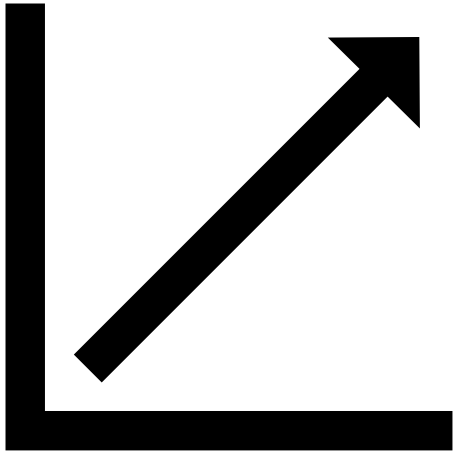
phuse·share
connect·share·advance

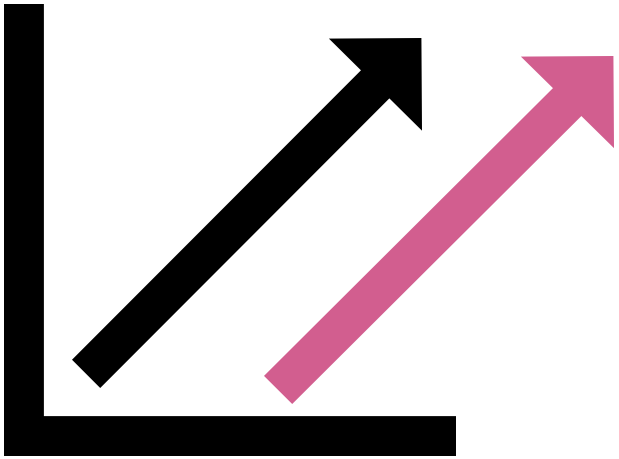
Single Day Event

Re-writing our global validation process for SAS programs using R, Azure DevOps and Git: a case study of code validation in a multi-lingual environment

Matthew Phelps & Nicolai Skov Johnsen
Novo Nordisk

Views and opinions expressed are those of the speakers and not necessarily Novo Nordisk





- Maintenance/Updates

- Maintenance/Updates
- Monolithic code

- Maintenance/Updates
- Monolithic code
- Manual versioning


DROWNING IN VALIDATION
STANDARD PROGRAMS
VALIDATE STANDARD PROGRAM
Gudim.
imgflip.com

- Maintenance/Updates
- Monolithic code
- Manual versioning

- Maintenance/Updates
- ~~Monolithic code~~
- ~~Manual versioning~~

- Maintenance/Updates
- ~~Monolithic code~~
- ~~Manual versioning~~

- Unit-testing
- Modularized code
- CI/CD infrastructure

UNIX

- Unit-testing
- Modularized code
- CI/CD infrastructure

UNIX

R Studio

python™

sas

- Unit-testing
- Modularized code
- CI/CD infrastructure
- Minimal language switching overhead

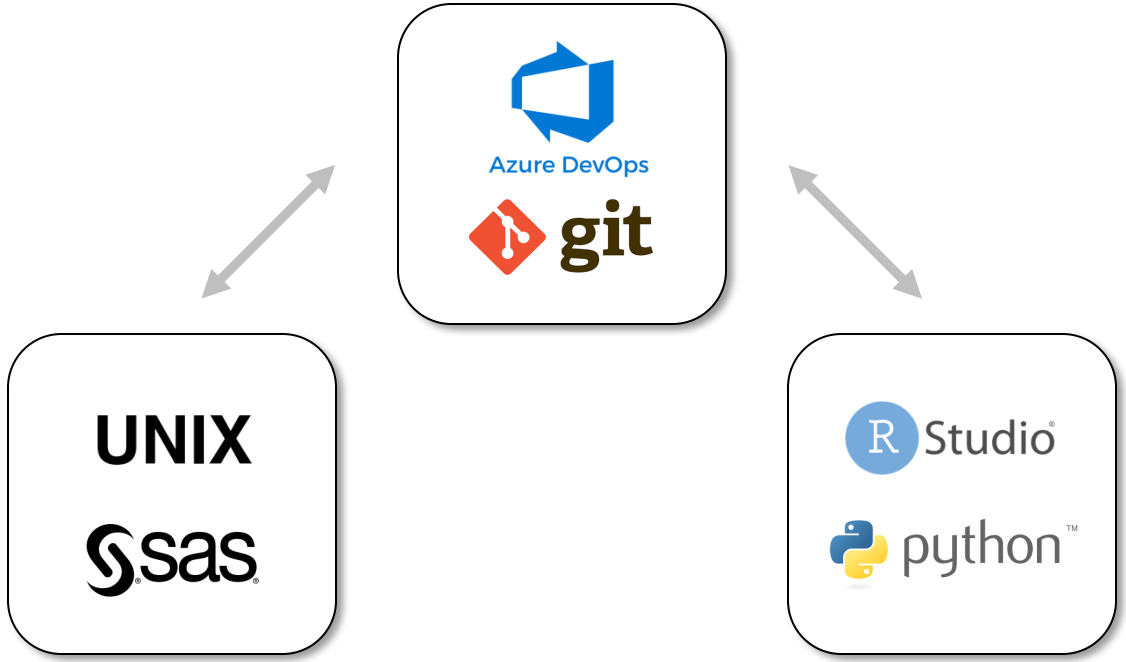
UNIX

- Unit-testing
- Modularized code
- CI/CD infrastructure
- Minimal language switching overhead

