4

ESM-Tools Terminology

Overview
YAML Hierarchy
Configuration files
Runscripts
YAML Sections
Feature Variables
Compilation Scripts
.run files

4 Terminology

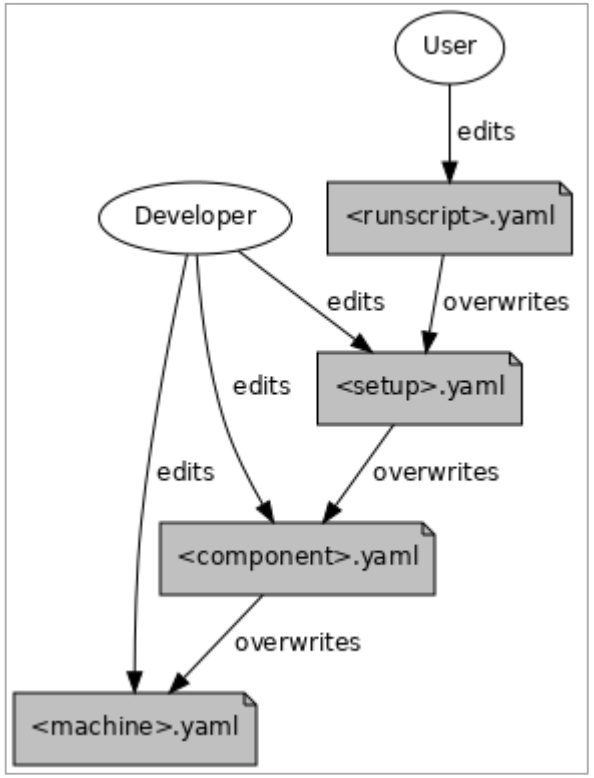
- Through out this workshop we will be using ESM-Tools-specific terms that you'll need to be familiar with. Those terms are defined in this power point.

- You'll see those terms in the other slides coloured in orange

4 YAML Hierarchy

- YAML files are always inherited from more general to more specific. Last one wins (eg. user runscript).

File Name	Order of Execution	Precedence	Location	Defined by
machine.yaml	First	Lowest	esm_tools/configs/machines	ESM-Tools
setup.yaml			esm_tools/configs/setups (Note: when using a coupled setup)	ESM-Tools
model.yaml			esm_tools/configs/components	ESM-Tools
runscript.yaml	Last	Highest	user defined directory	User



4 Terminology – configuration files

The yaml files that contain the **default configurations** for HPCs, models, coupled systems, job schedulers (SLURM, PBS), default ESM-Tools recipes, …

esm_tools/configs/

components	Stand-alone model, couplers, I/O libraries configurations
setups	Coupled system default configurations
machines	HPC default configurations
coupling	Source code branch information for coupled system stored here, only for use in ESM-Master (to be removed in the future)
default	ESM-Tools default configurations
esm_software	Recipes and defaults for ESM-Runscripts and ESM-Master
other_software	Job schedulers and other external software configurations

4 Terminology – runscripts

- User interface for running experiments
- Should include all the deviations from the defaults defined in the **configuration files**
- Can be shared to reproduce the same experiment
- A yaml file with **sections**

<your_fesom_runscript>.yaml

```yaml
general:
        account: <your_account>
        setup_name: fesom
        compute_time: "00:20:00"
        initial_date: '2001-01-01'
        final_date: '2001-03-01'
        base_dir: <your_basedir>
        nyear: 0
        nmonth: 1
        nday: 0
        use_venv: False
fesom:
        version: 2.1
        model_dir: <your_model_dir>
        lresume: 0
        restart_rate: 1
        restart_unit: 'm'
        post_processing: 0
```

4 Terminology – YAML sections

- 1st level keys on a yaml file
  - general
  - <model_name>/<component_name>
  - computer

- Only the **runscripts** and the **setups** files have **sections**

runscripts
configs
components
setups
machines
coupling

default
esm_software
other_software

<your_fesom_runscript>.yaml

```yaml
general:
        account: <your_account>
        setup_name: fesom
        compute_time: "00:20:00"
        initial_date: '2001-01-01'
        final_date: '2001-03-01'
        base_dir: <your_basedir>
        nyear: 0
        nmonth: 1
        nday: 0
        use_venv: False
fesom:
        version: 2.1
        model_dir: <your_model_dir>
        lresume: 0
        restart_rate: 1
        restart_unit: 'm'
        post_processing: 0
```

4 Terminology – feature variables

Some variables in the yaml files trigger functionalities in ESM-Tools. Through this variables the **yaml syntax is extended**.

We refer to this variables as **feature variables**

```yaml
# time variables
general:
    nday: 0
    nmonth: 1
    initial_date: "1850-01-01T00:00:00"
    final_date: "1860-01-01T00:00:00"
```

```yaml
# creating and accessing variables from different sections
ini_restart_dir: "${general.ini_restart_dir}/fesom/"
```

```yaml
# Changing Fortran namelists
namelist_changes:
    namelist.echam:
        runctl:
            out_expname: ${general.expid}
            dt_start:
                - ${pseudo_start_date!year}
                - ${pseudo_start_date!month}
```

```yaml
# adding and removing elements from lists and dicts

list1:
    - element1
    - element2

add_list1:
    - element3
    - element4
```

```yaml
# choose_ blocks allow select-case (aka switch) statements
resolution: CORE2

choose_resolution:
    CORE2:
        nx: 126858
        mesh_dir: "${pool_dir}/meshes/mesh_CORE2_final/"
        nproc: 288
        time_step: 450
    GLOB:
        nx: 830305
```

