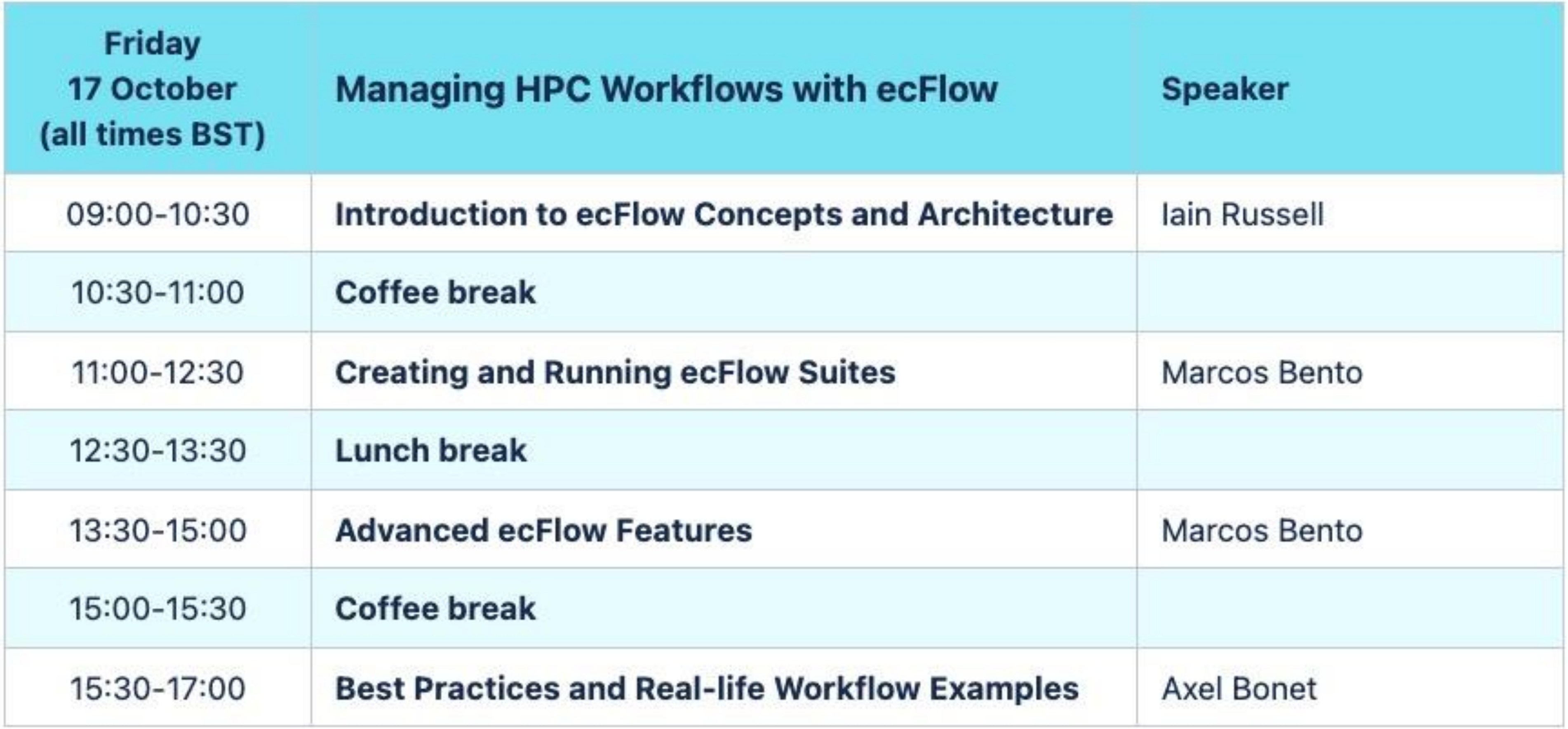
ecFlow

Introduction

Marcos Bento, Axel Bonet, Iain Russell

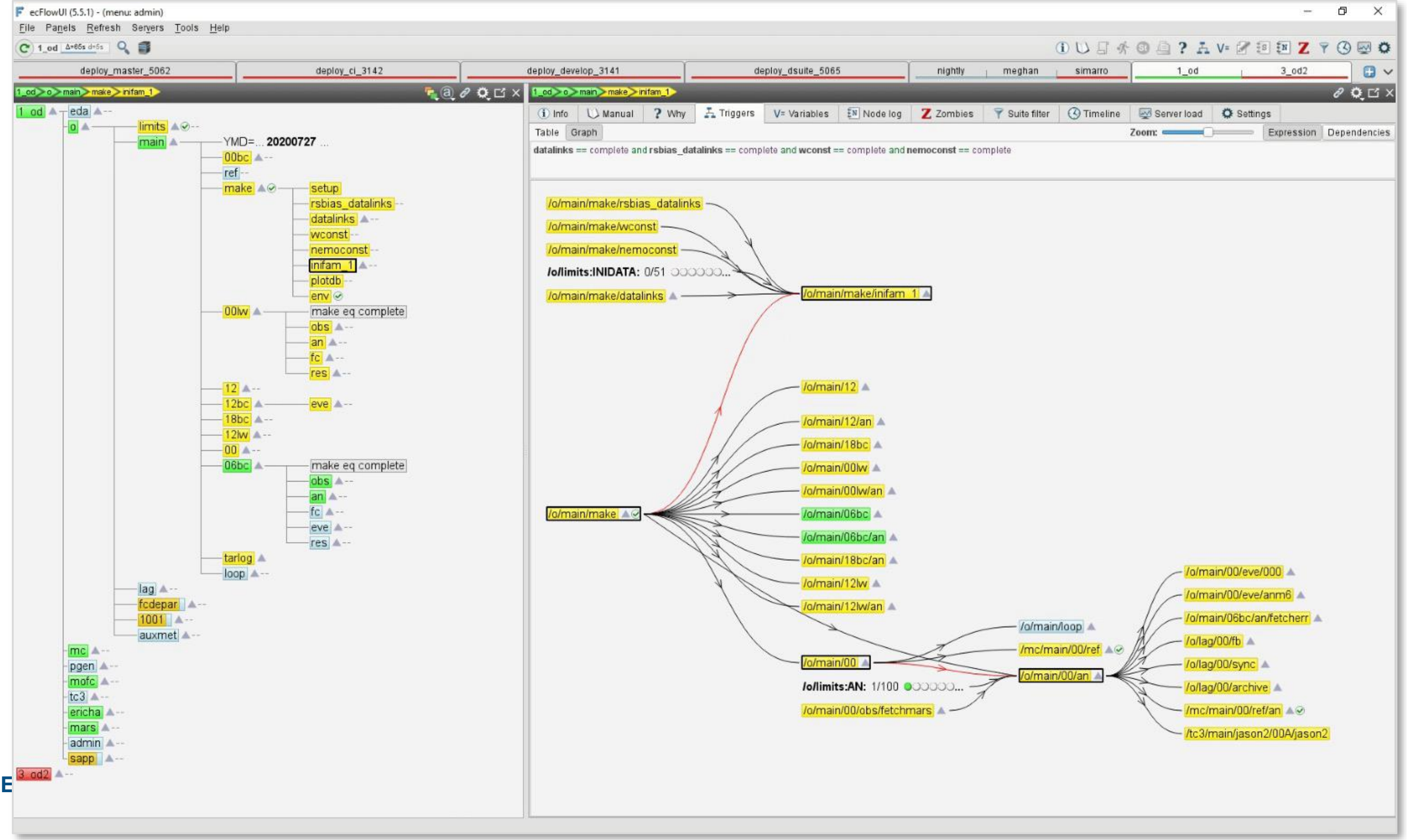
Schedule



Friday 17 October (all times BST)	Managing HPC Workflows with ecFlow	Speaker
09:00-10:30	**Introduction to ecFlow Concepts and Architecture**	Iain Russell
10:30-11:00	**Coffee break**	
11:00-12:30	**Creating and Running ecFlow Suites**	Marcos Bento
12:30-13:30	**Lunch break**	
13:30-15:00	**Advanced ecFlow Features**	Marcos Bento
15:00-15:30	**Coffee break**	
15:30-17:00	**Best Practices and Real-life Workflow Examples**	Axel Bonet

Topics in this section

- Main ecFlow principles, concepts and terminology
- Hands-on with a simple example
- Interactive inspection with ecFlowUI

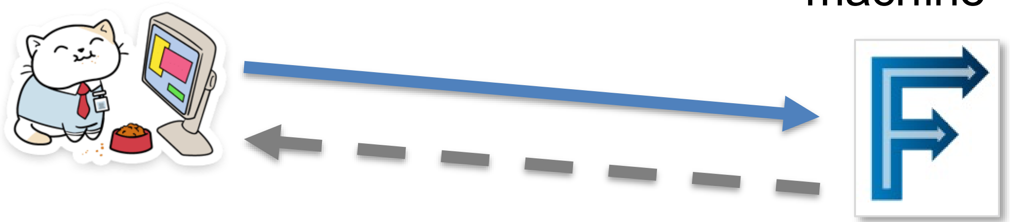


datalinks == complete and rsbias_datalinks == complete and wconst == complete and nemoconst == complete

