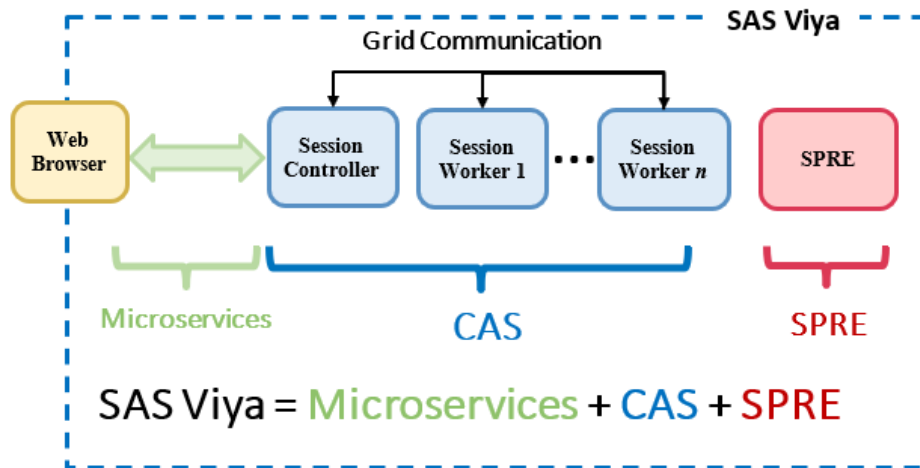


Figure 2: Major Components of SAS Viya

In SAS Viya and as shown in Figure 2, a web browser interacts with SAS VIYA using microservices. They are a modular set of discrete services, such as Audit, Credentials, and Identities. Each microservice runs in its own process and communicates using HTTP.

SAS Viya also includes Cloud Analytic Services or CAS which provides the run-time environment in which data management and analytics take place. CAS is a server that is suitable for both on-premises and Cloud deployments, and CAS can be deployed to a single node or across multiple nodes.

As depicted by the three blue squares in Figure 2, the in-memory analytic features of CAS can be achieved by distributing the CAS server across multiple nodes to enable massively parallel processing (MPP). One advantage or benefit of MPP is that, because data is in-memory and can be processed in parallel, results can be produced much more quickly.

In sum, the CAS server offers dynamic scalability, a virtual memory footprint, high availability, parallel data loading, shared library access, and integration with open-source languages (e.g., Python) and REST APIs.

To use data in CAS programmatically, you will need to load it from the filesystem to memory as shown in Display 6.

```

/* change the host and port to match your site */
options cashost="cloud.example.com" casport=5570;

/* start a session, if you do not already have one */
*cas casauto;

proc cas;
  session casauto;

  /* Load source data (CMS) into a table. */
  table.loadTable / /*#1*/
    path="CMS.sashdat"
    casout={name="CMS"};
run;

  table.tableInfo / table="CMS"; /*#2*/
  table.tableDetails / table="CMS"; /*#3*/
quit;

```

Display 6: Using SAS Code to Load Data into CAS

Note the following:

1. Use table.loadTable action to load a table from caslins' data source.
2. Use table.tableInfo action to view information about the table.
3. Use table.tableDetails action to get details information.

The last box denoted by red in Display Figure 2 is the SPRE engine or SAS Programming Runtime Environment (SPRE). It is also referred to as the compute server in SAS Viya, and it is the equivalent of most everything CMS has today. SPRE, in other words, is the runtime engine for the SAS language.

You can think of this engine as a SAS Workspace Server for legacy code. The primary difference between SPRE and a workspace server (from “legacy” SAS), as already mentioned, is that SPRE does not support the SAS® Metadata Server technology. SPRE does not provide this support because SAS Metadata is part a monolithic design that precedes the microservices Cloud design of SAS Viya.

In summary, it’s worth reiterating that SAS Viya has two compute engines as denoted by the blue and red boxes. Remember this simple formula which is:

$$\text{SAS Viya} = \text{Microservices} + \text{CAS} + \text{SPRE}$$

Microservices deserve particular attention. Microservices are an architectural approach to creating Cloud applications. Each application is built as a set of services, and each service runs in its own processes and communicates through APIs. A microservices architecture is a way of developing applications that has matured into a best practice over time and it provides many of the benefits available in SAS Viya such as improved performance (because of a lighter footprint), scalability, redundancy, and CI/CD or continuous delivery of software —i.e., a change to a small part of the application only requires rebuilding and redeploying only one or a small number of services. In the monolithic approach, an application supported by three microservices, for example, would have to be scaled in its entirety even if only one of these microservices had a resource constraint. Revisiting performance for a moment, it is common for microservices architectures to be adopted for Cloud-native applications because they are deployed in containers and are much better at scaling up to handle increased workloads.

SAS Viya Ecosystem (Conceptual Architecture)

The Centers for Medicare and Medicaid Services (CMS) intends to use SAS Viya to generate insights from large amounts of data since it leverages SAS high-performance analytic technologies which includes several runtime environments and applications services. Several that were discussed are in this paper including CAS, SPRE, and open source technologies such as Python.