4 Terminology – compilation scripts

- For each component that **esm_master** builds, it produces a compilation script **comp-*.sh** that includes the environment specified in the configuration files (machine + components + setups) files involved
- Written in the same directory where you execute **esm_master**
- Copied to the compilation folder after the building finishes

```
#!/bin/bash -l
# Dummy script generated by esm-tools, to be removed later:
set -e
module purge
module unload netcdf_c
module unload intel intelmpi
module load python3/2021.01-gcc-9.1.0
module load cmake/3.13.3
module load autoconf/2.69
module load nco
module load cdo
module load gcc/4.8.2
module unload intel intelmpi
module load intel/18.0.4 intelmpi/2018.5.288
module load libtool/2.4.6
module load automake/1.14.1
module unload gcc
module load gcc/4.8.2
export LC_ALL=en_US.UTF-8
export FC=mpiifort
export F77=mpiifort
export MPIFC=mpiifort
export CC=mpiicc
export CXX=mpiicpc
export MPIROOT="$(mpiifort -show | perl -lne 'm{ -I(.*?)/include } and print $1')"
...
cd fesom-2.1
mkdir -p build; cd build; cmake -DFESOM_COUPLED=ON ..; make install -j `nproc --all`
cd ..
```

4 Terminology –*.run files

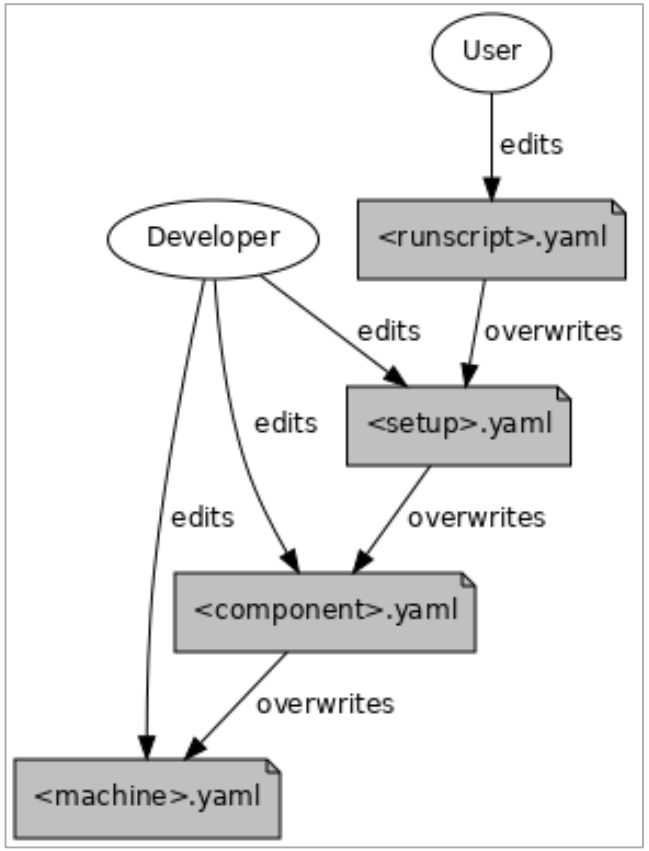
- esm_runscripts produces a ***.run** script with SBATCH headers that is then submitted to SBATCH. This script contains the combined environments specified in the configuration files (machine + components + setups) files involved

- This script can be found in <experiment_dir>/run_DATE/scripts for runs that have not being submitted or are still running, or in the <experiment_dir/scripts> directory for runs that have already run

```
#SBATCH --partition=compute
#SBATCH --time=01:45:00
#SBATCH --ntasks=896
...
module purge
module unload netcdf_c
module unload intel intelmpi
...
export LC_ALL=en_US.UTF-8
export FC=mpif90
export F77=mpif90
...

# Set stack size to unlimited
ulimit -s unlimited

# 3...2...1...Liftoff!
echo $(date +"%a %b  %e %T %Y") : compute 1 1850-01-01T00:00:00 1233 - start
>> /work/ab0995/a270152//workshop_test/log//workshop_test_awiesm.log

cd /work/ab0995/a270152//workshop_test/run_18500101-18501231/work/

time srun -l --kill-on-bad-exit=1 --cpu_bind=cores --multi-prog hostfile_srun
2>&1 &

...
```

4 Terminology – finished_config file

- Internally, esm_parser puts together all the information from the different yaml files for the given experiment into a Python object called config
- This object, containing all the information about the experiment, is passed to the different ESM-Tools functions
- esm_runscripts dumps this object into a yaml file in <experiment_dir>/run_DATE/configs/*finished_config.yaml for runs that have not being submitted or are still running, or in <experiment_dir>/configs/*finished_config.yaml for runs that have already run
- The **finished_config file** is used for checking that the final configuration works as expected



Terminology – check mode

- **esm_master** and **esm_runscripts** can run in check mode by adding the --check or -c flags to the command

- esm_master in check mode
  - Outputs the git commands and building commands but does not produce the **comp-*.sh** files

To be changed in the future to it procudes **comp-* .sh** files

- **esm_runscripts** in check mode
  - Bakes the yaml information and produces the **finished_config.yaml**
  - Prepares the experiment folder (copies in input, forcing, binaries, …)
  - Produces the ***.run** file
  - **Does not** submit the ***.run** script to sbatch

Terminology – models and components

FESOM-2

- AWI's ocean model
- Finite Volume
- Unestructure mesh

AWI-ESM-2.1

- ECHAM6 + FESOM-2.1 with OASIS3MCT
- Dynamic vegetation
- + REcoM (biogeochemistry) + PISM (WIP, offline coupled)

VILMA-PISM

- Offline coupling through ESM-Tools workflow manager