abbvie

CALLING R FUNCTIONS FROM SAS
INTERWORKING BETWEEN DIFFERENT SERVERS

Stefan Englert
AbbVie Inc., Data and Statistical Sciences

PhUSE Single Day Event, 2017
North Chicago, IL, USA

abbvie

Stefan Englert is an employee of AbbVie, Inc. The support of the presentation was provided by AbbVie. AbbVie participated in the review and approval of the content.

Crossing the Technical Frontier

Possibilities that exist in the programming language R:

- ✓ R provides efficient statistical features and comprehensive graphical output
- ✓ There are packages available to use in R that can help shorten the time to market
- ✓ Regulatory agencies are starting to accept submissions using R in addition to SAS

Scope of this talk:

Build synergies: Integration of R functionality into SAS;
Technical realization of 'Calling R Functions from SAS'.

Calling R Functions from SAS

PhUSE 2011, Paper CS 07

Calling R Functions from SAS

Dr. Peter Bewerunge

The article describes how R functions can be called directly through a built-in SAS procedure called PROC IML.

Functionality:

- ✓ Submit R Code from SAS
- ✓ Send SAS datasets to R
- ✓ Send R datasets to SAS
- ✓ Error handling (basic form)

Bewerunge, P. Calling R Functions from SAS PhUSE, 2011, 1-7

SAS interface to R realized trough PROC IML

Limitations:

- The R software comes along with absolutely no warranty (critical especially for the large amount of add-on-packages)
- The use of large data sets is limited. In R / SAS IML the data are loaded and processed in the main memory.
- Every data type has to be converted either to a *data.frame* or a *matrix* before transferring data to SAS.

Bewerunge, P. Calling R Functions from SAS PhUSE, 2011, 1-7

SAS interface to R realized trough PROC IML

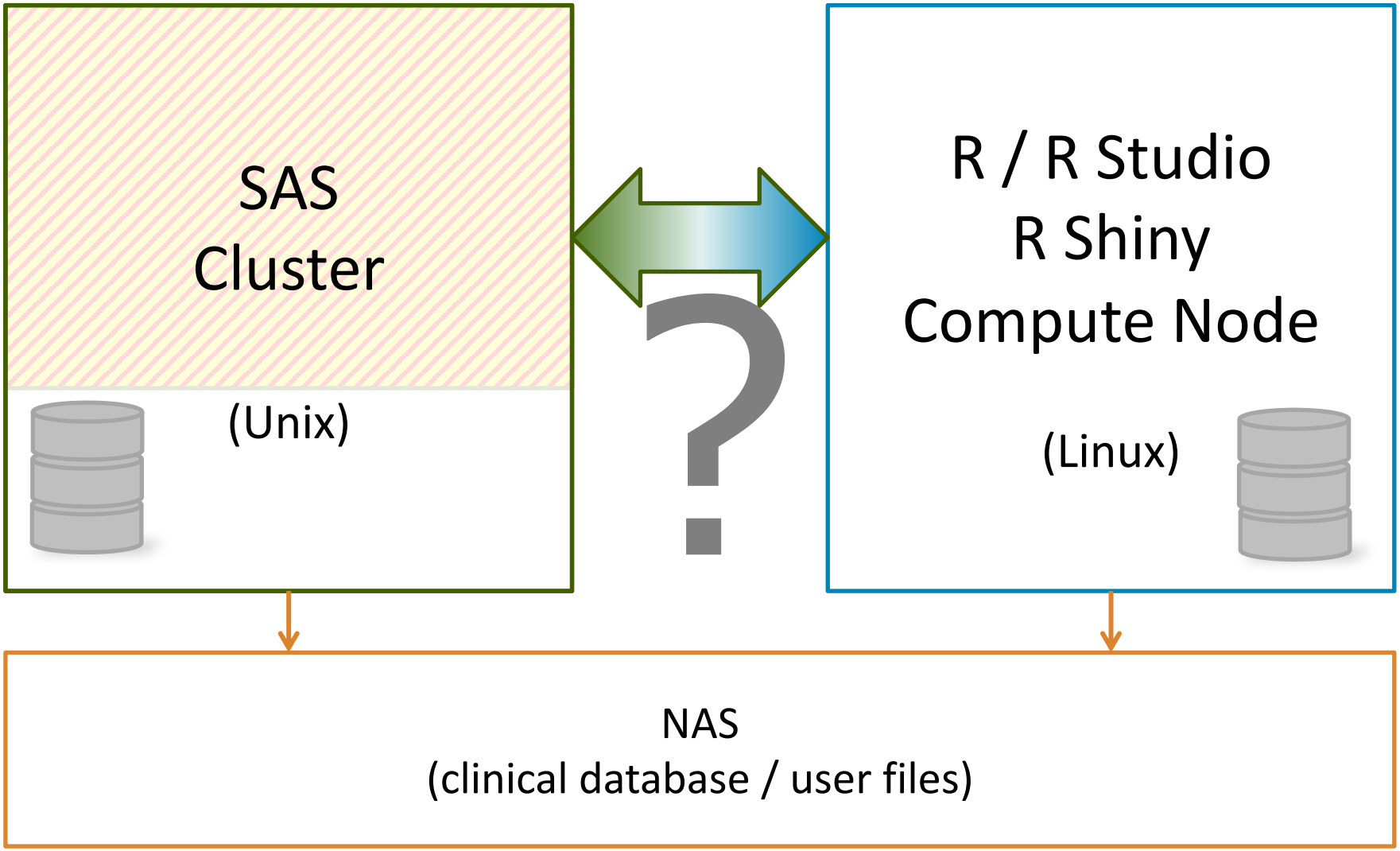
Key Requirement:

— SAS and R must be installed on the same computer

- From an infrastructure integrity and maintenance perspective, it is desirable to keep SAS and R separate on different servers.
- Especially true, if either or both run in a validated environment.
- SAS and R on different servers allow easier scalability; independently for SAS and R

Summary: A direct calling of R functions through the IML procedure in SAS may not be the preferable option.

Infrastructure Scenario



Solution

%RUNANYR (SAS macro)

Functionality:

- ✓ Submit R Code from SAS
- ✓ Send R datasets to SAS
- ✓ Error handling

Minimal Example – Code

MIN.sas
SAS System Program
340 Bytes

MIN.sas *

```
*START;
%RUNANYR(home, MIN.R);

PROC PRINT
     DATA=home.mydata;
RUN;

QUIT;
```

MIN.R
R-Datei
156 Bytes

= User Input

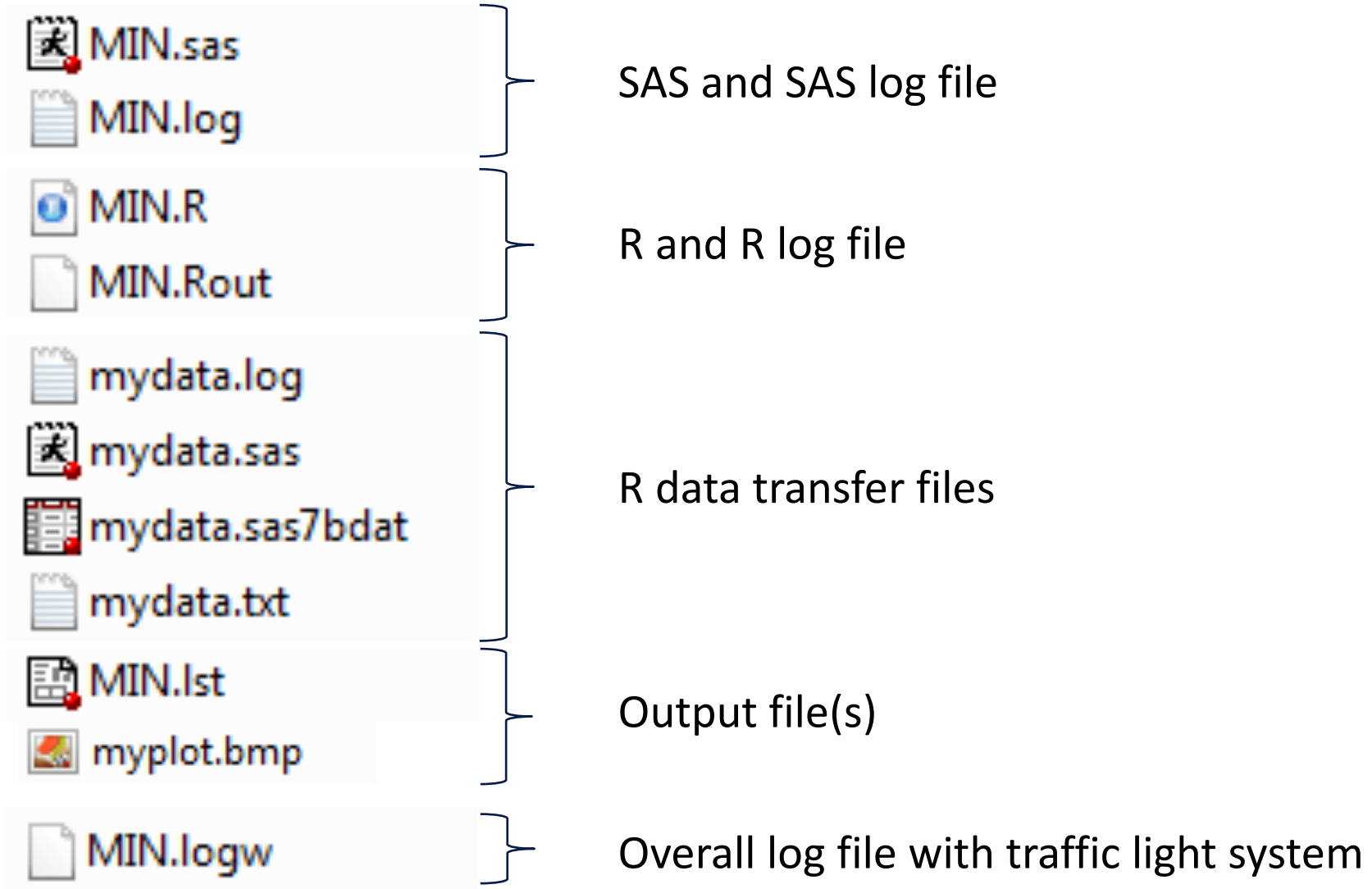
MIN.R*

Source

```
1  #START
2  library(foreign)
3
4  x <- data.frame(1:10)
5
6  write.foreign(x,
7    "/home/mydata.txt",
8    "/home/mydata.sas",
9    package="SAS")
10
11 bmp(filename=
12   "/home/myplot.bmp")
13   plot(x[[1]])
```