SAS Viya also empowers organizations to explore huge volumes of data quickly with several web browser applications to identify patterns, trends, and opportunities for further analysis. The highly visual, drag-and-drop data interface of SAS Visual Analytics, for example, combined with the speed of SAS Cloud Analytic Services (CAS), accelerates analytic computations and enables CMS to derive value from massive amounts of data quickly. Figure 3 captures the web applications in the box on the far right. Data sources are in the far left, and the underlying cloud infrastructure is depicted by the two bottom layers. Several data connectors or access engines allow SAS Viya to bring in data to its environment from Cloud sources such as Snowflake, Redshift and Databricks, as well as on-prem data sources.

The top layer in Figure 3 includes some of the analytic capabilities that SAS Viya offers, and at the center of the figure are the building blocks mentioned thus far. All of this is done in the Cloud with a simple, scalable and secure platform architecture that includes technologies such as containers and Kubernetes. Kubernetes or K8s are a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both scalability and high availability (HA).

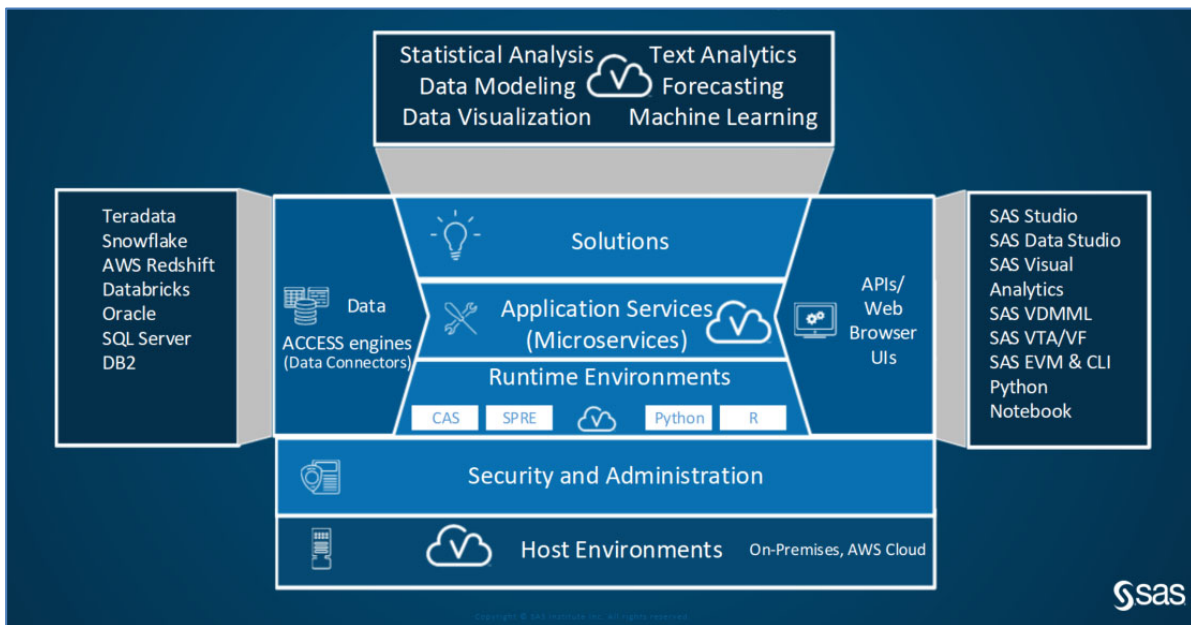


Figure 3: A Conceptual Architecture for SAS Viya

Considerations

You can almost think of the components in the conceptual architecture as lego blocks. The benefit to CMS is that it would be able to implement solutions that incorporate different components or lego blocks depending on business needs for statistical analysis, data modeling, data visualization, text analytics, forecasting and machine learning.

What’s more, CMS will be more easily able to rapidly traverse the analytics lifecycle in an iterative and incremental fashion, all in the spirit of Agile’s continuous improvement. As shown in Figure 4, this means moving from data management, to discovery (data exploration and modeling building), and deployment as effectively and efficiently as possible. Figure 4 provides the foundation for analytics prototyping, pipeline development, and more mature ModelOps processes.

When iterating through the analytics lifecycle, it is important for organizations to quickly adjust and change based on the risks encountered, which can be linked to requirements, technology, skills, and even getting buy-in from key stakeholders.

These powerful capabilities in SAS Viya allow CMS to solve such pressing business challenges such as fighting Medicare fraud, understanding patient and provider outcomes, and making enormous amounts of text data, such as Medicare regulations, easily digestible.

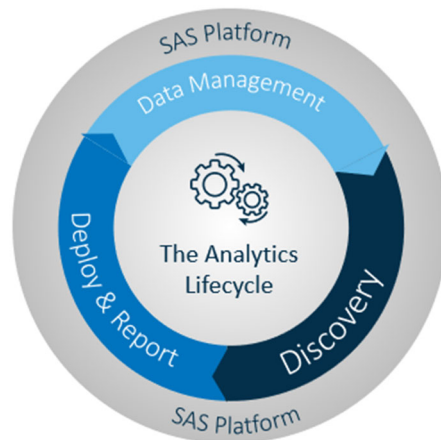


Figure 4: The SAS Analytics Lifecycle

Cloud Data

Every analytics journey begins with data. Cloud data is a model of computer storage in which the digital data is stored in logical pools in "the Cloud". With Cloud data, storage and compute are often separated. The physical storage spans multiple servers (sometimes in multiple locations), and the physical environment for compute is typically owned, managed, and "rented" from a hosting company, such as AWS. Both storage and compute are highly scalable and reliable in the Cloud, which are important attributes for end-users who demand fast performance.