Overview of ecFlow

- A workflow manager developed at ECMWF
  - Used in operations for many years, also in other organisations
- Client/Server based Scheduler, Monitor, Supervisor
- Designed to schedule a large number of computer processes in a heterogeneous environment
  - Jobs can be run on local machine or submitted to remote machines
- Flexible triggering of tasks – e.g. from clock time, or from completion of other tasks
- Interaction through:
  - command-line client,
  - Python API
  - graphical user interface (ecFlowUI)

Schematic of ecFlow usage – simplest case

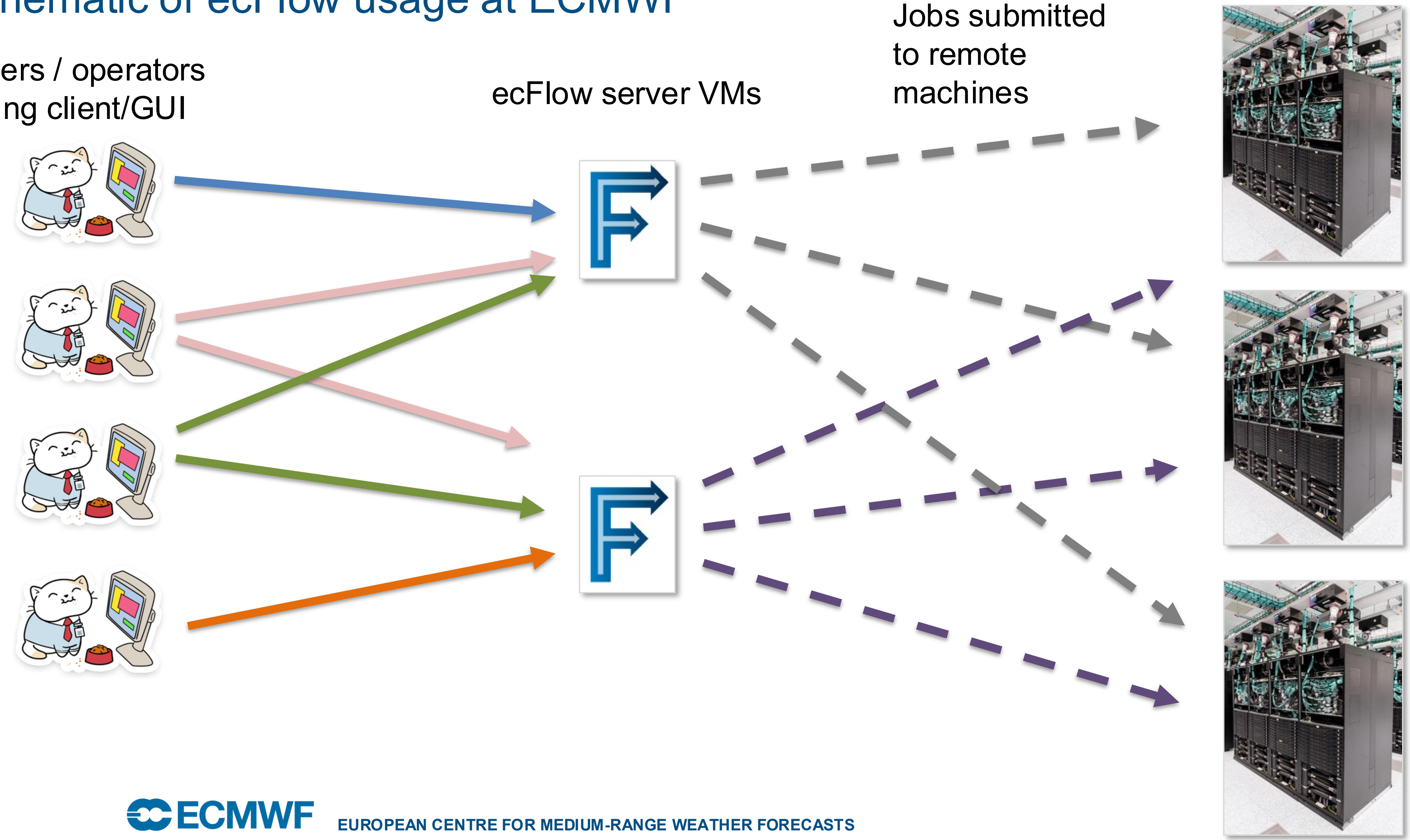


Users / operators using client/GUI

ecFlow server running on same machine

Jobs run locally

Schematic of ecFlow usage at ECMWF



Users / operators using client/GUI

ecFlow server VMs