1:1 (Top Level) R Script

Minimal Example – Files

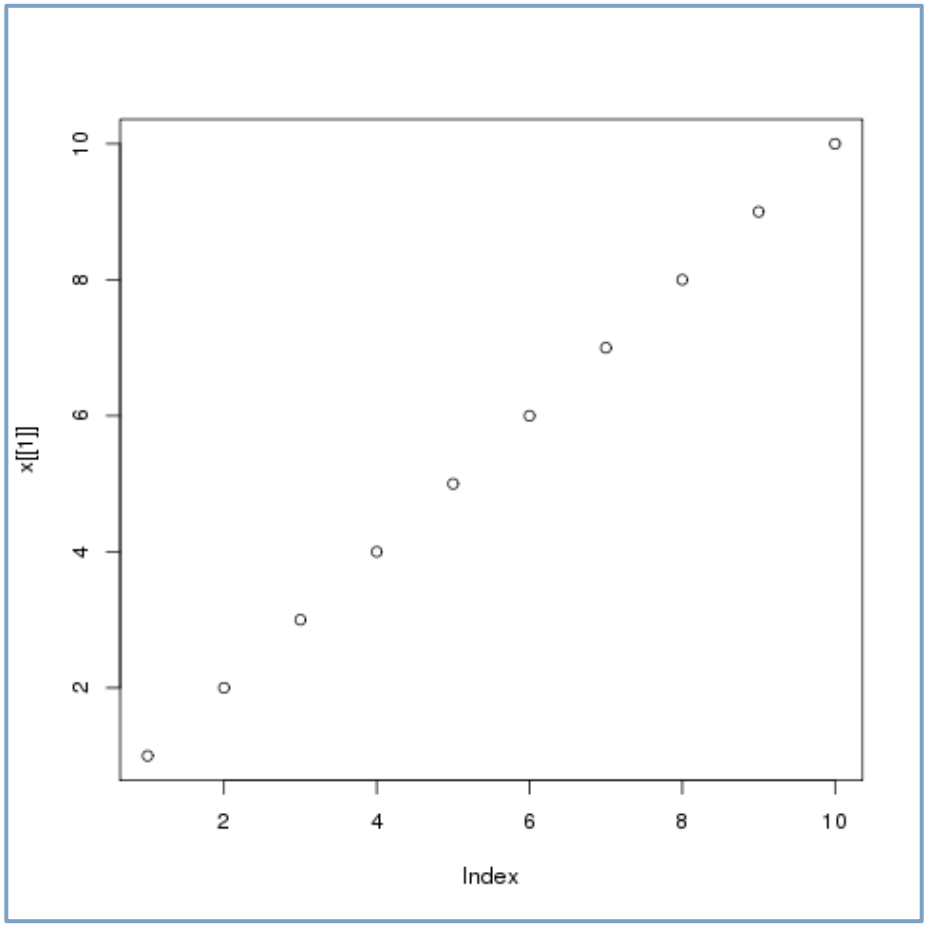


Minimal Example – Output

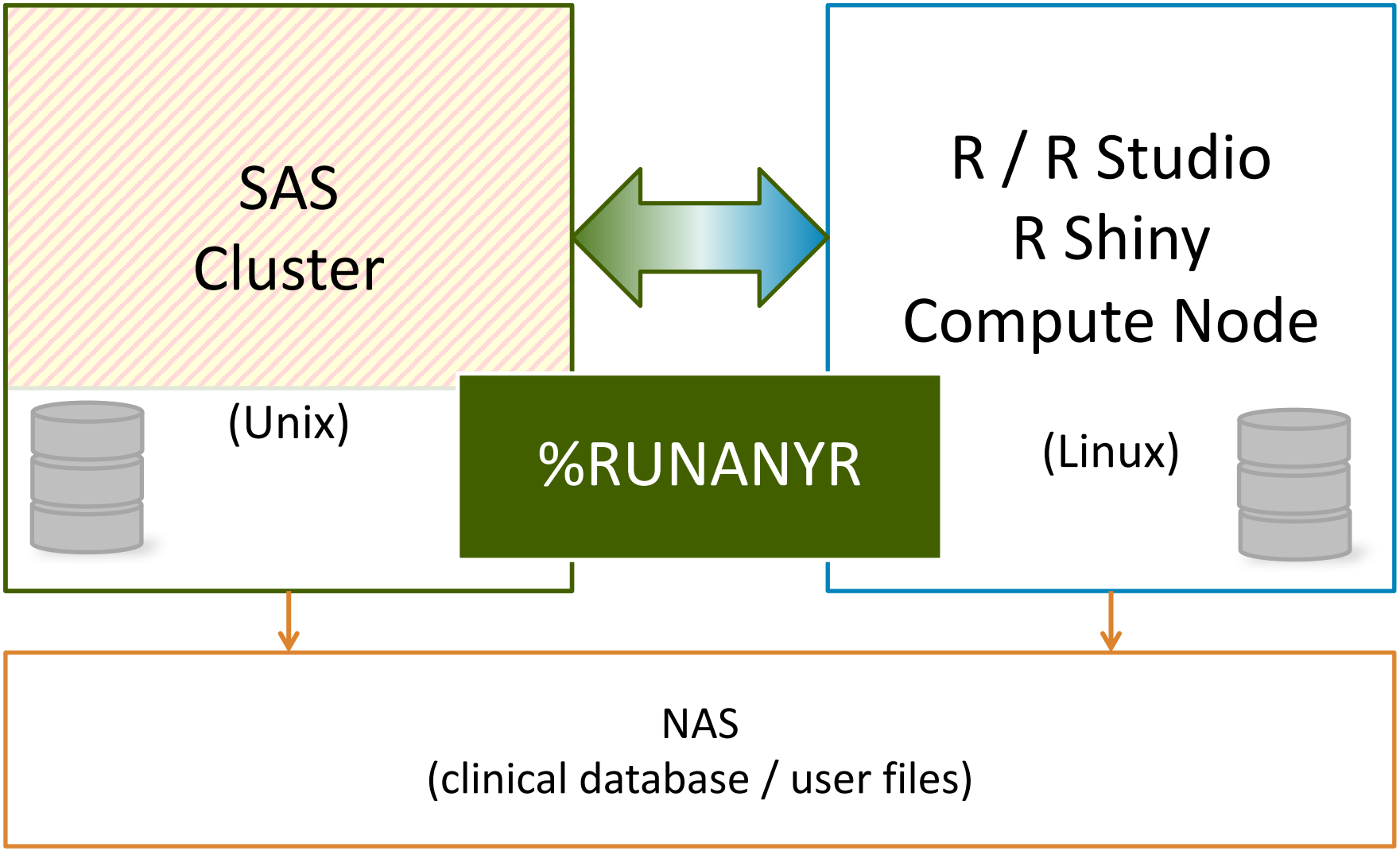
= SAS process
= R process

The SAS System

