

SM01

iGET It! Downloading data from a Centralised Repository using PROC HTTP

Nev Cope – Principal Programmer, Data Operations

Phuse Connect 2018, Frankfurt

4th-7th November 2018



4th-7th November
KAP Europa
Frankfurt, Germany

EU 2018

The Clinical Data
Science Conference



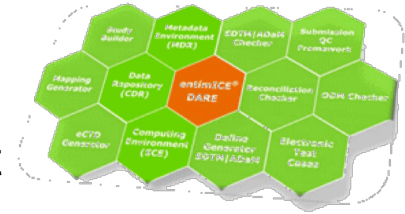
Introduction

New Platforms

- AstraZeneca released two new platforms in 2017.

1. **entimICE-AZ**, a customized version of the “entimICE” product offered by **Entimo**

- A clinical data repository analysis and reporting environment
- Aiming to be the “single source of truth” for all our data
- Has in-built SAS controllers for executing code



2. A high-performance SAS® Grid.

- Load balanced server cluster offering a quicker, faster and smarter way to analyse our clinical data



Harnessing New Ways of Working



- DO focus is on bringing efficiencies to the wider group
- This presentation will discuss the journey I took in connecting these two systems



Common Tasks

- Common ECB tasks:



Statistical simulations



Regulatory reporting (IB, DSUR, etc)



Exploratory analyses

- Exploratory analysis caused some problems to entimICE-AZ users

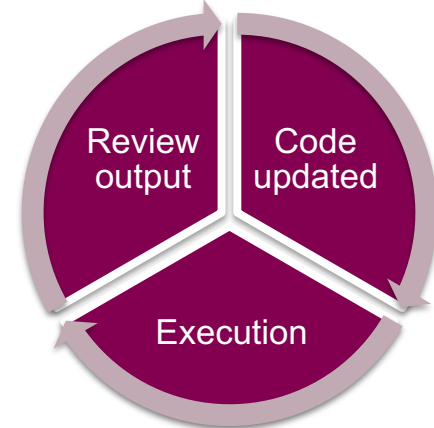


entimICE-AZ is great for batch execution



exploratory analysis
is an interactive process

- entimICE-AZ has only a simple a simple text editor



The Problem

- Exploratory work could be greatly improved if it could be performed on the SAS Grid instead, using SAS Studio or SAS Enterprise Guide.
- However, entimICE-AZ and the SAS Grid are physically located in different data centres with no hard connection between the two.

**If only there was a way to get data from
entimICE-AZ into SAS Grid...**



PROC HTTP and REST

Need a REST?

- After connecting with other users to find a solution I received some R code that interfaced with entimICE-AZ using Web Services.
- This used POST and GET calls using Representational State Transfer (REST), to connect to entimICE-AZ over an internet connection.
- Limited documentation on entimICE-AZ Web Services, so I began to do some research
- Discovered PROC HTTP in SAS



Feeling RESTful

- Each request sent with PROC HTTP also needs to specify a “method”

Method	CRUD Function	entimICE-AZ function
POST	Create	Log in to the system and return a “session ID” for use in future transactions with the server
GET	Read	Download a file, query the contents of a directory or check a session ID is valid/active
PUT	Update	N/A
DELETE	Delete	Removes Session ID from server



Using POST

- The code shown here performs an entimICE-AZ login using PROC HTTP

```
*--- Set FILENAMEs up to receive the PROC HTTP output ---;
filename resp TEMP;
filename headout TEMP;

*--- Request login ---;
proc http
    url = "https://entimicews.servername.net/entimicews/rest/sessions"
    out = resp
    method = "POST"
    headerout = headout headerout_overwrite;
    headers "username"="ABCD123" "password"="PASSWORD";
run;

*--- Read responses ---;
data _resp;
    infile resp length=len;
    length text $1000.;
    input text $varying1000. len;
run;

data _headout;
    infile headout length=len;
    length text $1000.;
    input text $varying1000. len;
run;
```



Logon Successful

Request was successful

_HEADOUT data set	
	text
1	HTTP/1.1 200 OK
2	Server: Apache-Coyote/1.1
3	Content-Type: text/plain
4	Content-Length: 36
5	Date: Wed, 01 Aug 2018 13:59:07 GMT
6	

_RESP data set	
	text
1	a05e4989-61ba-406c-a823-ba63e2a0ab09

Session ID associated with the new logged-in session.

- Returned session ID used to authorise future transactions
- Note that, for subsequent examples, this session ID has been assigned to a macro variable **&session**.