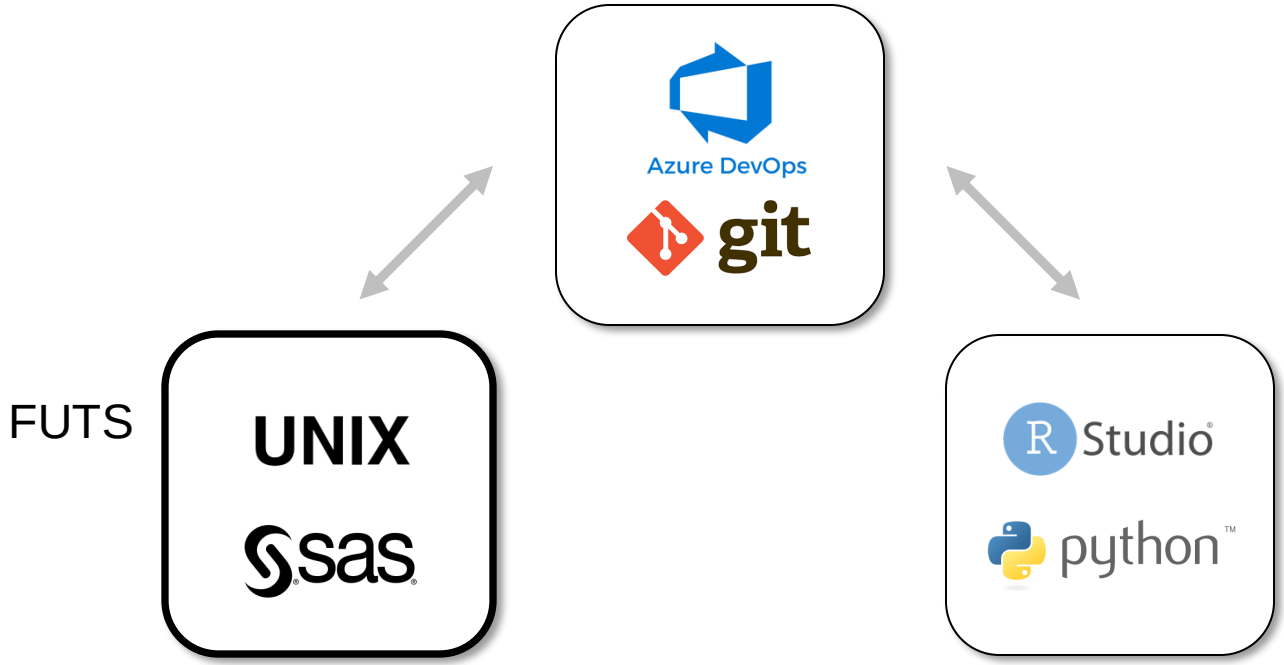
Tech landscape

Tech landscape

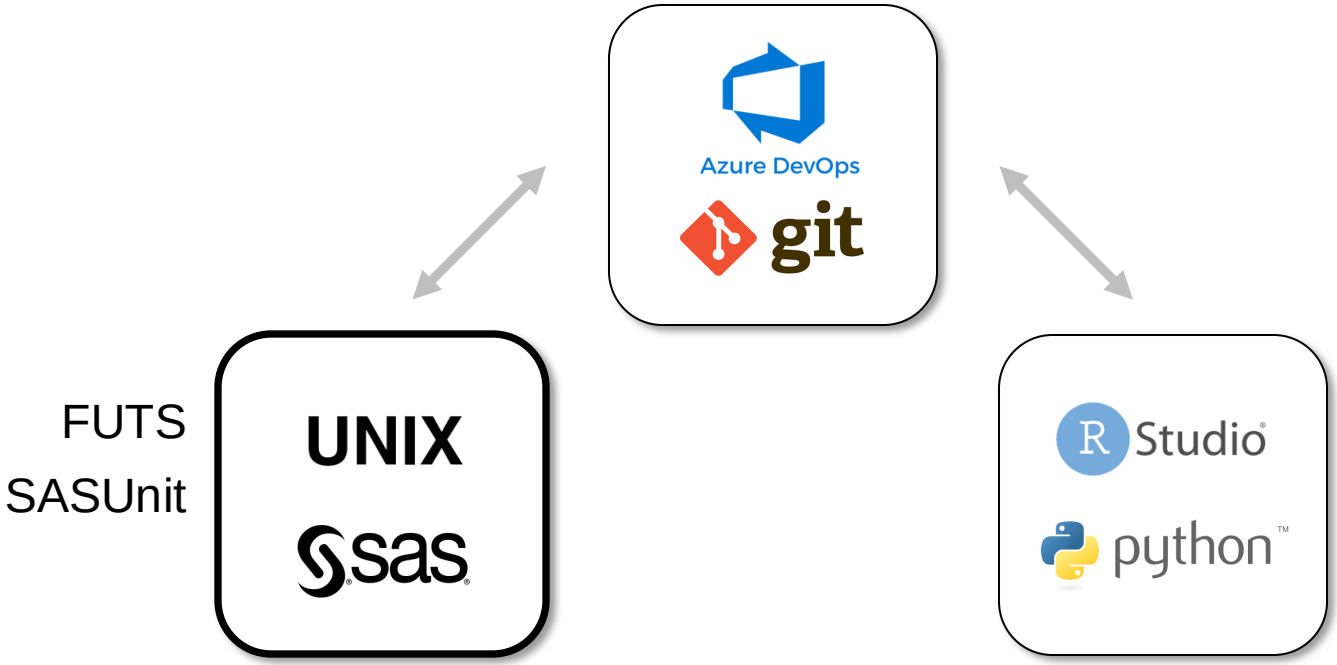
Tech landscape



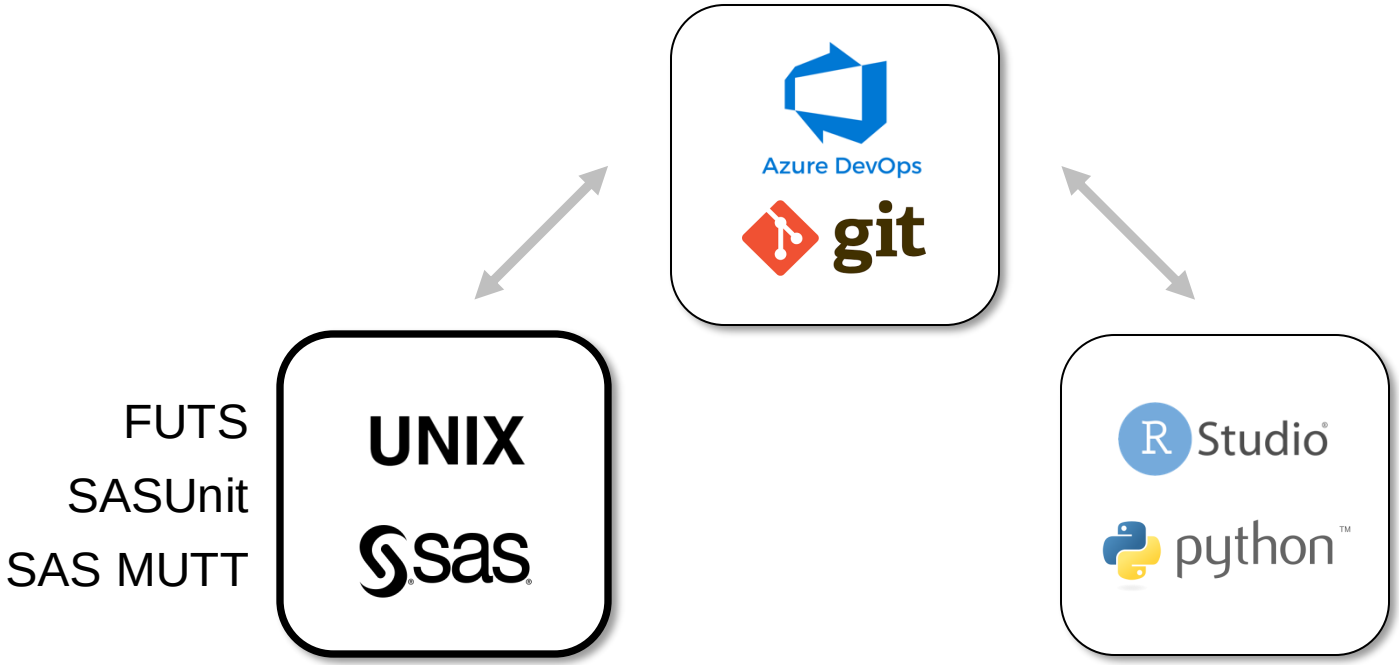
Tech landscape



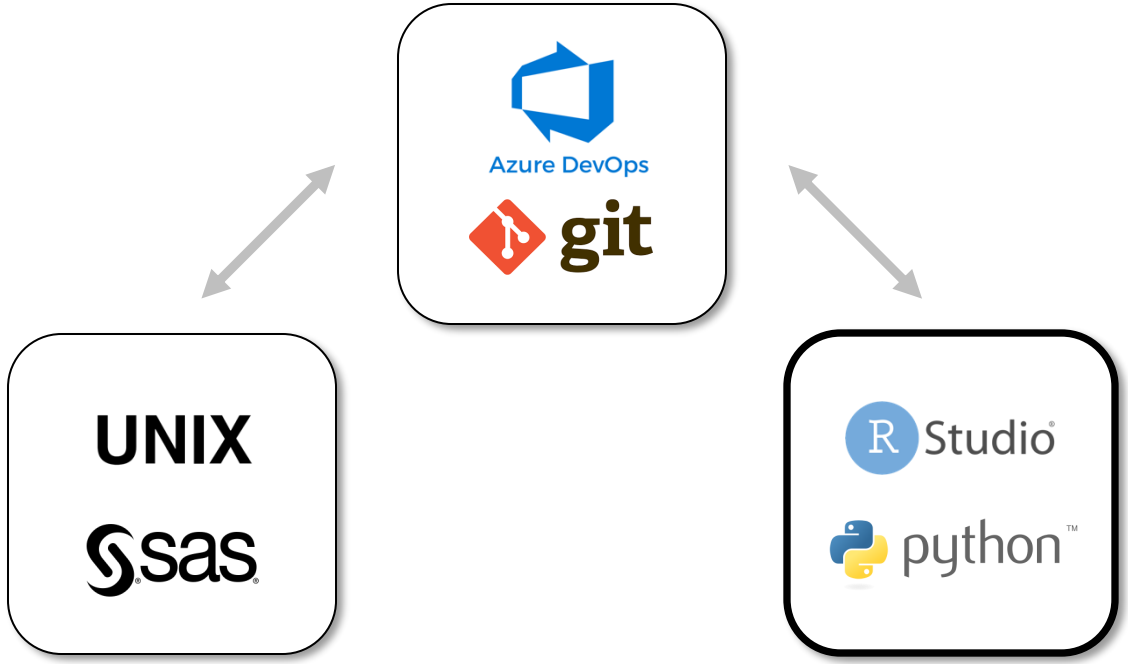
Tech landscape



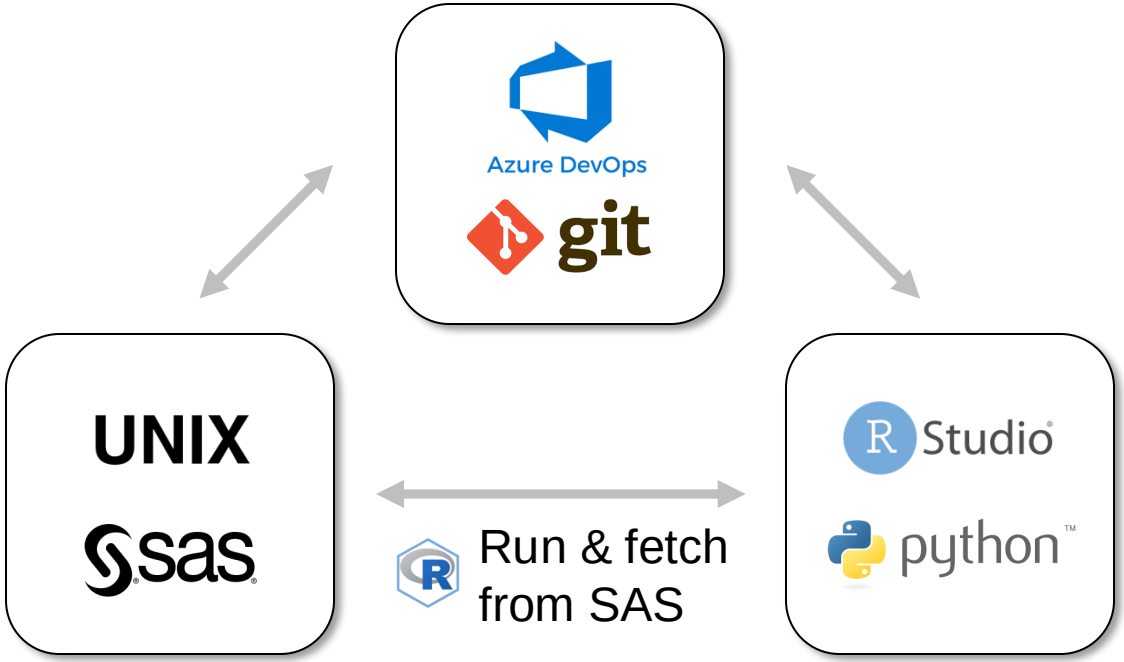
Tech landscape



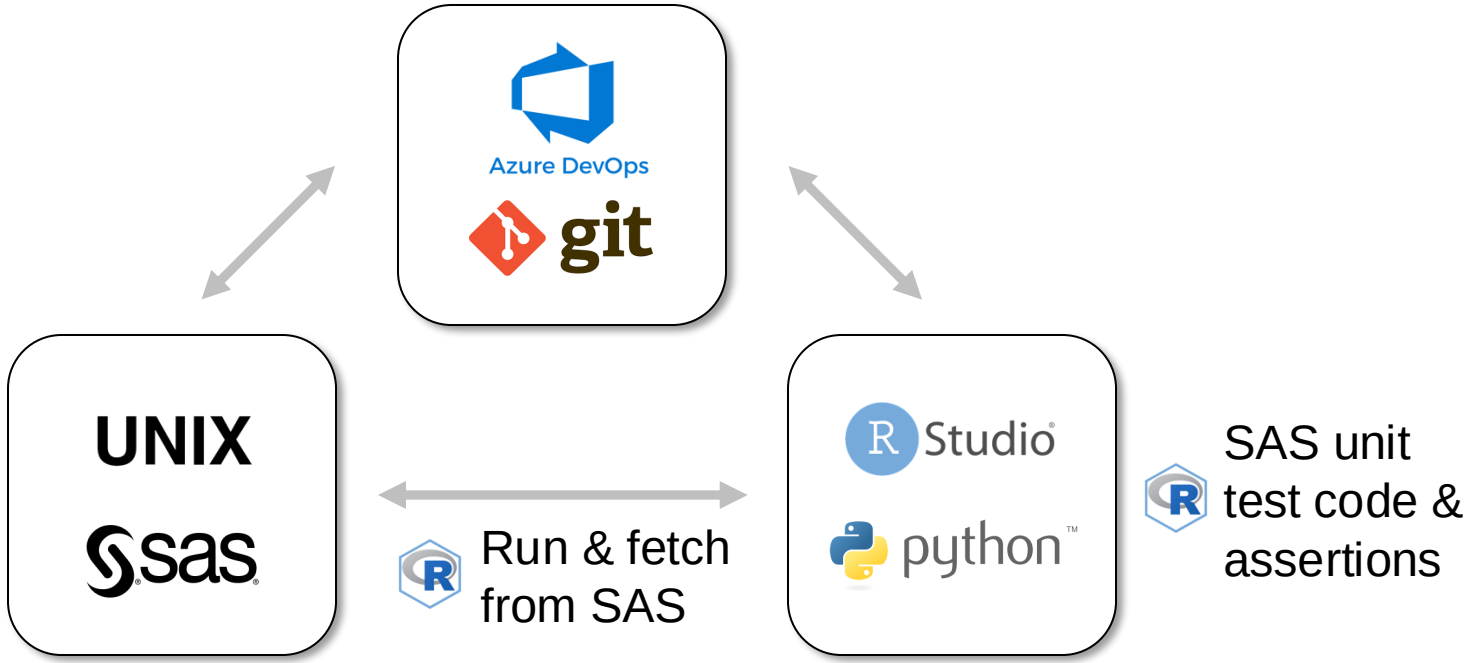
Tech landscape



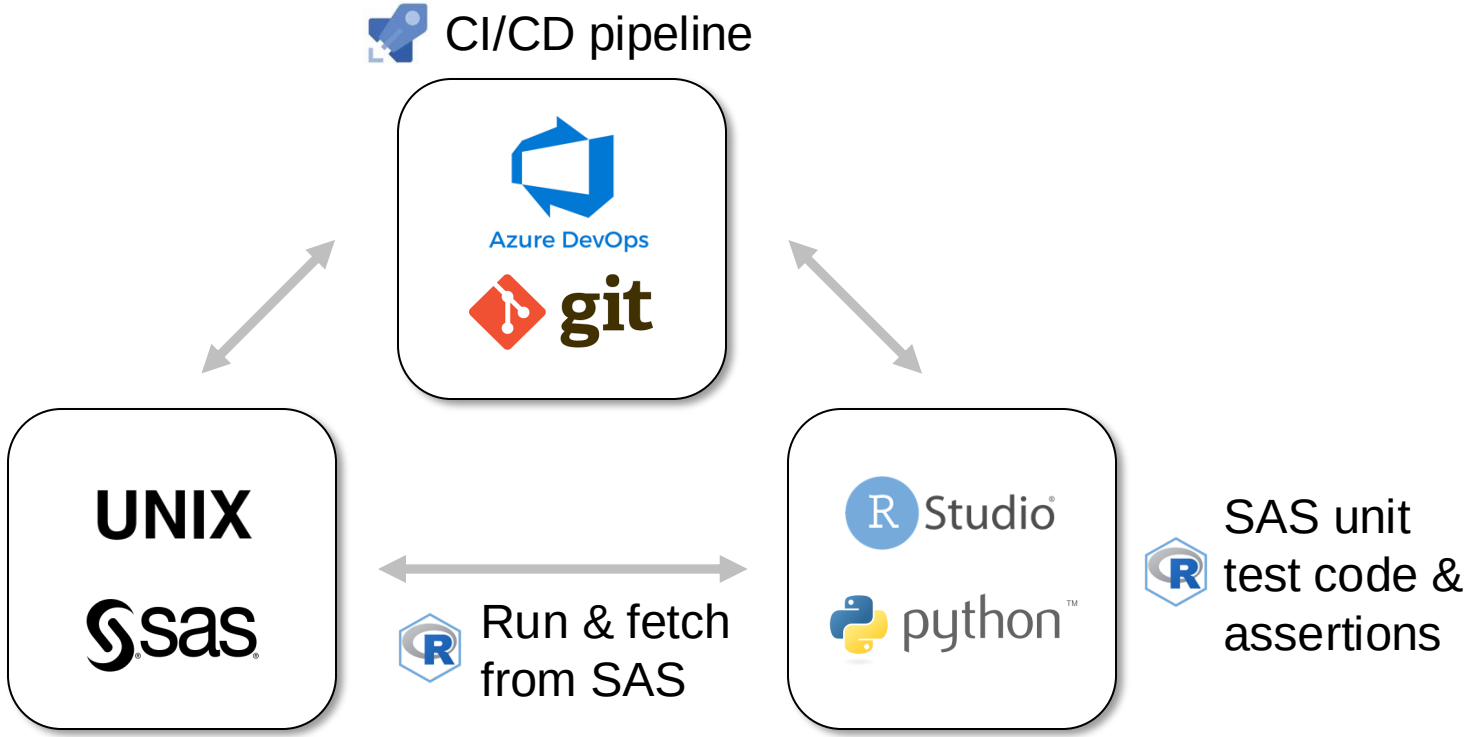
Tech landscape



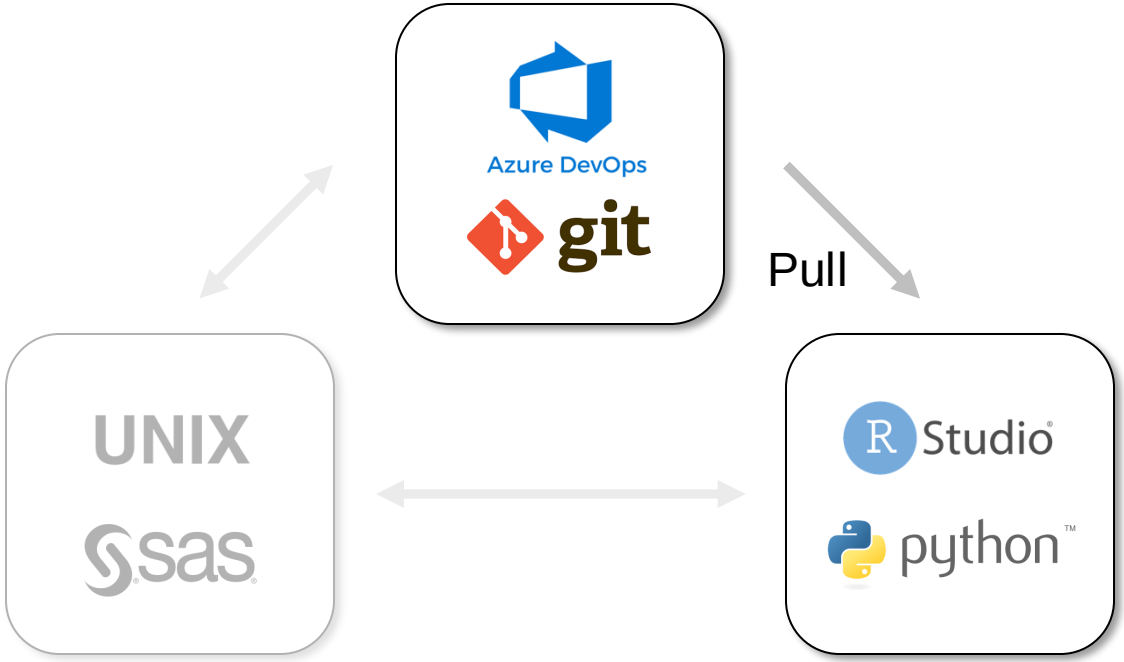
Tech landscape



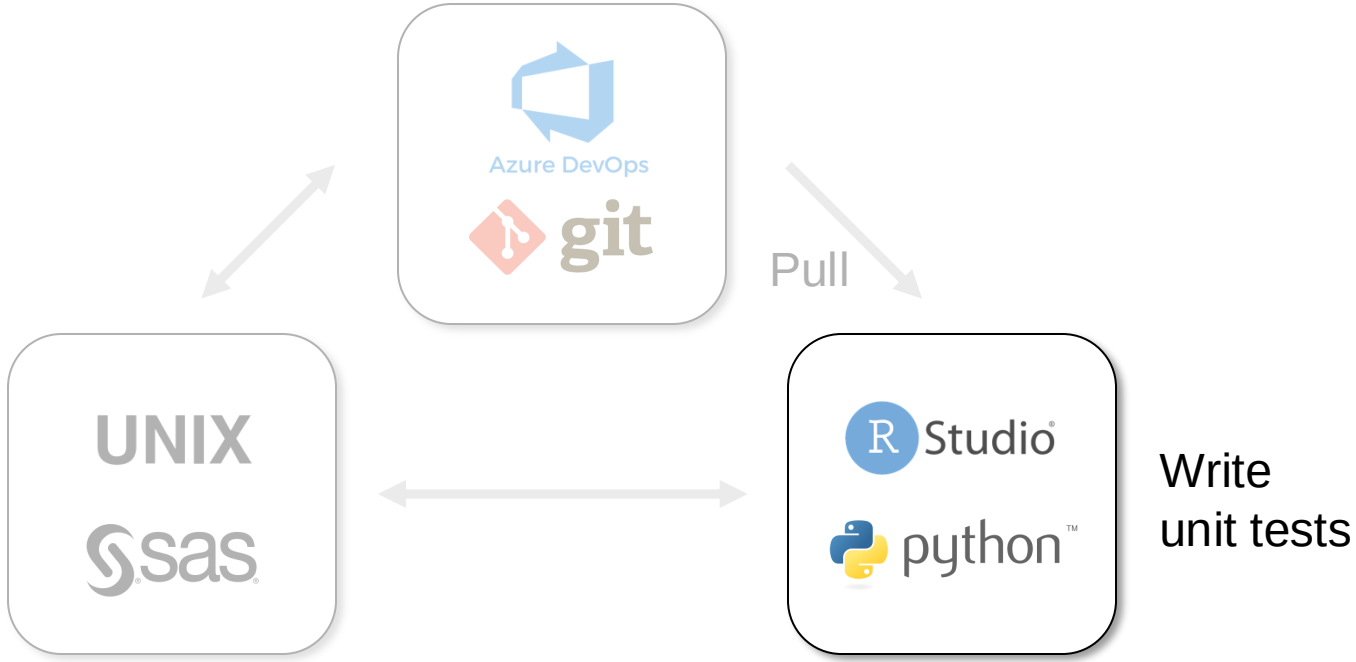
Tech landscape



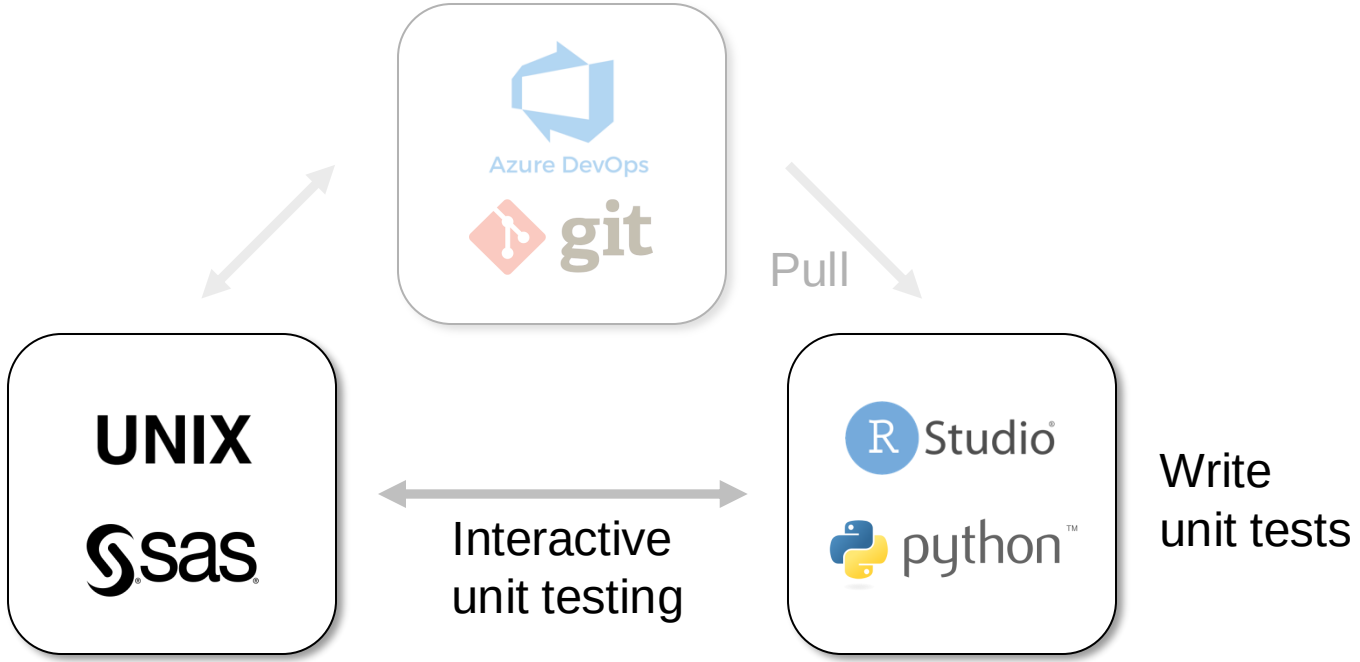
Tech landscape



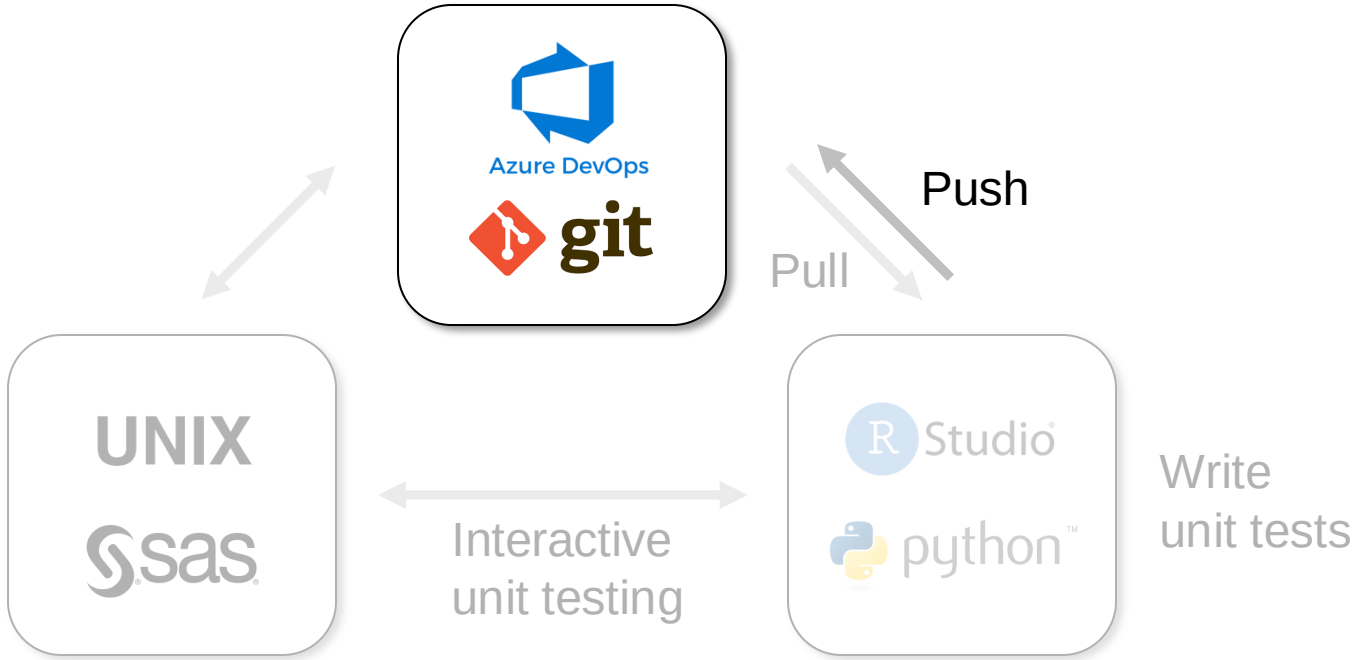
Tech landscape



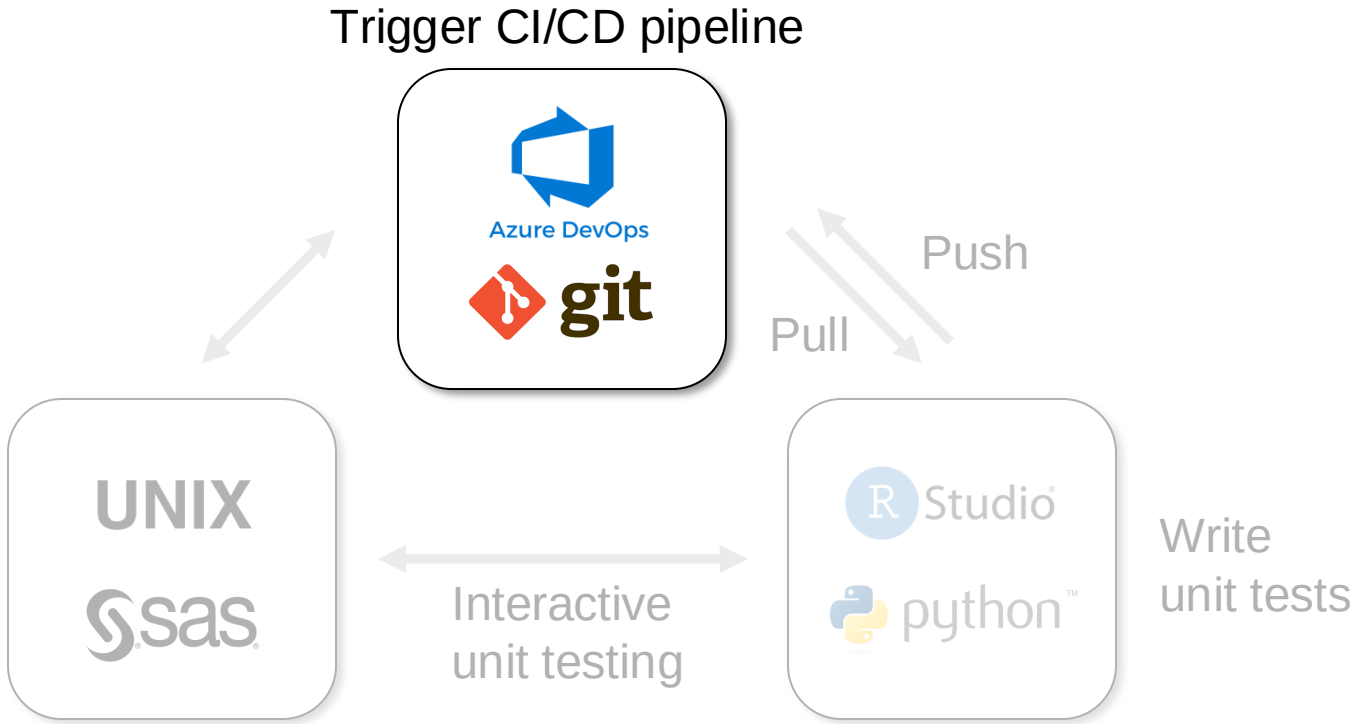
Tech landscape



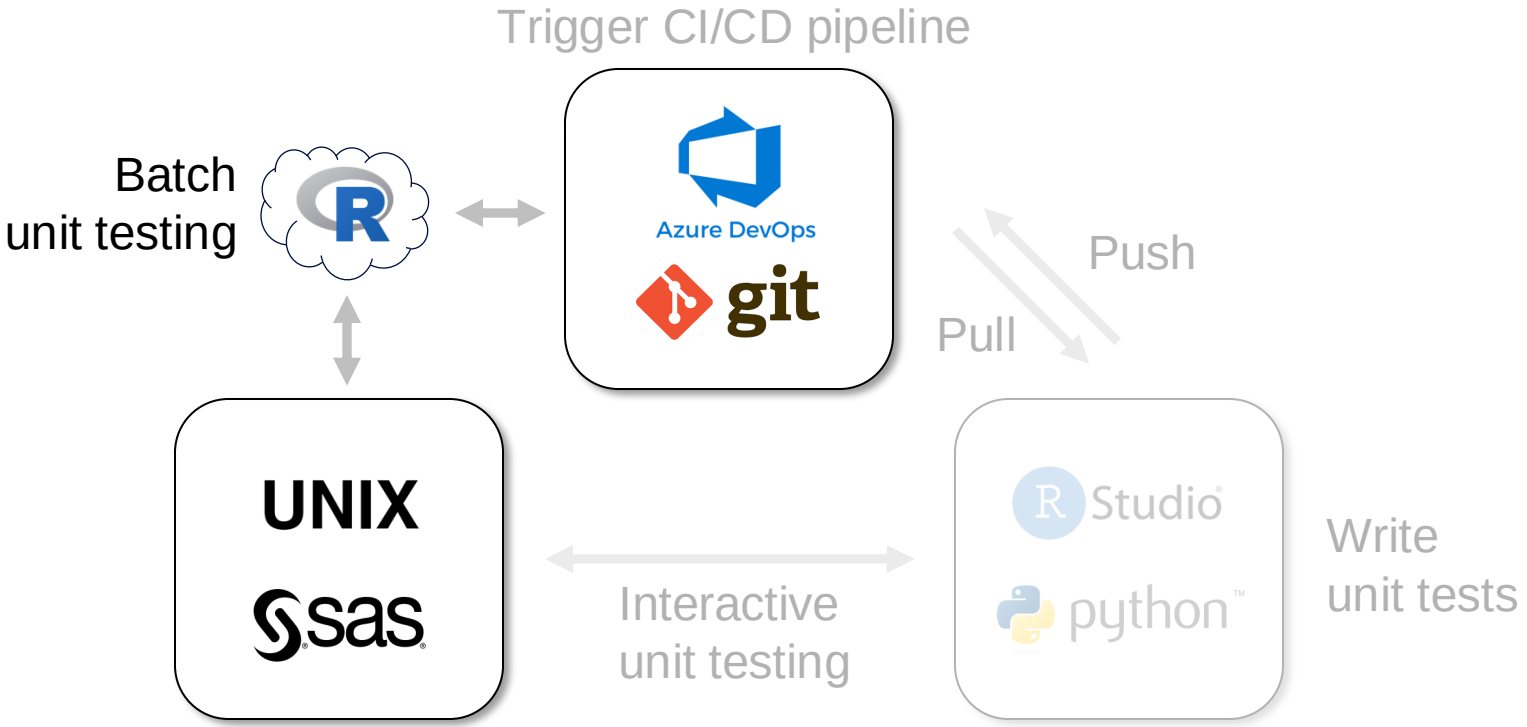
Tech landscape



Tech landscape



Tech landscape



Demo

Code to test

```
%macro update_students(new_students, min_height);

    /* Check that min_height is positive */
    %if &min_height. <= 0 %then %do;
        %put WARNING: min_height should be positive;
    %end;

    /* Combine tables and add session_tag variable */
    data new_class;
        set sashelp.class
            &new_students.;
        where height >= &min_height.;
    run;

%mend update_students;
```

Demo

Code to test

```