Obs	X1_10
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10



Infrastructure Scenario



Deep dive into %RUNANYR

%RUNANYR (SAS macro)

```
%MACRO RUNANYR(folder,program,sasversion=9.4);
DATA _NULL_;
CALL SYSTEM("sas930 /app/cs/sw/bin/run_any_r.sas -sysparm %str(%"&folder, &program, &sasversion%")");
RUN;
%MEND RUNANYR;
```

Deep dive into %RUNANYR

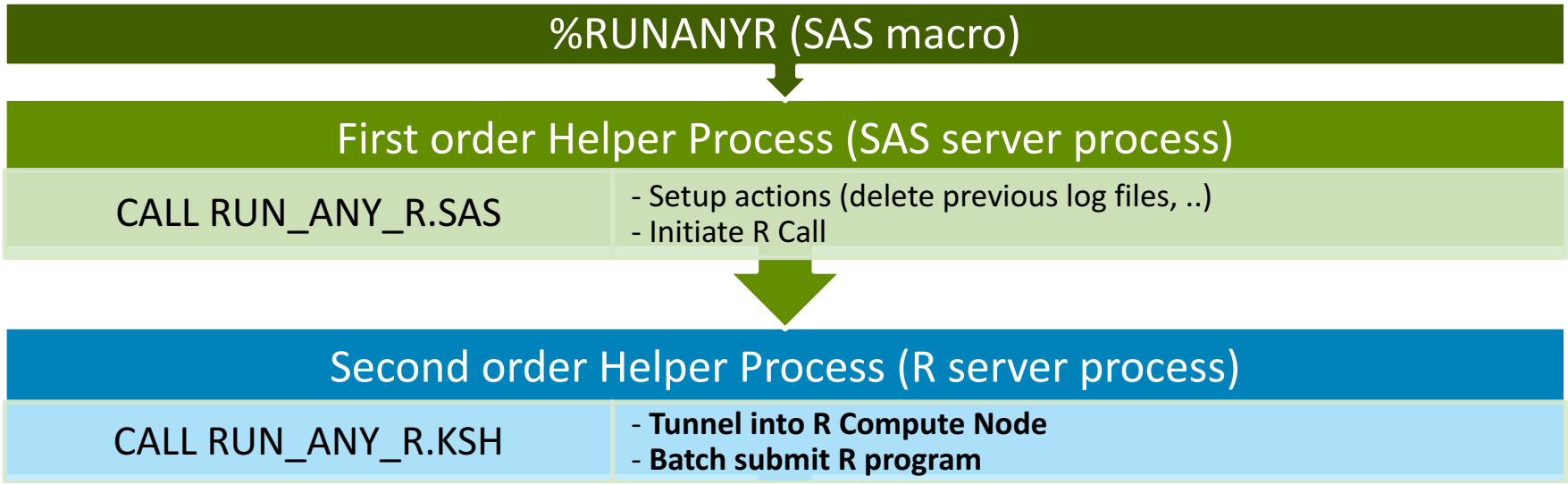
%RUNANYR (SAS macro)