Jobs submitted to remote machines

HPC clusters

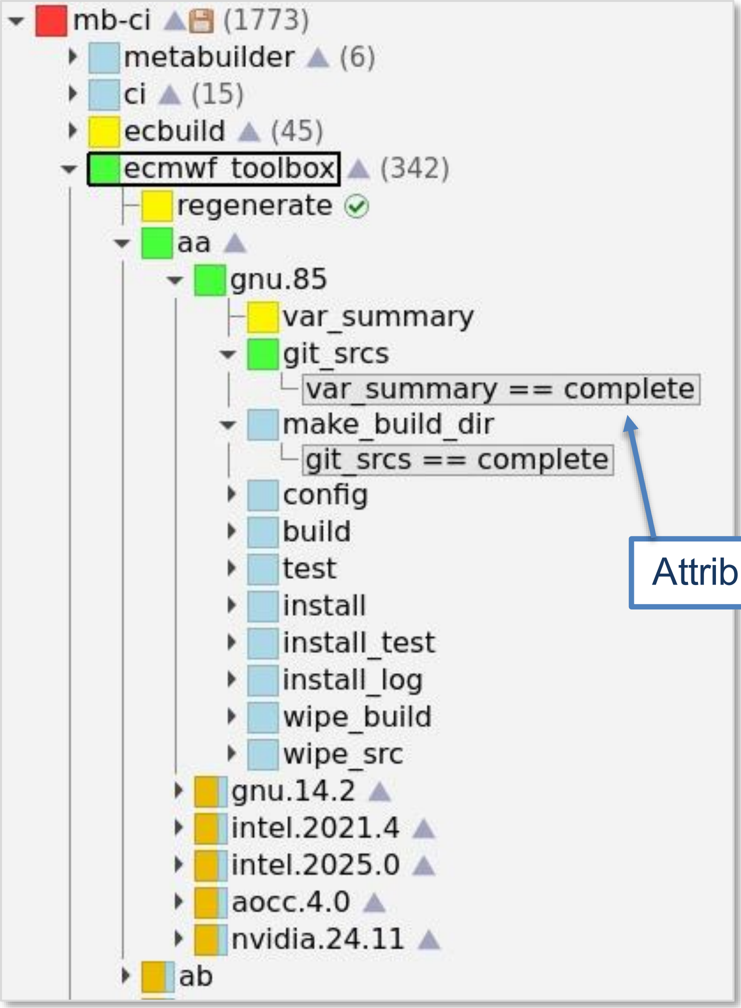
Anatomy of an ecFlow workflow

ecFlow server

Suites
(contain families and tasks)

Families
(Contain families and tasks)

Tasks – leaf nodes
(Responsible for running actual jobs)

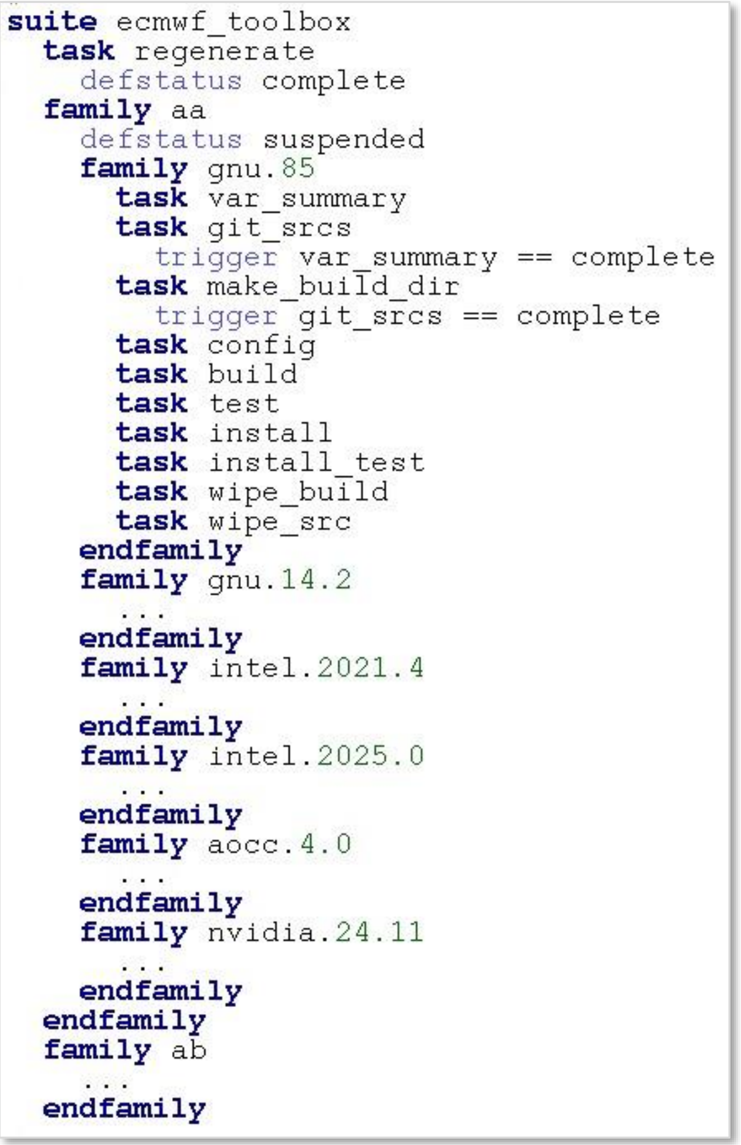


```
mb-ci ▲ (1773)
  metabuilder ▲ (6)
  ci ▲ (15)
  ecbuild ▲ (45)
  ecmwf_toolbox ▲ (342)
    regenerate ✓
    aa ▲
      gnu.85
        var_summary
        git_srcs
          var_summary == complete
        make_build_dir
          git_srcs == complete
        config
        build
        test
        install
        install_test
        install_log
        wipe_build
        wipe_src
      gnu.14.2 ▲
      intel.2021.4 ▲
      intel.2025.0 ▲
      aocc.4.0 ▲
      nvidia.24.11 ▲
    ab
```