%macro update_students(new_students, min_height);

    /* Check that min_height is positive */
    %if &min_height. <= 0 %then %do;
        %put WARNING: min_height should be positive;
    %end;

    /* Combine tables and add session_tag variable */
    data new_class;
        set sashelp.class
            &new_students.;
        where height >= &min_height.;
    run;

%mend update_students;
```

Unit test

```
test_that("update_students - basic test", {
  # SETUP ---------------------------------------------------------------------

  # Test data
  data_from_r <- data.frame(Name = "Peter", Height = 61)

  # Macro function
  update_students <- SASunittest::r_function_from_sas_macro("update_students")

  # Prepare SAS code
  macro_code <- update_students(new_students = "rin.data_from_r",
                                min_height = 0)

  # ACT -----------------------------------------------------------------------

  # Run SAS code
  out <- NNremote::SAS(main_code  = macro_code,
                       data_input = list("data_from_r" = data_from_r))

  # EXPECT --------------------------------------------------------------------

  # Extract new_class
  test_out <- out[["tables"]][["new_class"]]

  # Check number of rows
  testthat::expect_equal(nrow(test_out), 20)

  # Check length of Name
  testthat::expect_equal(attr(test_out[["Name"]], "length"), "8")

  # Check warning message
  SASunittest::expect_sasWarning(out, "min_height should be positive")
})
```

Demo

Code to test

```
%macro update_students(new_students, min_height);

    /* Check that min_height is positive */
    %if &min_height. <= 0 %then %do;
        %put WARNING: min_height should be positive;
    %end;

    /* Combine tables and add session_tag variable */
    data new_class;
        set sashelp.class
            &new_students.;
        where height >= &min_height.;
    run;

%mend update_students;
```

Unit test

```
test_that("update_students - basic test", {
  # SETUP ---------------------------------------------------------------------

  # Test data
  data_from_r <- data.frame(Name = "Peter", Height = 61)

  # Macro function
  update_students <- SASunittest::r_function_from_sas_macro("update_students")

  # Prepare SAS code
  macro_code <- update_students(new_students = "rin.data_from_r",
                                min_height = 0)

  # ACT -----------------------------------------------------------------------

  # Run SAS code
  out <- NNremote::SAS(main_code  = macro_code,
                       data_input = list("data_from_r" = data_from_r))