First order Helper Process (SAS server process)

CALL RUN_ANY_R.SAS	- Setup actions (delete previous log files, ..) - **Initiate R Call**

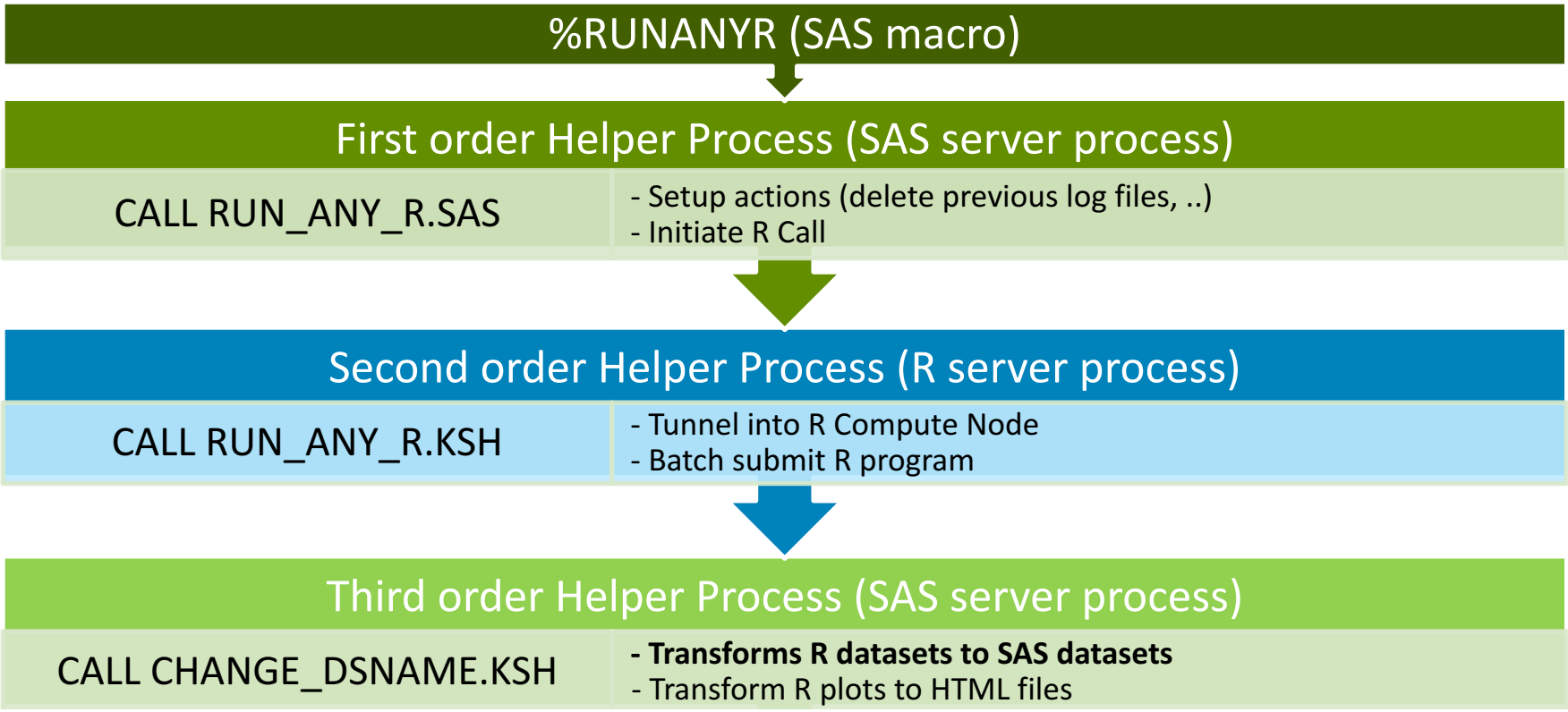
```
DATA _NULL_;
 call system(". run_any_r.ksh &r_dir &r_prog");
 RUN;
```

