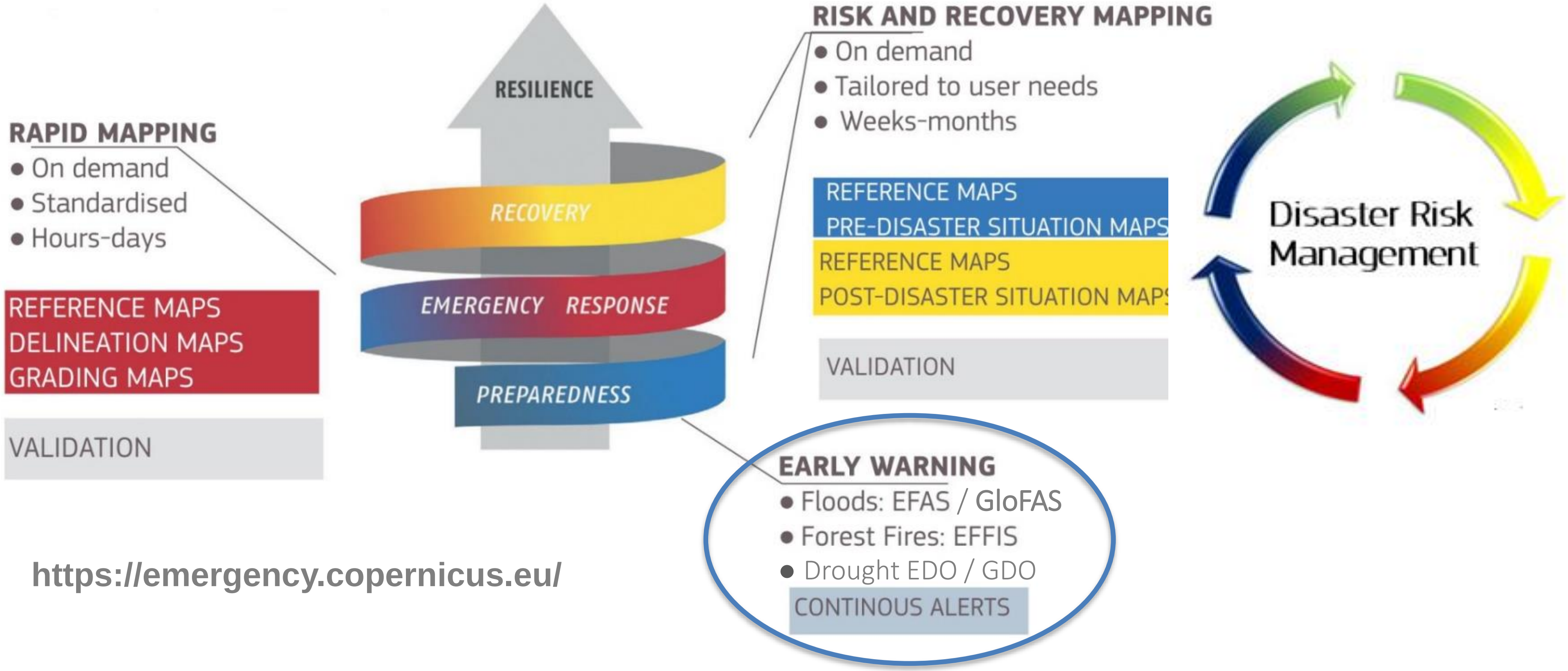
From loose scripts to ad-hoc reproducible workflows: a methodology using ECMWF's ecflow

Damien Decremer, Cinzia Mazzetti,
Christel Prudhomme, Corentin Carton De Wiart, Axel Bonet, …
Peter Salamon, Valerio Lorini, …

damien.decremer@ecmwf.int

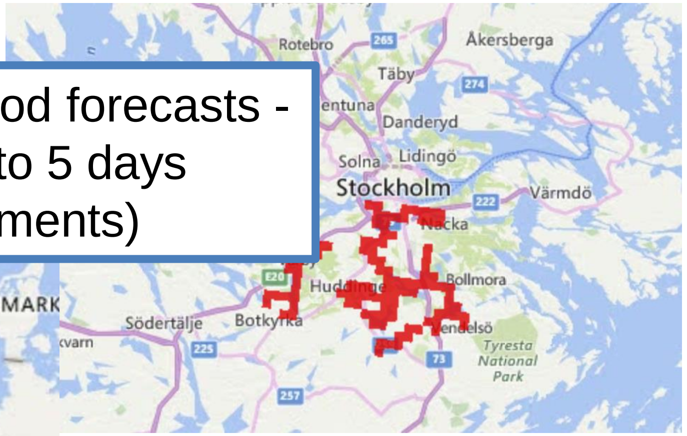
Copernicus Emergency Management Service (CEMS)