One additional advantage of Cloud data is that government agencies, such as CMS, need only pay for the storage they actually use, typically an average of consumption during a month. This does not mean that Cloud storage is less expensive, only that it incurs operating expenses rather than capital expenses. And if managed appropriately, it can result in cost savings. Please refer to Snowflake documentation for additional details on Cloud data or read Jeff Bailey's excellent paper, "An Insider's Guide to SAS/ACCESS® Interface to Snowflake."

Snowflake Integration

SAS/ACCESS Interface to Snowflake enables you to connect SAS with your data in a Snowflake data source. The Snowflake interface includes SAS Data Connector to Snowflake. The data connector enables you to load large amounts of data into the CAS server for parallel processing.

Here is an example to establish a connection between your Snowflake database and SAS Cloud Analytic Services:

```
&macro extractSnowflakeData(SnowflakeDB_Source=, SASDataset_Target=);

libname SF_LIB odbc
noprompt="driver={SnowflakeDSIIDriver};server=test.snowflakedatawarehouse.com;
warehouse=SAS_WH; database=SAS_TEST; Uid=sasdemo_user; Pwd=XXXXXX;" schema=PUBLIC;

data &SASDataset_Target;
  set SF_LIB.&SnowflakeDB_Source;
run;

proc contents data=&SASDataset_Target; run;

&mend extractSnowflakeData;

&extractSnowflakeData(SnowflakeDB_Source=CMS_SF_DATA, SASDataset_Target=work.CMS_DATA)
```

Display 7: A SAS Macro to Extract Snowflake Data Into SAS.

All users with SAS/ACCESS Interface to Snowflake can use SAS Data Connector to Snowflake.

Python

One best practice is to augment Python capabilities with Viya applications in the Cloud for discovery and deployment in a scalable and secure fashion as shown in Figure 5.



Figure 5: “Bridging the Gap” also involves integrating Python with SAS Viya

Here we are showing a Python pipeline integrated with SAS Viya to produce role-based and secure dashboards and reports for widespread distribution within CMS. We have used this approach at SAS to better understand hospital and nursing home care during the COVID-19 pandemic among Medicare beneficiaries.

A sample Python is available in our Github location:

<https://github.com/ManuelFigallo/sasgovernment-/tree/main/tutorials/SGF/2021>

To use Python with SAS Cloud Analytic Services, the client machine that runs Python must meet the following requirements:

- Use 64-bit Linux or 64-bit Windows.
- Use a 64-bit version of Python, such as the Anaconda platform from Continuum Analytics.
- Use Python version 2.7.x or 3.4+. The single Python package from SAS, SWAT, is compatible with both versions.

The SAS Scripting Wrapper for Analytics Transfer (SWAT) package is available for download from <http://support.sas.com/downloads/package.htm?pid=1977>. Information for installing is available from a README that is included in the download. Let’s reiew SWAT in more detail.

Integrating Python with SAS Viya (SWAT)

Python users gain access to the SAS Cloud Analytic Services (CAS) engine through SAS SWAT. The SAS Scripting Wrapper for Analytics Transfer (SWAT) package is a Python interface that lets the user leverage the power of CAS while working in a Python environment. The user can access in-memory tables and utilize the CAS engine for accelerated large data processing and modeling, all while using a familiar syntax. Data manipulation and processing is made easy in SWAT by mimicking much of the Pandas API.

Handling data with SWAT will feel familiar to Pandas users with methods like head, tail, summary, and functionality of both loc and iloc available.

SWAT allows for uploading data into CAS, meaning the user can take data local to their machine, upload it to into memory, and then combine with other in-memory tables. Alternatively, summary data can be brought down to the local level to be worked with on the client side and combined with local tables for further processing. Bringing down summary data or subsets of data out of CAS and into the Python environment is a best practice when working with large data, because moving the data out of CAS means it will be written to disk and could cause performance issues.

Python is versatile and has a vast user base, especially in the data science community. In particular, it is an excellent tool to use in traversing the entire analytics lifecycle beginning with data management, then data exploration, model building and finally model deployment—as discussed above.

Many Python packages are specifically made for parts of the machine learning pipeline. Pandas and NumPy are great for cleaning data and data transformations.

```
# Convert column 'Week_Ending' to date format using pandas 'to_datetime' function
nh4.loc[:, "Week_Ending"] = pd.to_datetime(nh3["Week_Ending"])
```

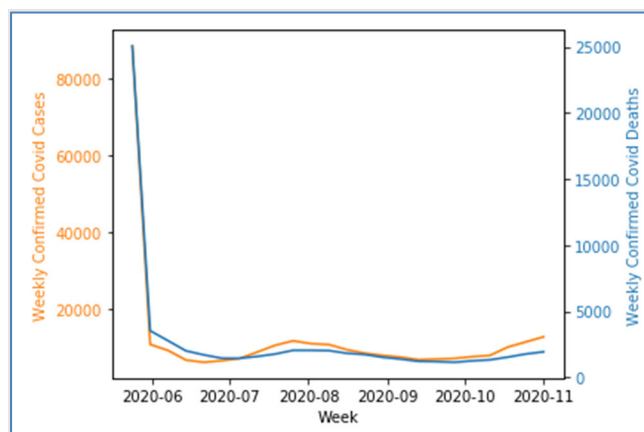
Display 8: Data Management Tasks (Such As Type Conversion in Python) Are easy and efficient.

