Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

DKRZ
DEUTSCHES
KLIMARECHENZENTRUM

FORTRANTESTGENERATOR: AUTOMATIC AND FLEXIBLE UNIT TEST GENERATION FOR LEGACY HPC CODE

CHRISTIAN HOVY
UNIVERSITÄT HAMBURG
christian.hovy@uni-hamburg.de

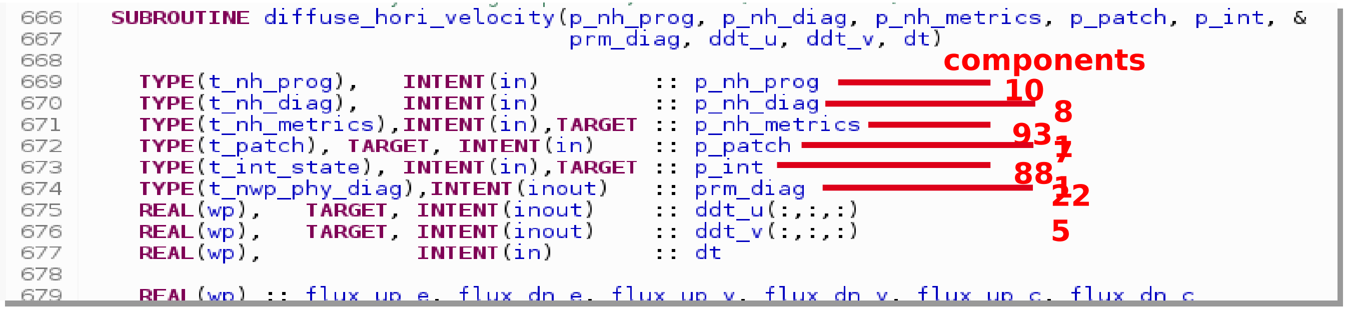
JULIAN KUNKEL
GERMAN CLIMATE COMPUTING CENTER
kunkel@dkrz.de

ISC HPC 2017, Poster Presentation
Frankurt, Germany

2017-06-20

Obstacles for Testing

- Limited knowledge of expected results
- Effort of setting up of test data for unit testing
  - Large and complicated data layouts (i.e., unstructured grids, cache blocking etc.)
- Long run times
- Parallelism
- Legacy Code



```fortran
  SUBROUTINE diffuse_hori_velocity(p_nh_prog, p_nh_diag, p_nh_metrics, p_patch, p_int, &
                                   prm_diag, ddt_u, ddt_v, dt)

    TYPE(t_nh_prog),    INTENT(in)          :: p_nh_prog
    TYPE(t_nh_diag),    INTENT(in)          :: p_nh_diag
    TYPE(t_nh_metrics),INTENT(in),TARGET    :: p_nh_metrics
    TYPE(t_patch), TARGET, INTENT(in)       :: p_patch
    TYPE(t_int_state), INTENT(in),TARGET    :: p_int
    TYPE(t_nwp_phy_diag),INTENT(inout)      :: prm_diag
    REAL(wp),    TARGET, INTENT(inout)      :: ddt_u(:,:,:)
    REAL(wp),    TARGET, INTENT(inout)      :: ddt_v(:,:,:)
    REAL(wp),            INTENT(in)         :: dt

    REAL(wp) :: flux_up_e, flux_dn_e, flux_up_v, flux_dn_v, flux_up_c, flux_dn_c
```

Example snippet from the earth system model ICON
[https://www.mpimet.mpg.de/en/science/models/icon]