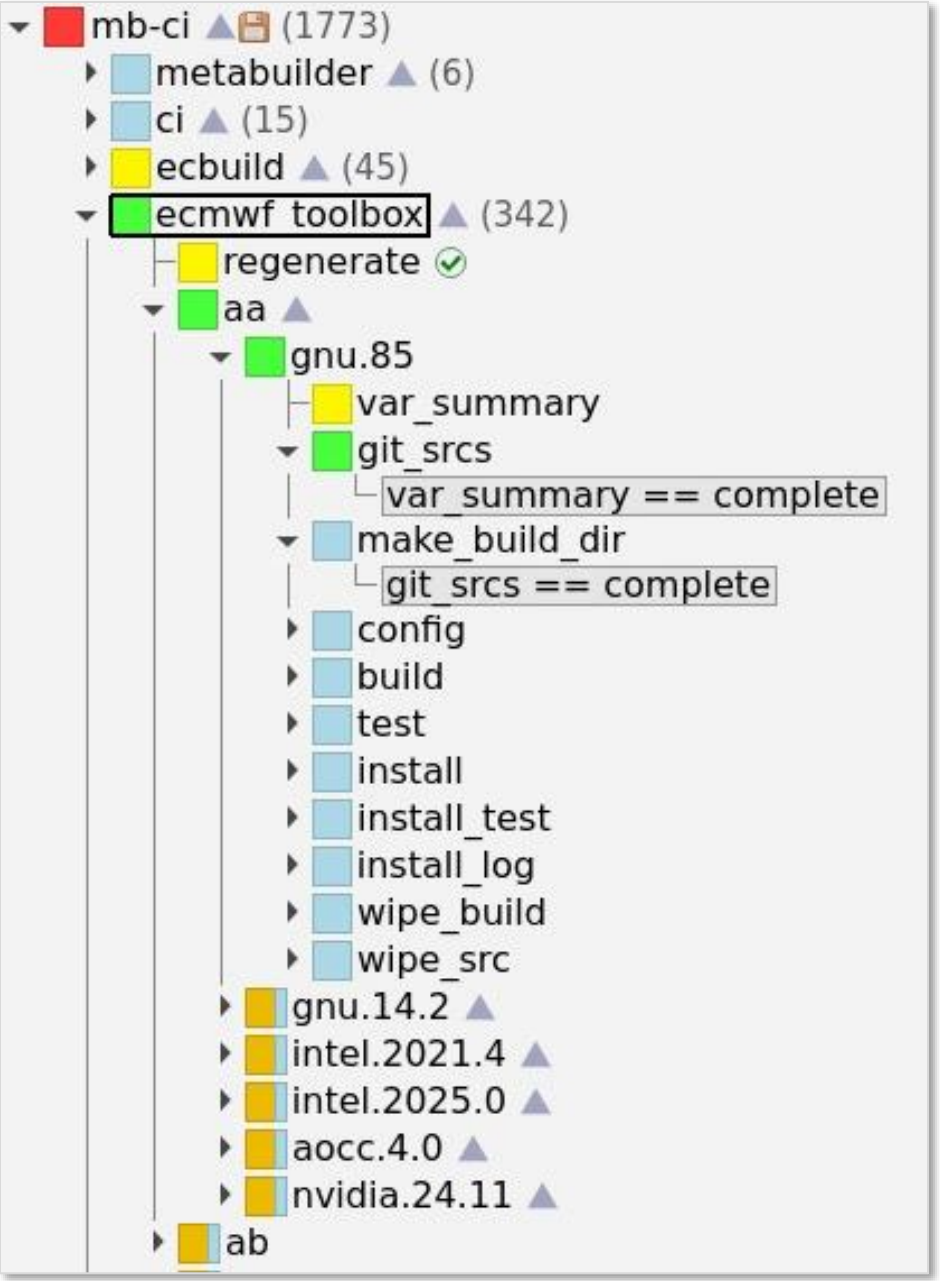
Attributes

The whole tree can be defined in a 'definition file'

Cut-down version of the definition file to define the ecmwf_toolbox suite



```
suite ecmwf_toolbox
  task regenerate
    defstatus complete
  family aa
    defstatus suspended
    family gnu.85
      task var_summary
      task git_srcs
        trigger var_summary == complete
      task make_build_dir
        trigger git_srcs == complete
      task config
      task build
      task test
      task install
      task install_test
      task wipe_build
      task wipe_src
    endfamily
    family gnu.14.2
      ...
    endfamily
    family intel.2021.4
      ...
    endfamily
    family intel.2025.0
      ...
    endfamily
    family aocc.4.0
      ...
    endfamily
    family nvidia.24.11
      ...
    endfamily
  endfamily
  family ab
    ...
  endfamily
```

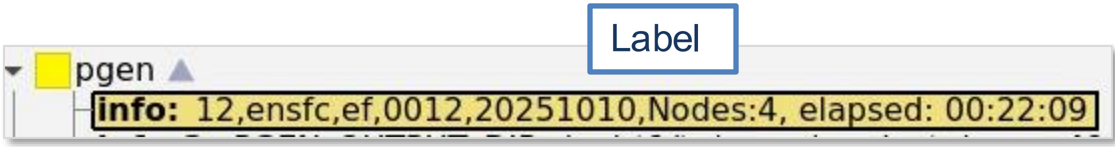


```
▾ 🟥 mb-ci ▲ (1773)
   ▸ metabuilder ▲ (6)
   ▸ ci ▲ (15)
   ▸ ecbuild ▲ (45)
   ▾ ecmwf_toolbox ▲ (342)
      regenerate ✓
      ▾ aa ▲
         ▾ gnu.85
            var_summary
            ▾ git_srcs
               var_summary == complete
            ▾ make_build_dir
               git_srcs == complete