Seaborn and Matplotlib offer visualizations to aid in understanding the data and validating model assumptions.

```
# plot total weekly cases and weekly deaths together in same figure
fig, ax1 = plt.subplots()
color = 'tab:orange'
ax1.plot(weekly_cases_deaths.index, weekly_cases_deaths['Residents_Weekly_Confirmed_COVID'], color=color)
ax1.tick_params(axis='y', labelcolor=color)
ax1.set_xlabel('Week')
ax1.set_ylabel('Weekly Confirmed Covid Cases', color=color)

ax2 = ax1.twinx()
color = 'tab:blue'
ax2.plot(weekly_cases_deaths.index, weekly_cases_deaths['Residents_Weekly_COVID_19_Deaths'], color=color)
ax2.set_ylabel('Weekly Confirmed Covid Deaths', color=color)
ax2.tick_params(axis='y', labelcolor=color)

fig.tight_layout()
```



Display 9: Python can be for highly customizable multilayer visuals for data discovery and exploration.

Keras and Scikit-learn offer a multitude of models and assessment tools. These packages are only a drop in the bucket of all that python has to offer to any data pipeline.

```
from xgboost import XGBRegressor

boost = XGBRegressor(random_state=76, learning_rate=0.2)

boost.fit(X_train, y_train)
```

Display 10: Building and Fitting Models is easy in Python.

For the purposes of this paper and as demonstrated above, Python is used for data management and data discovery tasks. One best practice is to augment these Python capabilities with SAS Viya capabilities for deployment, such as model management and governed reports. Not only will this close the loop in an analytics lifecycle iteration but it will better position any organization, including CMS, to adopt more mature analytical capabilities to solve pressing business and policy challenges.

SAS SWAT is the bridge between Python and Viya. Where and when to place the bridge is up to the user and circumstance. Data preprocessing such as cleaning the data, feature engineering, data transformations, and data validation might be easier to code in Python. Then, the resulting table could be brought into CAS through SAS SWAT to be used in model studio for model tracking and comparison. Results can then be easily passed to SAS Visual Analytics for a dashboard report.

```
# read in pandas DataFrames to CAS
sess.read_frame(train, casout={'caslib':'NH', 'name':'nh_train', 'promote':True})
sess.read_frame(test, casout={'caslib':'NH', 'name':'nh_test', 'promote':True})
```

Display 11: Uploading Pandas data frames into memory and promoting the tables to CAS using SWAT

The opposite approach can also be taken. SAS Viya Data Explorer and Data Studio offers a point and click interface for loading data into CAS and cleaning it up for analysis. The clean table can then be loaded into a Python environment through SWAT and brought in as a Pandas DataFrame. Any Python modeling package can be used on the data, and then imported into SAS. From there the model can be put into production or into Model Studio for further comparison with other built-in models. After a champion model is crowned, it's a few clicks to transfer that information to a SAS Visual Analytics Dashboard for reporting.

```
# set model and project name
model_name = 'XGBoost Regressor NH3'
project_name = 'Nursing Home Covid-19'

# Register the model in SAS Model Manager
model = register_model(boost, model_name, project_name, input=features_train, force=True)
```

Display 12: Registering the XGBoost Python model to Model Manager in SAS Viya

SAS Viya and Python are two powerful tools. However, there are some best practices to keep in mind. One is to avoid sending the data back and forth between disk and memory, especially if the dataset is large. In-memory CAS tables can be accessed in a Python environment and be processed in CAS through the Pandas API and CAS Action Sets. This is preferred due to the processing prowess of SAS CAS using in-memory tables.

The advantage of in-memory tables is taken away when the in-memory table is written to disk and saved as a Pandas DataFrame. Bringing the table down into disk permits more

flexibility in Python since it can be used with packages, but at a processing and storage cost. As such, sending data back and forth between CAS and the Python environment should be minimal.

Options for a Python Environment

SAS Viya is the Cloud-native analytics platform on which the latest software offerings from SAS are built and run. SAS Viya offerings are delivered as a set of container images that are deployed into a Kubernetes (K8) cluster as mentioned earlier.

In this section, we explore key considerations when deploying SAS Viya in K8.

The recommended topology that includes a Jupyterhub instance deployed outside the SAS Viya Kubernetes cluster is shown in Figure 6.