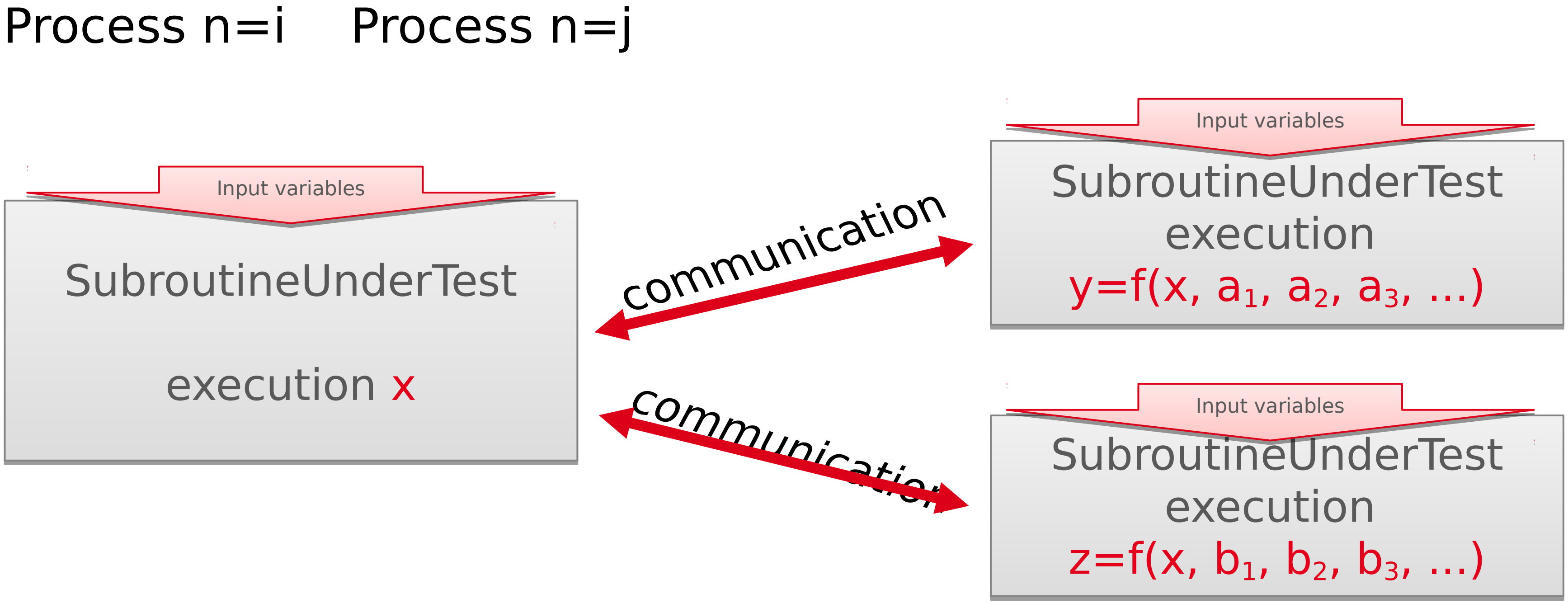
FortranTestGenerator (FTG)

- Automatic unit test generation for subroutines of Fortran applications
- Strategy: Capture & Replay
  - Extract a set of input data by running the original application
  - Run subroutine independently with one process feeding captured data
- Features
  - Instrumentation for capture code
    FTG generates code for serializing and storing a subroutines input data
    - Developer can define an event upon which the input data is captured
    - FTG captures only necessary variables using static code analysis
  - Test driver creation: loads this data and runs the subroutine (replay code)
    - Checks and modification of data can be done by the developer
  - FTG uses templates to generate instrumentation and test driver
    - Customizable

Capture & Replay with MPI

- Results of subroutines may depend on MPI communication
- Enables unit testing with only a single process
- Capture/Replay MPI communication (Ongoing effort)
  - Use either HDF5 to store data or create code to initialize data
  - Replay using a dummy MPI implementation
    - Feeds in recorded responses from MPI
    - In the order it has been recorded
  - Python editor to add / modify input

Capture & Replay with MPI


Process n=i
Process n=j
Input variables
SubroutineUnderTest
execution x
communication
communication
Input variables
SubroutineUnderTest
execution
$y=f(x, a_1, a_2, a_3, \ldots)$
Input variables
SubroutineUnderTest
execution
$z=f(x, b_1, b_2, b_3, \ldots)$