            ▸ config
            ▸ build
            ▸ test
            ▸ install
            ▸ install_test
            ▸ install_log
            ▸ wipe_build
            ▸ wipe_src
         ▸ gnu.14.2 ▲
         ▸ intel.2021.4 ▲
         ▸ intel.2025.0 ▲
         ▸ aocc.4.0 ▲
         ▸ nvidia.24.11 ▲
      ▸ ab
```

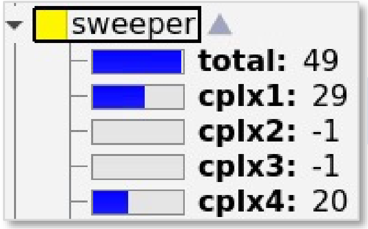
Node attributes

- ecFlow nodes (suites, families, tasks) can have *attributes*
- Examples include:
  - **Labels and meters**: can be changed dynamically by a task gives the Operator some feedback on what's happening
  - **Limits**: constrain the number of concurrent tasks within groups
  - **Variables**: can be injected into script templates
  - **Triggers, crons, repeats (and more)**: define the workflow graph by specifying dependencies, e.g.
    - "Run task B once family A is complete"
    - "Run task C at 1 o-clock every morning"
    - "Run task Z if task X fails"
  - **Important: the structure of the tree does not define the order of task execution – the attributes do!**

Label



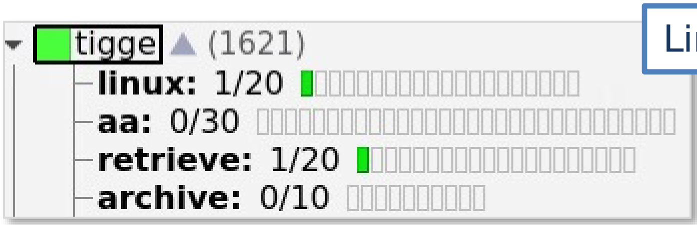
```
pgen
  info: 12,ensfc,ef,0012,20251010,Nodes:4, elapsed: 00:22:09
```



```
sweeper
  total: 49
  cplx1: 29
  cplx2: -1
  cplx3: -1
  cplx4: 20
```