- **CEMS** set-up to “*Provides information for emergency response in relation to different types of disasters as well as prevention, preparedness, response and recovery activities*.”
- **EFAS**, European domain operational since in 2012, pre-operational since 2003
- **GloFAS**, Global domain operational from March 2018, pre-operational since 2011

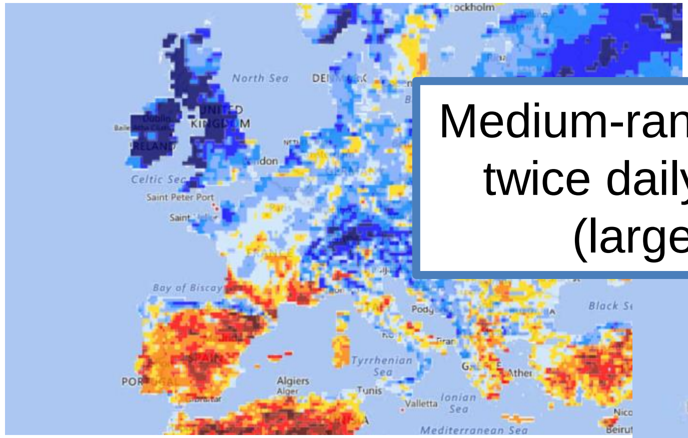


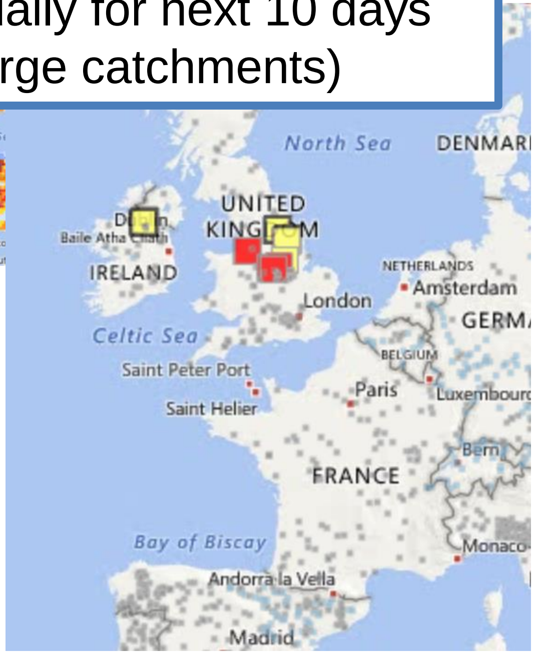
https://emergency.copernicus.eu/

EFAS Products



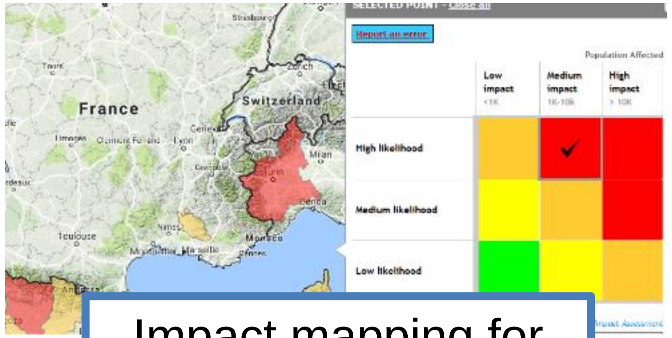
Ensemble Flash-flood forecasts - twice daily up to 5 days (small catchments)





Medium-range flood forecasts - twice daily for next 10 days (large catchments)