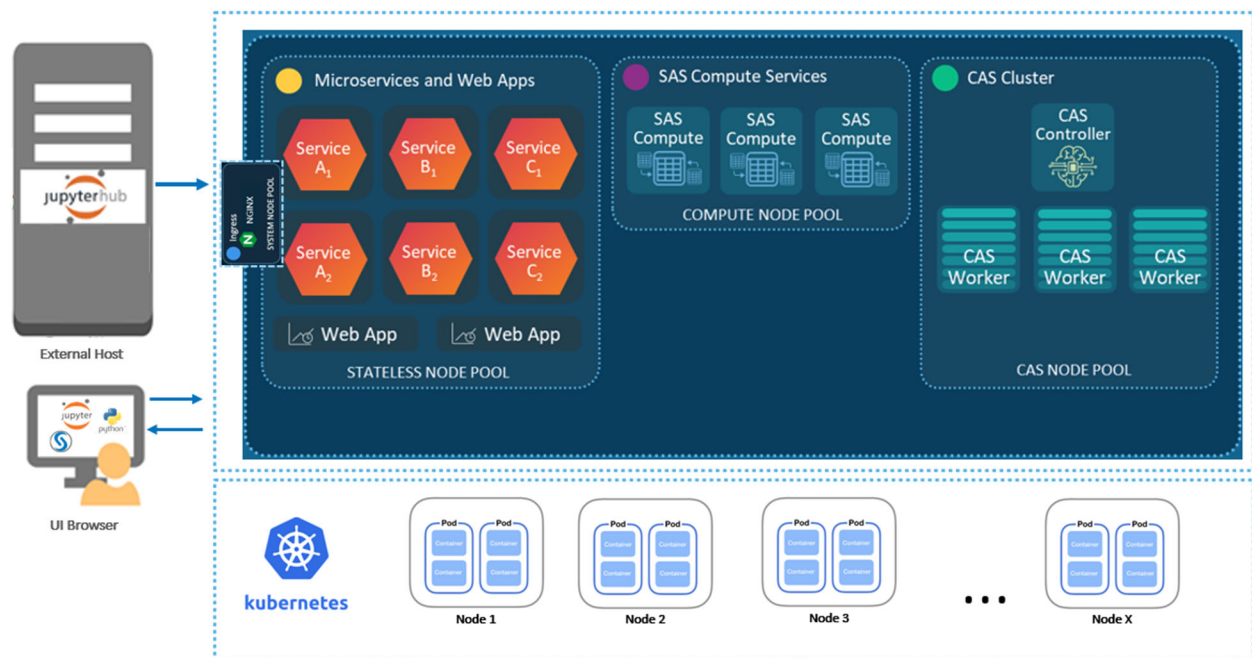


Figure 6: The SAS Architecture is a two layer "data cake" consisting of a highly-scalable Kubernetes layer at the bottom, and the SAS Viya Platform that integrates with Python on top.

As shown in Figure 6, SAS Viya implements strategies using Docker containers and Kubernetes orchestration to provide more composable, scalable, and maintainable deployments that can include additional Cloud services such as Python.

Once deployed there are several methods to connect a Python environment with the CAS Server:

1. Binary Connection method. If a server is listening on the host name and port that are specified, and you authenticate, then the SWAT CAS class makes a connection to the server, starts a session on the same hosts as the server, and returns the connection object. For example:

```
import swat
s = swat.CAS("cloud.example.com", 5570)
```

2. [REST Connection Directly to CAS](#). The best practice in this case is to connect to CAS through the HTTP server. The `cas-shared-default-http` portion of the URL applies to a typical SAS Viya deployment. If the deployment instance did not use the default deployment name, then that portion of the URL is different.

```
s = swat.CAS("http://cloud.example.com:8777", protocol='http')
```

In some network topologies, you might have network connectivity with the CAS controller. The CAS controller listens on port 8777 for REST connections. The port number is configurable and can be different. Specify HTTPS in the URI and for the protocol argument if the CAS controller is configured to use TLS.

Scalability is the property of a system to handle a growing amount of work by adding resources to the system, and it's one of the main attractions of SAS Viya in the Cloud.

To decide whether SAS Viya should be scaled it is helpful to determine if it takes "too long" to run. This may mean that the time required to run a job exceeds the batch window of time that you have available. Or it may mean that it takes "too long" for you to get the information from your application in order to make timely decisions. Next it is important to identify the pieces of the application that seem to consume the most time. Then you can determine if these portions of your task are compute intensive or if they are I/O bound. This will help you to understand how scalable a particular task may be.

The primary drivers / reasons for customers wanting to scale their SAS Viya environment are:

1. To improve performance
2. Improve availability and implement High Availability or HA
3. Cost optimization

Scalability can be addressed from two directions: scale up and scale out. It is important to realize that these are not mutually exclusive choices. Scaling up, from a hardware perspective, means increasing the number of processors, disk drives, I/O channels, etc. on a single server machine or VM (virtual machine). Scaling out, on the other hand, means adding more hardware, not bigger hardware. In the case of SAS Viya, that may mean, for example, adding additional K8 nodes to increase the number of worker nodes in a CAS cluster.

It is not true that a computation that runs in parallel will always run faster than a serial (single-threaded) version of the same computation. When you distribute a computation, the performance benefit depends on several factors, including the size of the problem, the number of threads, the work done in each thread, and the cost of transmitting data between the controller node and worker nodes.

The following statements are often generally true:

- Small computations run faster in a single thread. For small problems, the overhead costs of communicating with multiple threads are often greater than the cost of the computation.
- Using more nodes is more expensive than using more threads.

- Sending large amounts of data between nodes is expensive.

Finally, larger numbers of elements increases management complexity, more sophisticated programming to allocate tasks among resources and handle issues such as throughput and latency across nodes.

High availability or HA occurs when servers support server applications that can be reliably utilized with a minimum amount of down-time. They operate by using high availability software to harness redundant computers in groups or clusters that provide continued service when system components fail. Without clustering, if a server running a particular application crashes, the application will be unavailable until the crashed server is fixed. HA clustering remedies this situation by detecting hardware/software faults, and immediately restarting the application on another system without requiring administrative intervention, a process known as failover. As part of this process, clustering software may configure the node before starting the application on it.