Deep dive into %RUNANYR



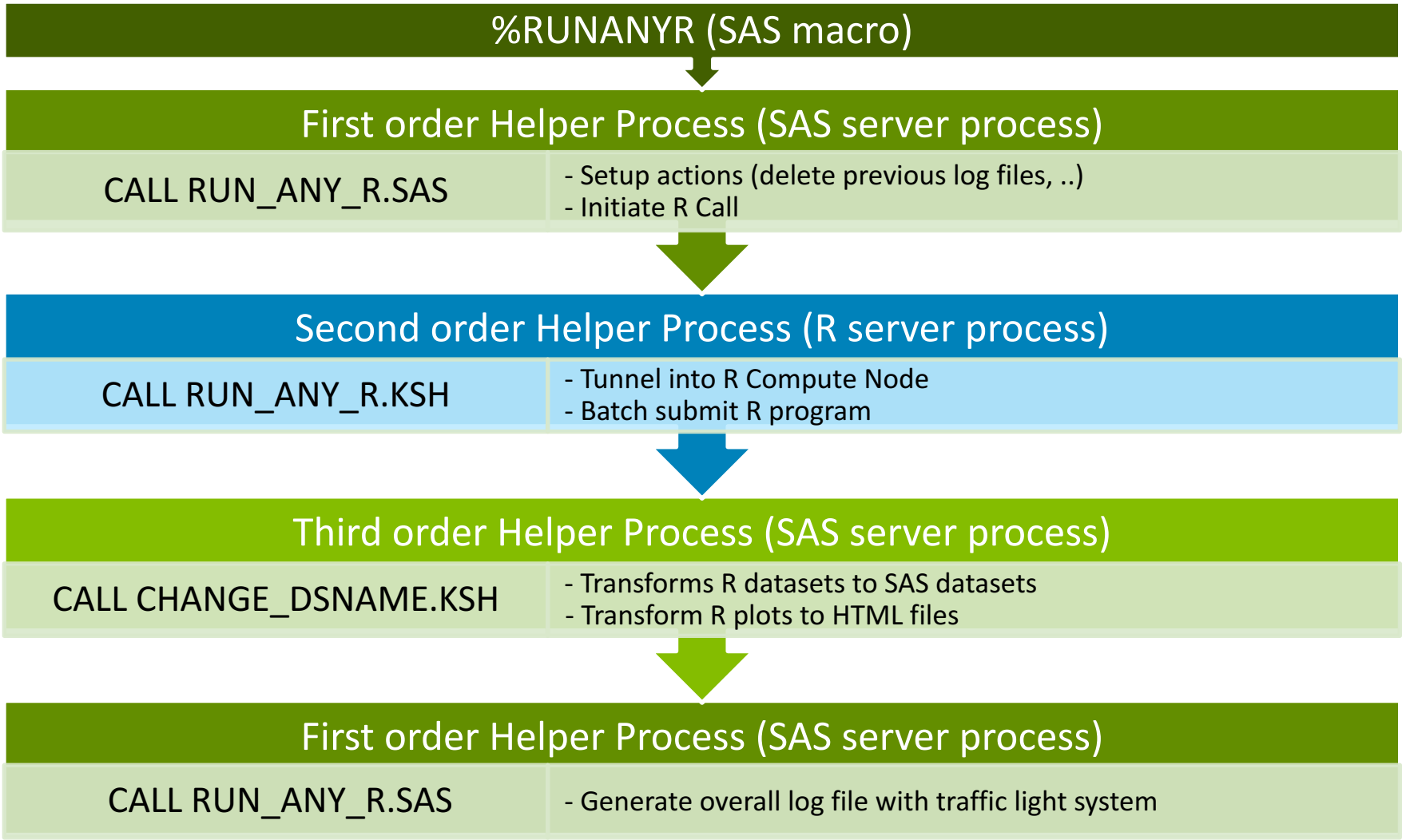

```
ssh rserver.abbvienet.com 2>/dev/null << EOT
cd $1
R CMD BATCH $2
EOT
```