GET

- The GET method will download files and datasets

```
*--- Set FILENAMEs up to receive the PROC HTTP output ---;
filename _headout TEMP;
filename _outfile "$files/sasuser.v94/test.sas7bdat";

*--- Request SAS dataset ---;
proc http
  url          = "https://entimicews.servername.net/entimicews/rest
                 /system?path root/cdar/common/ecd/test.sas7bdat"
  method       = "GET"
  headerout    = _headout
  out          = _outfile;
  headers      "Session-ID"="&session." "Accept"="application/octet-stream";
run;
```



DELETE

- The DELETE method removes the session token from the server

```
*--- Set FILENAMES up to receive the PROC HTTP output ---;  
filename resp TEMP;  
filename headout TEMP;  
  
*--- Issue the Webservice DELETE command ---;  
proc http  
    url=" https://entimicews.servername.net/entimicews/rest/sessions/&session."  
    method = "DELETE"  
    headerout = headout headerout_overwrite  
    out=resp;  
    headers "Session-ID"="&session." "Accept"="application/json";  
run;
```



It Works...But Is It Secure?

- REST and PROC HTTP download files
- By downloading the data in this way to the SAS Grid I'd just violated several internal data security policies
- All regulatory work needs to be done in entimICE-AZ
 - BUT exploratory work is temporary, typically using small data sets



Data should only be downloaded to SASWORK



Macrotisation

Building a Suite of Macros (1)

- Created small utility macros created from the PROC HTTP tests



Setup

- Store the server names (dev/test/prod)



Security

- Log in
- Determine if a session ID is active
- Log out



Download

- Query the contents of a remote folder
- Download a single file
- Download all files from a folder



Building a Suite of Macros (2)

- Smaller macros were chained together to form larger macros
- These were well received
- One frequent request was:
“can we do our development for regulatory work on the SAS Grid as its easier”

This led to some more brainstorming....



%ecb_entimice_dir_list

- Log in
- Return the contents of a directory, recursively if required
- Log out



%ecb_entimice_download_ds

- Log in
- Create a subdirectory in WORK
- Download selected datasets to WORK subdirectory
- Assign a libname
- Log out



%ecb_entimice_libname

- Log in
- Create a subdirectory in WORK
- Download ALL datasets to WORK subdirectory
- Assign a libname
- Log out



I Get It!
...or more accurately “iGET it!”

Working Cross Platform

- Can the same piece of code to execute on both platforms in the same way?
- Our biggest issue was that entimICE-AZ is a metadata-driven system.
- There is metadata detailing:
 - Data storage location
 - Output destinations
 - Logging area
 - Study numbers and other title/footnote text strings
- These would need to be mocked for code running on SAS Grid.



A Typical Program

- A typical SAS program consists of 5 steps/concepts:

1. Initialisation
2. Setting up utility macros
3. Data acquisition
4. Analysis/manipulation
5. Output (data set or RTF)

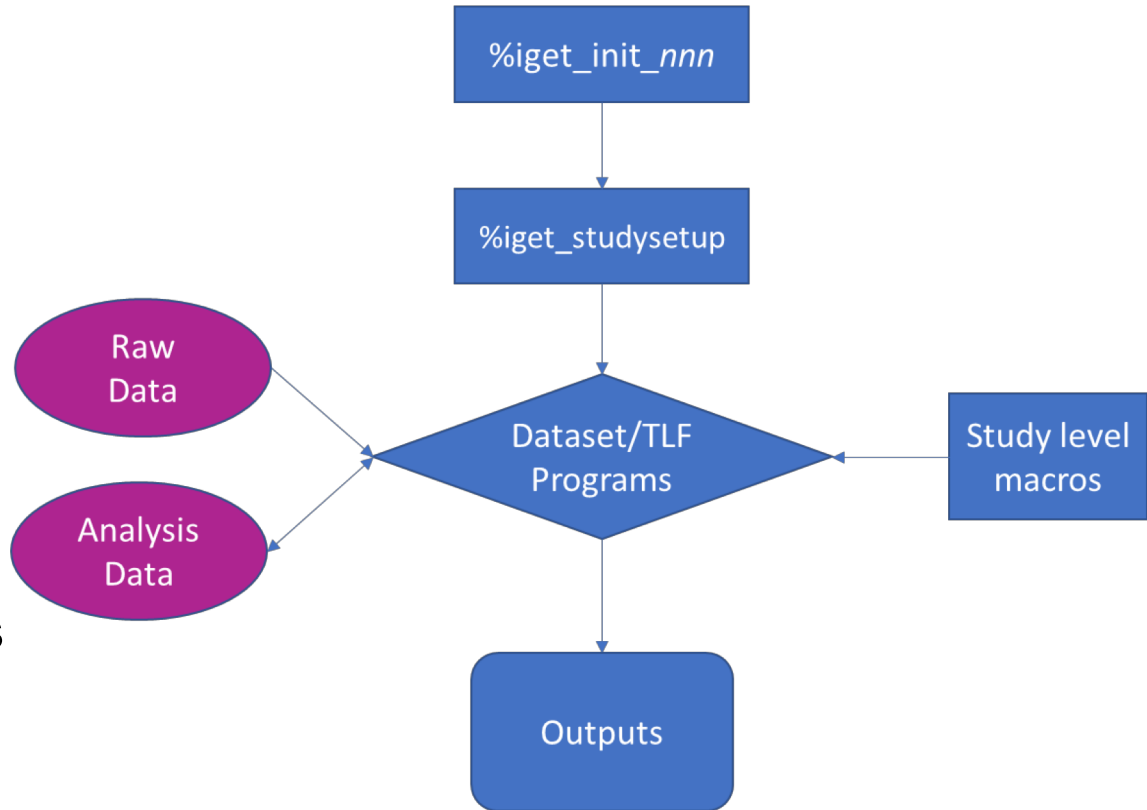
- 2, 3 and 5 automated in entimICE-AZ, could be simulated on SAS Grid
- Both systems running SAS 9.4M5, so no changes needed for point 4

This leaves the key ingredient, point 1 – the initialisation file.



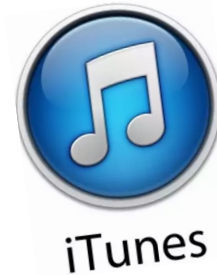
The Workflow

- We ran through several prototypes
- A couple of study teams applied the final prototype to real study work
- Feedback gathered was very useful, highlighting many unanticipated issues



iGET?

- Why did we call it iGET?
- Lots of cool things that start with “i”...



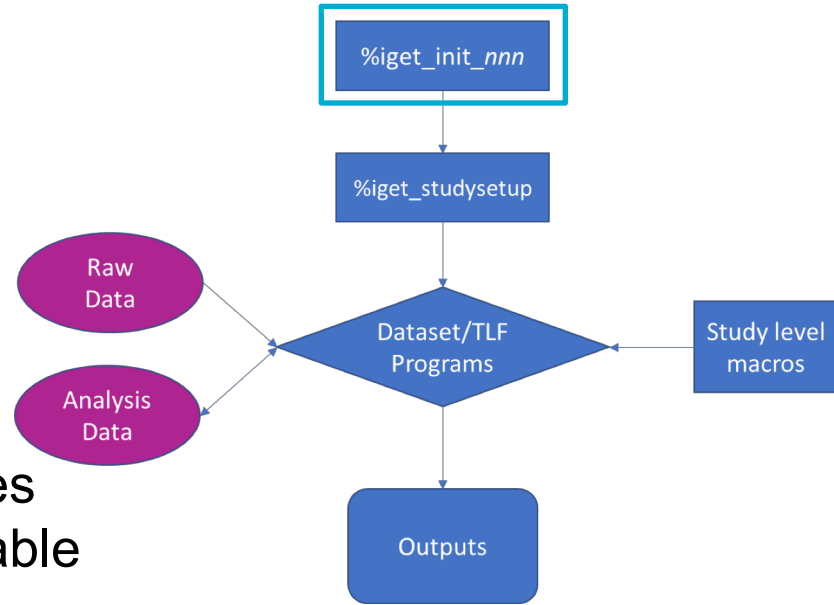
- ...and the original macros were there to “get” the data

Integration of **Grid** and **EntimICE Tool**.