  # EXPECT --------------------------------------------------------------------

  # Extract new_class
  test_out <- out[["tables"]][["new_class"]]

  # Check number of rows
  testthat::expect_equal(nrow(test_out), 20)

  # Check length of Name
  testthat::expect_equal(attr(test_out[["Name"]], "length"), "8")

  # Check warning message
  SASunittest::expect_sasWarning(out, "min_height should be positive")
})
```

Demo

Unit test

```r
test_that("update_students - basic test", {
  # SETUP ---------------------------------------------------------------------

  # Test data
  data_from_r <- data.frame(Name = "Peter", Height = 61)

  # Macro function
  update_students <- SASunittest::r_function_from_sas_macro("update_students")

  # Prepare SAS code
  macro_code <- update_students(new_students = "rin.data_from_r",
                                min_height = 0)

  # ACT -----------------------------------------------------------------------

  # Run SAS code
  out <- NNremote::SAS(main_code  = macro_code,
                       data_input = list("data_from_r" = data_from_r))

  # EXPECT --------------------------------------------------------------------

  # Extract new_class
  test_out <- out[["tables"]][["new_class"]]

  # Check number of rows
  testthat::expect_equal(nrow(test_out), 20)

  # Check length of Name
  testthat::expect_equal(attr(test_out[["Name"]], "length"), "8")

  # Check warning message
  SASunittest::expect_sasWarning(out, "min_height should be positive")
})
```

Code to test

```
%macro update_students(new_students, min_height);

    /* Check that min_height is positive */
    %if &min_height. <= 0 %then %do;
        %put WARNING: min_height should be positive;
    %end;

    /* Combine tables and add session_tag variable */
    data new_class;
        set sashelp.class
            &new_students.;
        where height >= &min_height.;
    run;

%mend update_students;
```

Demo

Code to test

```
%macro update_students(new_students, min_height);

    /* Check that min_height is positive */
    %if &min_height. <= 0 %then %do;
        %put WARNING: min_height should be positive;
    %end;

    /* Combine tables and add session_tag variable */
    data new_class;
        set sashelp.class
            &new_students.;
        where height >= &min_height.;
    run;

%mend update_students;
```

Unit test

```
test_that("update_students - basic test", {
  # SETUP ---------------------------------------------------------------------

  # Test data
  data_from_r <- data.frame(Name = "Peter", Height = 61)

  # Macro function
  update_students <- SASunittest::r_function_from_sas_macro("update_students")

  # Prepare SAS code
  macro_code <- update_students(new_students = "rin.data_from_r",
                                min_height = 0)

  # ACT -----------------------------------------------------------------------

  # Run SAS code
  out <- NNremote::SAS(main_code  = macro_code,
                       data_input = list("data_from_r" = data_from_r))

  # EXPECT --------------------------------------------------------------------

  # Extract new_class
  test_out <- out[["tables"]][["new_class"]]

  # Check number of rows
  testthat::expect_equal(nrow(test_out), 20)

  # Check length of Name
  testthat::expect_equal(attr(test_out[["Name"]], "length"), "8")

  # Check warning message
  SASunittest::expect_sasWarning(out, "min_height should be positive")
})
```

Demo

Code to test

```
%macro update_students(new_students, min_height);

    /* Check that min_height is positive */
    %if &min_height. <= 0 %then %do;
        %put WARNING: min_height should be positive;
    %end;

    /* Combine tables and add session_tag variable */
    data new_class;
        set sashelp.class
            &new_students.;
        where height >= &min_height.;
    run;

%mend update_students;
```