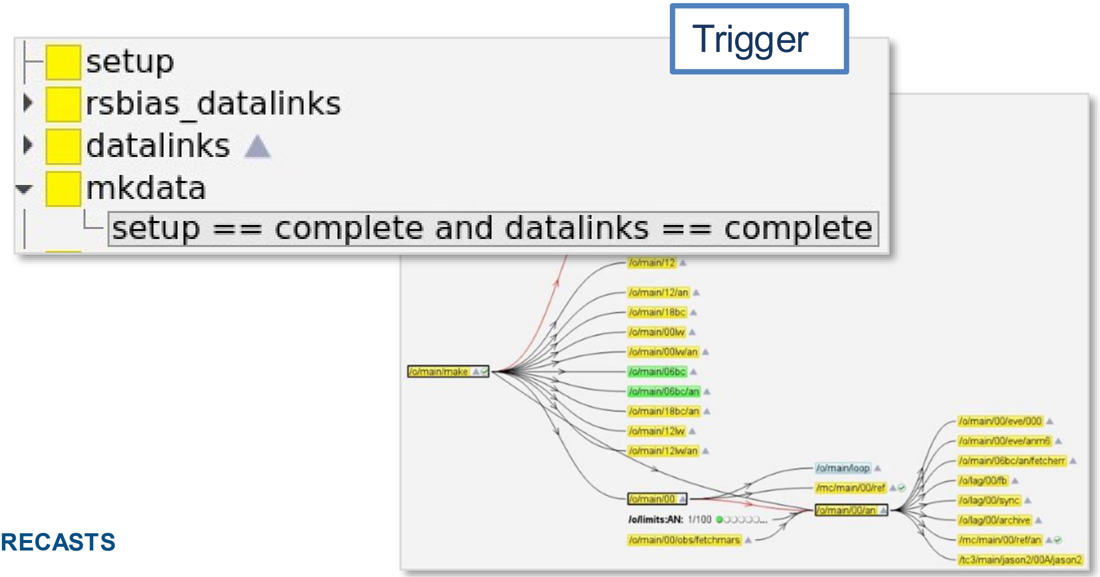
Meter

Limits



```
tigge (1621)
  linux: 1/20
  aa: 0/30
  retrieve: 1/20
  archive: 0/10
```

Trigger



```
setup
rsbias_datalinks
datalinks
mkdata
  setup == complete and datalinks == complete
```

ecFlow server – communication

- The default communication protocol is TCP/IP (HTTPS and UDP are also allowed, but are quite new, and beyond the scope of this training course)
- An ecFlow server can be run to communicate on any available port of choice – many ecFlow servers can thus run on the same machine, each using a different port
- The 'address' of an ecFlow server, then, is the combination of its host and port
- Command-line commands for server and client need to establish this address – either via environment variables
    - `ECF_HOST` and `ECF_PORT`
- Or through command-line flags
    - `--host=<hostname> --port=<portnumber>`
- The default is `--host=localhost --port=3141`

Starting an ecFlow server (outside ECMWF)

- Only works if ecFlow is installed, of course!
- Use the ecflow_server command with defaults:

  -
    ```
    ecflow_server &
    ```
- Or specify a port:

  -
    ```
    ecflow_server --port=3245 &
    ```

  -
    ```
    export ECF_PORT=3245
    ecflow_server &   # alternative to the above
    ```
- Check that it is working by using the client, e.g.

  -
    ```
    ecflow_client --ping
    ```

  -
    ```
    ecflow_client --ping --port=3245 --host=superduperhost
    ```

  -
    ```
    export ECF_PORT=3245
    export ECF_HOST=superduperhost
    ecflow_client --ping
    ```