Distributed CAS servers are fault tolerant. If communication with a worker node is lost, a surviving worker node uses a redundant copy of the data to complete the data analysis.

In summary, with SAS Viya, HA exists at multiple levels: Infrastructure (Cloud provider); Kubernetes cluster; and the SAS Viya deployment.

ModelOps: Looking Further Ahead

ModelOps is how analytical models are cycled from the data science team to the IT production team in a regular cadence of deployment and updates.

It is a set of techniques and technical capabilities to help operationalize machine learning models and pipelines in the Cloud. Without ModelOps, it is difficult to achieve the kind of impact intended by operationalizing machine learning, for example.

Three goals define the purpose of ModelOps:

1. Faster time to value
 - a. from development to deployment
2. Scale Analytics
 - a. scale model and data
3. Analytics driven, justifiable and better decisions
 - a. monitor business impact of models
 - b. toward real-time models

More and more, organizations such as CMS are relying on machine learning (ML) models to turn massive volumes of data into fresh insights and information. These ML models are not limited by the number of data dimensions they can effectively access and use vast amounts of unstructured data to identify patterns for predictive purposes. Figure 7 offers a simplified view of Models, which includes inputs (A Question), outputs (Decisions) and a description of how the end-user will interact with the system as denoted by the three circles in-between for data management, model development, and deployment.

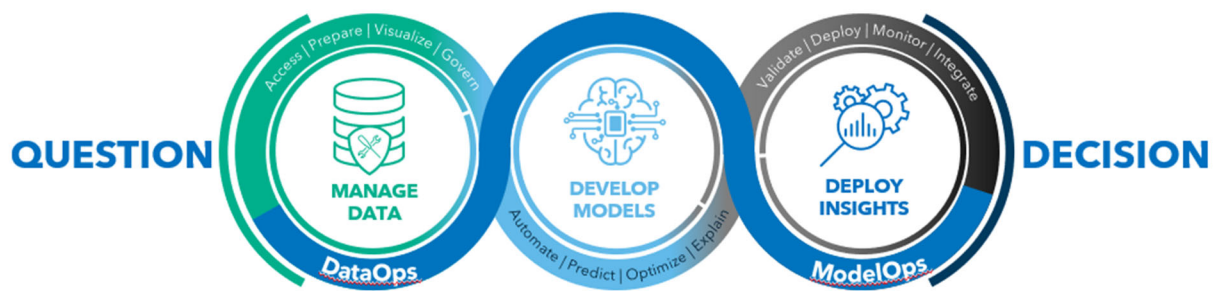


Figure 7: A Simplified View of ModelOps

ModelOps lies at the heart of any enterprise machine learning strategy. It orchestrates the model life cycles of all models in production across the entire enterprise, from putting a model into production, then evaluating and updating the resulting application according to a set of governance rules, including both technical and business KPI's. It grants business domain experts the capability to evaluate models in production, independent of data scientists.

It's worth noting that SAS Viya includes an HTML5-based interface for ModelOps called Model Manager, which runs on its open, modern, microservices architecture. SAS Viya's centralized model repository not only supports the ability to import models from a SAS 9.4 environment, but also offers model registration from a broader set of modeling applications that run on SAS Viya, including SAS Model Studio (available with SAS Visual Data Mining and Machine Learning [VDMML] and SAS Visual Text Analytics), SAS Studio, SAS Visual Analytics, and SAS Visual Statistics.

SAS Viya's version also supports Python and R models for more open modeling support, including publishing to run-time containers. Once registered in the common model repository, you can proactively monitor both SAS and open-source classification and prediction model performance in order to identify deviations in model output or model input data. And like the other microservice-based products in SAS Viya, SAS Model Manager supports open REST APIs for ease of access by clients such as SAS Intelligent Decisioning, which can be used to develop workflows.

USE CASES

Example Migration Use Cases

SAS users at CMS have been preparing for the SAS Viya migration with help from SAS system administrators, trainers, and coaches. The use cases presented here were developed to assist with migration and have yielded some best practices to prepare for migration.

Practice Using SAS in a Web Browser

SAS Studio 3.8 Enterprise Edition is available in the SAS EBI system at CMS. SAS Studio offers many of the same capabilities as Enterprise Guide ("EG"), and it can also use data and programs that were created in EG. Since SAS Viya applications run in a web browser, SAS Studio 3.8 is useful for establishing familiarity and comfort with a browser-based SAS editing environment. Some CMS users have recreated EG process flows using the tasks available in SAS Studio 3, as illustrated in Figure 8, while others have opened and modified existing Enterprise Guide Project (".EGP") files in the Visual Programmer perspective. Other users who prefer writing and working with code have utilized their existing programs and

created new ones in SAS Studio 3.8. Practice with SAS Studio has prepared these users for the user interface changes to come in SAS Viya.