Unit test

```
test_that("update_students - basic test", {
  # SETUP ---------------------------------------------------------------------

  # Test data
  data_from_r <- data.frame(Name = "Peter", Height = 61)

  # Macro function
  update_students <- SASunittest::r_function_from_sas_macro("update_students")

  # Prepare SAS code
  macro_code <- update_students(new_students = "rin.data_from_r",
                                min_height = 0)

  # ACT -----------------------------------------------------------------------

  # Run SAS code
  out <- NNremote::SAS(main_code  = macro_code,
                       data_input = list("data_from_r" = data_from_r))

  # EXPECT --------------------------------------------------------------------

  # Extract new_class
  test_out <- out[["tables"]][["new_class"]]

  # Check number of rows
  testthat::expect_equal(nrow(test_out), 20)

  # Check length of Name
  testthat::expect_equal(attr(test_out[["Name"]], "length"), "8")

  # Check warning message
  SASunittest::expect_sasWarning(out, "min_height should be positive")
})
```

Demo

Code to test

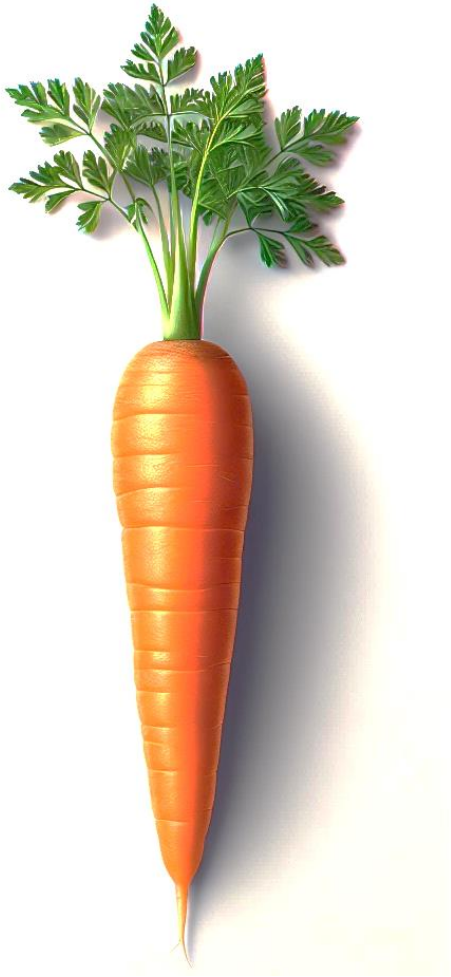
```
%macro update_students(new_students, min_height);

    /* Check that min_height is positive */
    %if &min_height. <= 0 %then %do;
        %put WARNING: min_height should be positive;
    %end;

    /* Combine tables and add session_tag variable */
    data new_class;
        set sashelp.class
            &new_students.;
        where height >= &min_height.;
    run;

%mend update_students;
```

Unit test

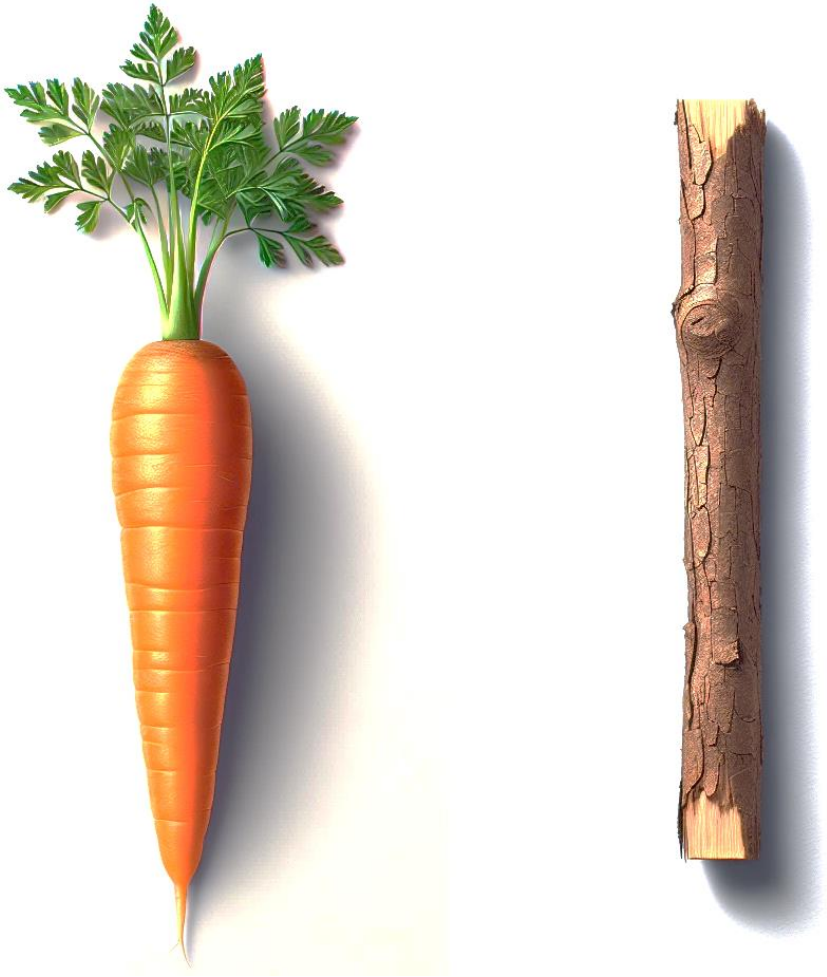
```
test_that("update_students - basic test", {
  # SETUP ---------------------------------------------------------------------

  # Test data
  data_from_r <- data.frame(Name = "Peter", Height = 61)

  # Macro function
  update_students <- SASunittest::r_function_from_sas_macro("update_students")

  # Prepare SAS code
  macro_code <- update_students(new_students = "rin.data_from_r",
                                min_height = 0)

  # ACT -----------------------------------------------------------------------

  # Run SAS code
  out <- NNremote::SAS(main_code  = macro_code,
                       data_input = list("data_from_r" = data_from_r))

  # EXPECT --------------------------------------------------------------------

  # Extract new_class
  test_out <- out[["tables"]][["new_class"]]

  # Check number of rows
  testthat::expect_equal(nrow(test_out), 20)

  # Check length of Name
  testthat::expect_equal(attr(test_out[["Name"]], "length"), "8")

  # Check warning message
  SASunittest::expect_sasWarning(out, "min_height should be positive")
})
```

People

People



People



People

People

Perspectives

Perspectives

- Standard programs = good!

Perspectives

- Standard programs = good!
- Change management from the start

Perspectives

- Standard programs = good!
- Change management from the start
- 

Perspectives

- Standard programs = good!
- Change management from the start
- 

- Foundation for future?

Matthew Phelps
mewp@novonordisk.com

Michael Hedegaard Thomsen
mhzt@novonordisk.com

Nicolai Skov Johnsen
nosj@novonordisk.com

The Global Healthcare Data Science Community