%iget_init

- All Programs begin %iget_init.
- Macro sits at a global level.
- It performs 4 main functions:
 - Determines the platform
 - Updates SASAUTOS
 - Mirrors global system macro variables
 - Makes any study-level macros available

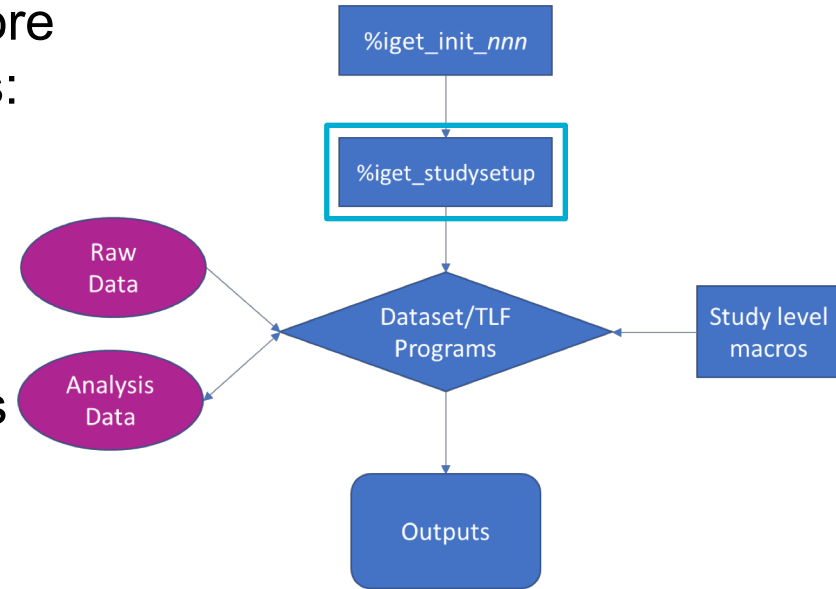


- After execution the SAS environment on both systems will be identical



%iget_studyssetup

- The **%iget_studyssetup** macro adds more study-specific information. This includes:
 - General SAS options
 - Output font sizes
 - Format catalogs
 - Titles/footnotes
 - Data reporting period or cut-off dates
 - Global macro variables
 - SAS Grid output location

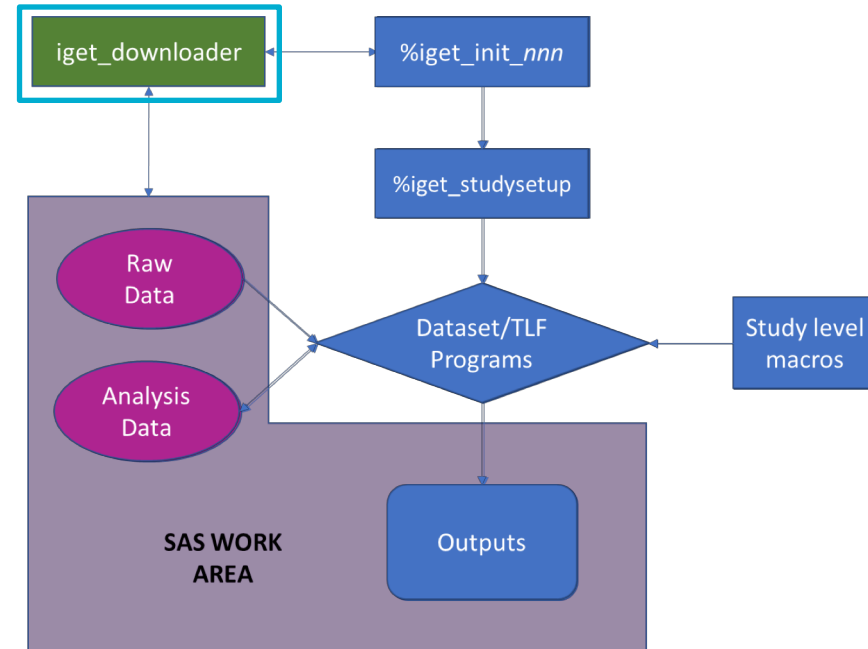
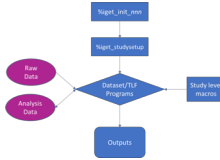


- There can be multiple **%iget_studyssetup** macros for each project



iget_downloader

- For SAS Grid only.
- **iget_downloader** contains entimICE-AZ data download calls.
- Is called FIRST in a SAS Grid programming session
- Multiple **iget_downloader** programs for any given project.
 - Specific deliverable (IB, DSUR, etc)
 - Exploratory analysis
 - User specific



The Finished Product

- A workflow that:



Acquires data from central repository



Is cross platform



Follows our data security guidelines



Enables both interactive development and batch execution



Is customisable



Conclusions

Conclusion

- I've shown you what was delivered, but what was my journey?



- Traversed a few technological obstacles, but a robust procedure was established.

There will be many hurdles but don't let these bring you down, you *can* find solutions



Questions?