Deep dive into %RUNANYR

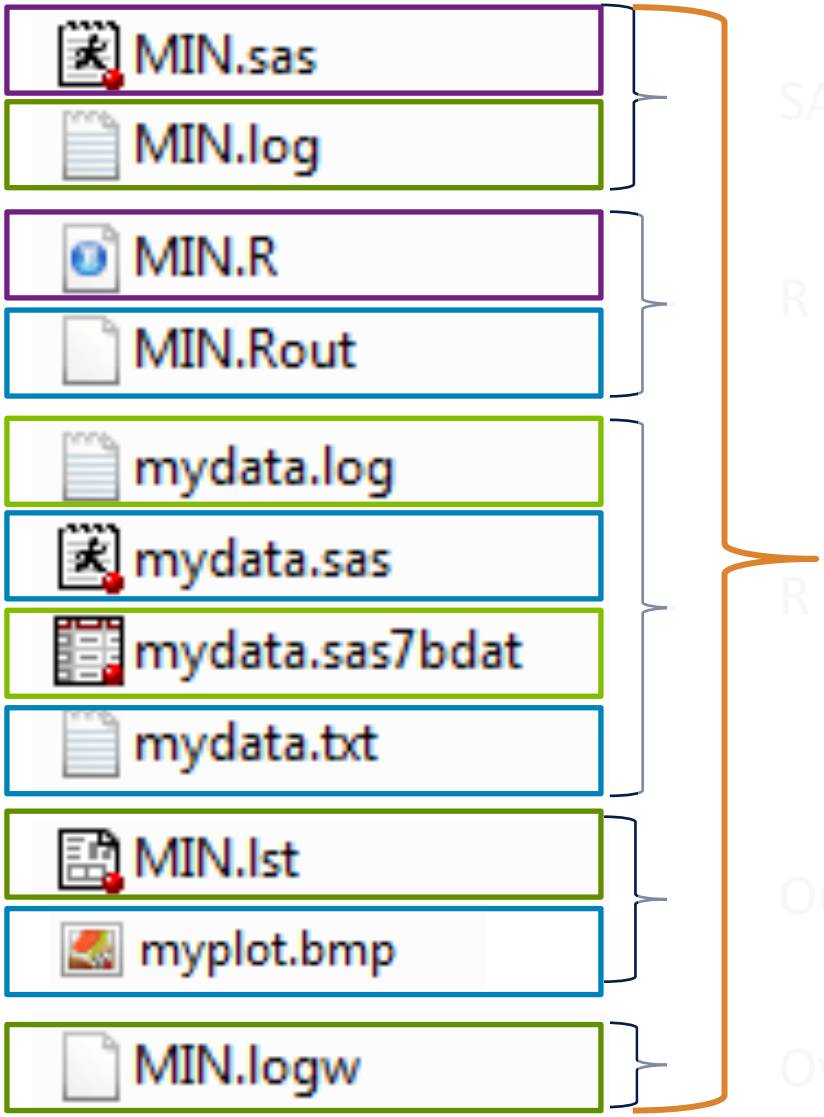



```
for file1 in $sas_progs
 do
    $sasperl $dir_name/${sas_name}.sas
 done
```

Deep dive into %RUNANYR



Revisiting the Minimal Example – Why does it all work?



= User Input
= SAS process
= R process
= NAS

Seamless communication between SAS and R ensured as all physical files are located on a common NAS drive.

Summary

Access R through PROC IML

Functionality:

- Submit R Code from SAS
- Send SAS datasets to R
- Send R datasets to SAS
- Error handling (basic form)

Limitations:

- As above

Access R with CALL functions

Functionality:

- Submit R Code from SAS
- Send SAS datasets to R
- Send R datasets to SAS
- Error handling

Limitations:

- As above

Summary

Access R through PROC IML

Key Requirement:

SAS and R must be installed on the same computer/server

Additional factors:

Easy setup and installation

Single file containing both SAS and R code

Access R with CALL functions

Key Requirement:

SAS and R can be located on different servers

Additional factors:

Flexibility in choice of R version

File containing SAS code and multiple files containing R code

Files can be edited with the naïve editors (SAS/RStudio)

Thank you for your attention!

Stefan Englert

stefan.englert@abbvie.com