ecFlow servers at ECMWF

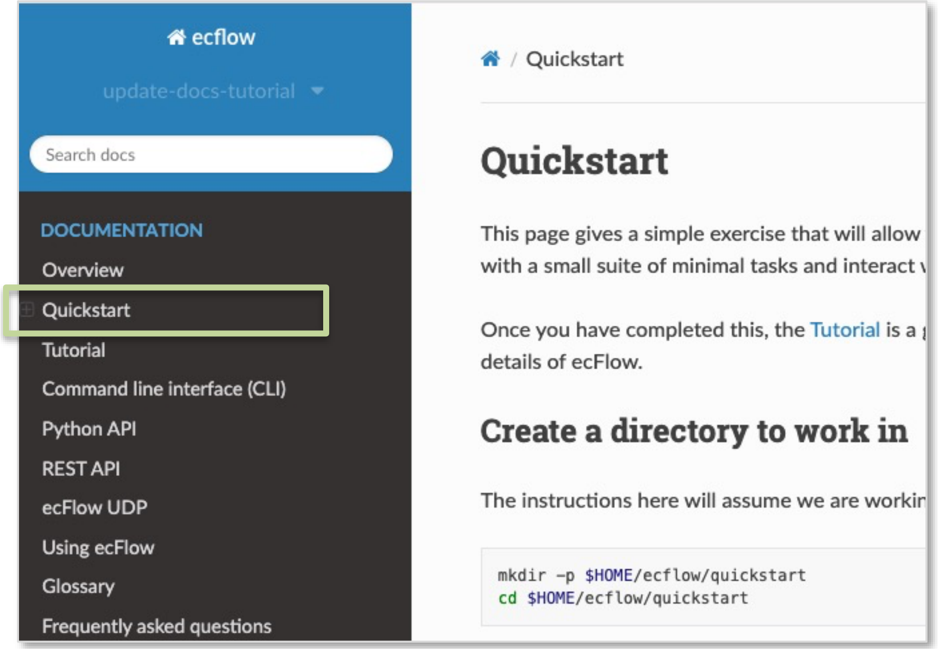
- https://confluence.ecmwf.int/display/UDOC/HPC2020%3A+Using+ecFlow
- At ECMWF, if you ask for one, you can have your own dedicated ecFlow server set up for you on a dedicated VM, separate from the HPC
- The host will typically be `ecfg-$USER-1`
  - Although some set up in the past will be `ecflow-gen-$USER-001`
- It has been ensured that everyone in this course has one set up and running!
- It is contactable from the VDI via ecflow_client and ecflow_ui
- Also from the Atos HPC through ecflow_client
- For this course, we will use the one that is supplied and already running for us!

Communicating with an ecFlow server via command-line client

- Only works if ecFlow is installed, of course!
- Remember that the `ecflow_client` commands need to know the host and port of the server you are trying to communicate with!
- Example things you can do with the `ecflow_client` command:
  - Load a suite definition to a server
  - Replace/add/remove selected parts of a suite
  - Query a server
  - Start/stop a suite
  - Manually execute tasks
  - Kill tasks
  - And more!
  - Tasks also communicate information back to the ecFlow server using ecflow_client
- All this works from the Python API too, but we will concentrate on the command-line to avoid confusion

Introductory exercise

- Go to https://ecflow.readthedocs.io/en/latest/
- Click on Quickstart
- Do what it says, and only that page :)
- When you come to the 'Start an ecFlow Server' section, choose the 'At ECMWF' tab



Quickstart

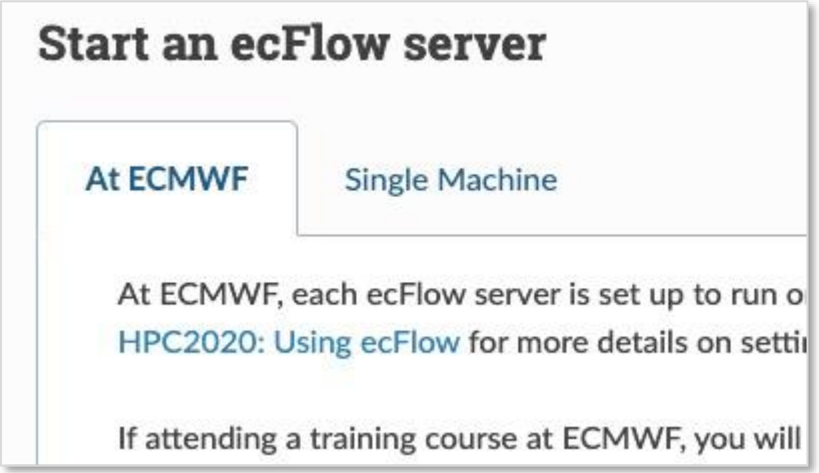
This page gives a simple exercise that will allow with a small suite of minimal tasks and interact v

Once you have completed this, the Tutorial is a details of ecFlow.

Create a directory to work in

The instructions here will assume we are workir

```
mkdir -p $HOME/ecflow/quickstart
cd $HOME/ecflow/quickstart
```



Start an ecFlow server

At ECMWF, each ecFlow server is set up to run o HPC2020: Using ecFlow for more details on setti

If attending a training course at ECMWF, you will