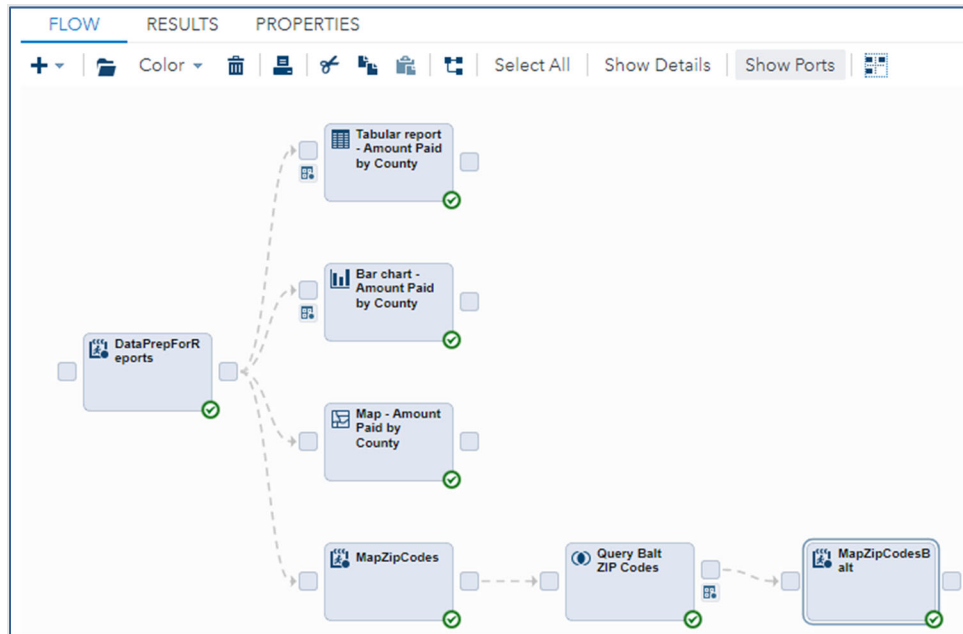


Figure 8 - Existing Process Flow Recreated in SAS Studio

Know Your Content

The existing SAS EBI system has thousands of metadata objects, mostly stored processes and tables registered in metadata. There are also many thousands of SAS programs, datasets, and Enterprise Guide Project (".EGP") files saved on the file server. There are many project teams utilizing the system and each team is responsible for its own content. The SAS administrators have little insight into each project team's work, let alone each item to migrate. Therefore, teams and individual users are being encouraged to assess their content on the file server and within metadata. Teams have started to examine EGP process flows and SAS programs to determine the purpose of each item and whether it should be retained and migrated to SAS Viya. They've also begun inventory of stored processes to determine the same information along with the purpose/results of each stored process. Some stored processes will be better served as reports in Visual Analytics on Viya while others will function best as Viya jobs. Part of the inventory includes disposition of what to do with each stored process so that only required content is migrated to SAS Viya. An informed migration that transfers only needed items will be more efficient than simply migrating everything and will likely result in a more successful migration.

Example Integration Use Cases

As discussed, "bridging the gap" also means integrating new Cloud services and capabilities that did not exist before model management of both SAS and Python models. Although this section is primarily focused on integrating with Model Manager, there are many other services in the Cloud that SAS Viya can integrate with, including databases, machine learning and AI capabilities, as well as just about any modern computing capability imaginable.

Integrating Open Source in a ModelOps Process using Model Manager

As new data comes in, old models phase out. Model Manager allows for both the scoring of current models against new data for more static situations, and model comparison for when input data has been updated and a new model has been created for comparison against the previous iteration. Model comparison can also be accomplished using SAS Model Studio. The Covid-19 Nursing Home data set is an example of needing to create a model to compare to the existing model in production. It is available in the following GitHub location:

<https://github.com/ManuelFigallo/sasgovernment-/tree/main/tutorials/SGF/2021>

Remember that new data means a different type of model might also be best, so putting the data through the Python pipeline for ETL processing and initial model determination is best in this case. Next, SASCTL can be used to import the new python champion model into model manager to be scored against the existing model.

ROADMAP

“Bridging the gap” involves a set of best practices covered in this paper to either 1) **migrate** your current or legacy environment with future state capabilities using SAS Viya; or, 2) **integrate** with future state capabilities in the Cloud such as Data Warehouses – Snowflake or Databricks—and even Python, technologies that enrich and enhance the SAS experience.

Any roadmap that’s part of a move from a legacy environment to a future state one should benefit from the best practices described in this paper.

Other practices and consideration for any migration or integration efforts include:

- **Creating a “Sandbox”.** At CMS, we call the “sandbox” environment for SAS Viya the “Innovation Lab”. A sandbox environment can be used for the following reasons:
 - **Collecting Use Cases.** Many end-users may be curious about SAS Viya capabilities and a sandbox provides them with an opportunity to solve their use case with these new capabilities. It’s important to provide user coaching and support to facilitate this evaluation.
 - **Building Community.** Anyone who has done analytics in the past knows the importance of having a good network. A sandbox is one way to build that community or network to share and support new users as they embark on an analytics journey.
- **Change Management.** As organization iterate through the analytics lifecycle, it will be clear that new skills will be required. Identifying those skills gaps and addressing them with training, coaching, or mentoring is important. It can be done in a grassroots way using a sandbox environment, or through formal training offered by SAS. More broadly, organizational buy-in is also important and so a sandbox can also be used to showcase capabilities.
- **Content Assessment.** SAS provides a tool for content assessment to examine the characteristics of your current state environment. Each application examines your SAS 9 system for relevant information, gathers key details, and produces results from each part of the assessment. More information is available at:

https://go.documentation.sas.com/doc/en/pgmsascdc/9.4_3.5/whatsdiff/n1w49eit6yfog6n1w6t81ue288b5.htm

CONCLUSION

The Cloud is arguably a *revolutionary* technology and this paper has shown that the transition to the Cloud for organizations such as CMS can be *evolutionary*. The addition of SAS Viya to the list of applications available to employees, contractors, researchers, and policy makers promises to bring a new breadth of tools to analyze Medicare and Medicaid data.

SAS Viya also leverages all the powerful analytical techniques SAS is known for, as well as the latest developments in technology of the last few years, such as Python. As importantly, SAS Viya's architecture enables you to run your jobs faster.

In addition to tools like SAS Studio that provide a user with all of the historical capabilities of the SAS software, it also allows individuals to run processes from a web browser, which removes the need for additional interfaces. In addition, SAS Viya enables new technologies such as SAS Visual Analytics, SAS Scripting Wrapper for Analytics Transfer, and SAS Visual Data Mining and Machine Learning that allows users to create analytic results easier than ever.

REFERENCES

Andrews, Richard. 2020. "Accessing Medicare Data at the Centers for Medicare and Medicaid Services using SAS®." Proceedings of the SAS Global Forum 2020 Conference. Available at: <https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2020/4285-2020.pdf>

Bailey, Jeff. 2020. "An Insider's Guide to SAS/ACCESS® Interface to Snowflake." Proceedings of the SAS Global Forum 2020 Conference. Available at: <https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2020/4103-2020.pdf>

Crevar, Margaret. 2020. "Important Performance Considerations When Moving SAS® to a Public Cloud." Proceedings of the SAS Global Forum 2020 Conference. Available at: <https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2020/4312-2020.pdf>

Figallo-Monge, Manuel. 2016. "Pedal-to-the-Metal Analytics with SAS® Studio, SAS® Visual Analytics, SAS® Visual Statistics, and SAS® Contextual Analysis." Proceedings of the SAS Global Forum 2016 Conference. Available at: <http://support.sas.com/resources/papers/proceedings16/SAS6560-2016.pdf>

Figallo-Monge, Manuel. 2018. "Location Matters: Evidence from Spatial Econometric Analysis of Opioid Prescribing Rates." Proceedings of the SAS Global Forum 2018 Conference. Available at: <https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2018/2141-2018.pdf>

Grance, Timothy and Peter Mell. "The NIST Definition of Cloud Computing." Available at: <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>

Kumar, Deva. "restAF – A JavaScript Library for Rapidly Developing SAS® Viya® Applications Based on SAS® REST APIs." Available at: <https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2018/1882-2018.pdf>

SAS Institute, Incorporated. "Introduction to SAS Viya." Available at: https://go.documentation.sas.com/doc/en/sasadmincdc/v_011/viyaov/n00000sasviya000architecture.htm

SAS Institute, Incorporated. "Scaling Up and Scaling Down a SAS Viya Deployment." Available at:

https://go.documentation.sas.com/doc/en/sasadmincdc/v_011/calchkadm/p17xfmmjkm1dn1b5dcx3e5ejxq.htm

SAS Institute, Incorporated. "Getting Started with SAS® Viya® for Python." Available at: <https://go.documentation.sas.com/doc/en/pgmcdc/8.11/caspg3/titlepage.htm>

SAS Institute, Incorporated. "Using the SAS® System Evaluation Tool in SAS 9 Content Assessment." Available at:

https://go.documentation.sas.com/doc/en/pgmsascdc/9.4_3.5/whatsdiff/p0steczr2ay418n1stwuxpykrc8j.htm

SAS Institute, Incorporated. "SAS® 9.4 and SAS® Viya® Functional Comparison." Accessed February 3, 2021. Available at:

<https://www.sas.com/content/dam/SAS/support/en/technical-papers/sas9-4-sas-viya-functional-comparison.pdf>

Sober, Steven and Brian Kinnebrew. "Best Practices for Converting SAS® Code to Leverage SAS® Cloud Analytic Services." Available at:

<https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2020/4147-2020.pdf>

ACKNOWLEDGMENTS

The authors would like to thank our colleagues for their support and assistance while writing this paper. There are too many to mention but the following individuals, in particular, contributed a lot of their technical and analytic insights while writing this paper: Gene Grabowski, Timo Kettunen, Deva Kumar, John Stultz, and Jonathan Walker. Our partners at Carnegie Mellon University's Heinz College (Rema Padman, Alexandra Allen, Anzhi Mou, Zhaoyu Qiao, Harvir Singh Virk, Xiaoyu Zhu) showed us innovative, influential and inspiring work with Python and SAS Viya that served as the foundation for some of the topics in this paper. In addition, as always, Kim Andrews for her tireless efforts with all things SAS!

CONTACT INFORMATION

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

Richard Andrews
Office of the Actuary
Centers for Medicare and Medicaid Services
Richard.Andrews@cms.hhs.gov

Manuel Figallo
Principal Systems Engineer
SAS Institute, Incorporated
Manuel.Figallo@sas.com

Kevin Boone
SAS User Coach
SAS Institute, Incorporated
Kevin.Boone@sas.com

Prakash Subramanian
Senior Technical Architect
SAS Institute, Incorporated
Prakash.Subramanian@sas.com

Logan Perry
Technical Consultant
SAS Institute, Incorporated
Logan.Perry@sas.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.