Radar-based flash-flood ‘nowcast’ every 15 minutes for next 3 hours



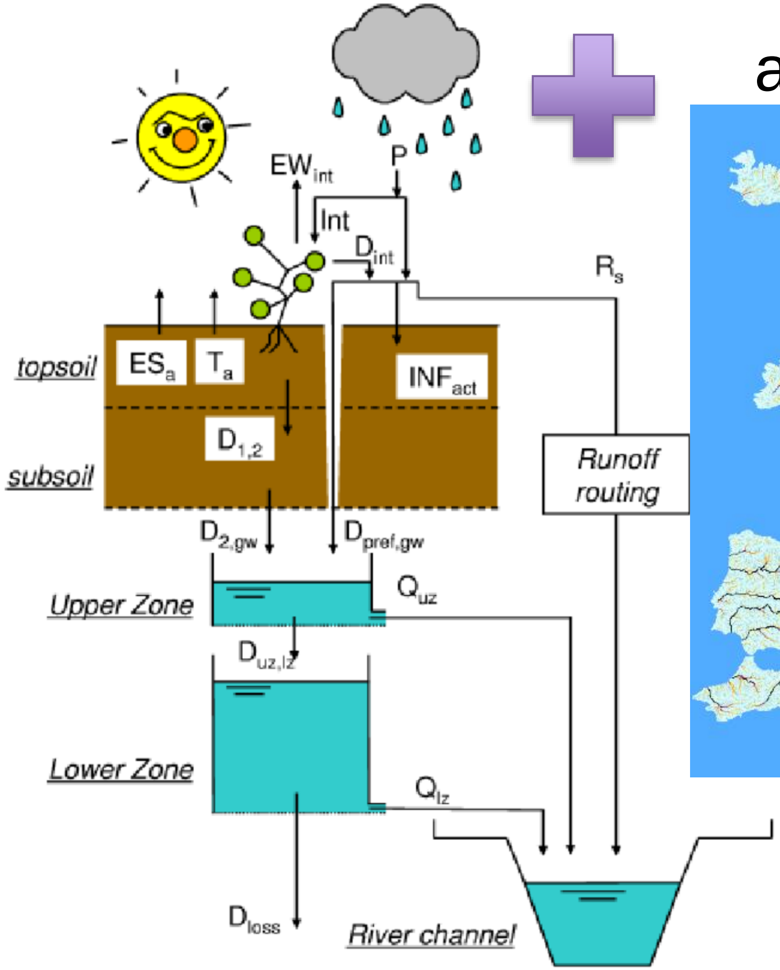
Impact mapping for rapid risk assessment

Seasonal hydrological anomalies outlooks once a month

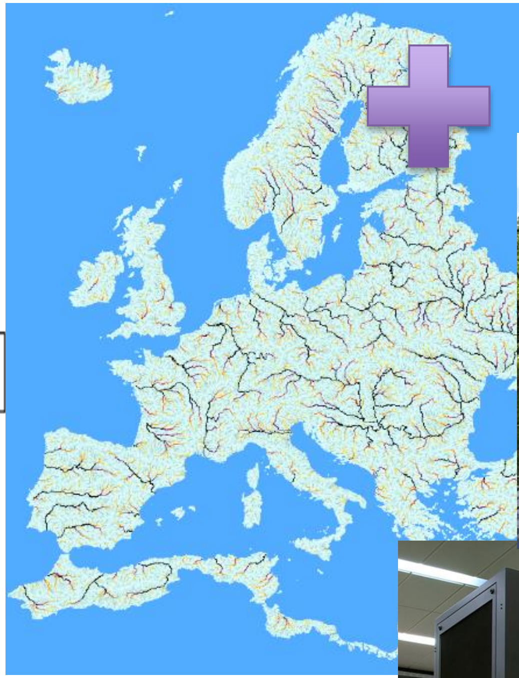
Fascinating story

Reproducible workflow

Imperfect hydrologic models



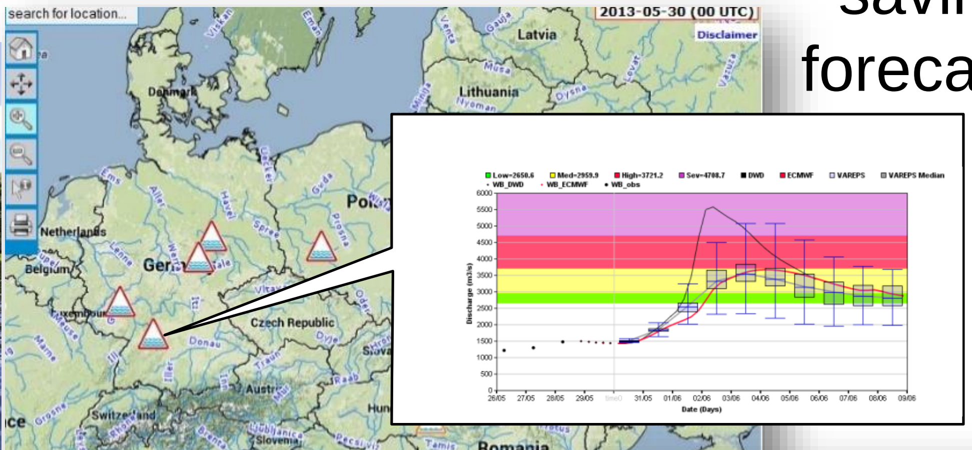
and river network



calibration

obs

Life-saving forecasts



on HPC

Passau Suffers Worst Flood in 500 Years

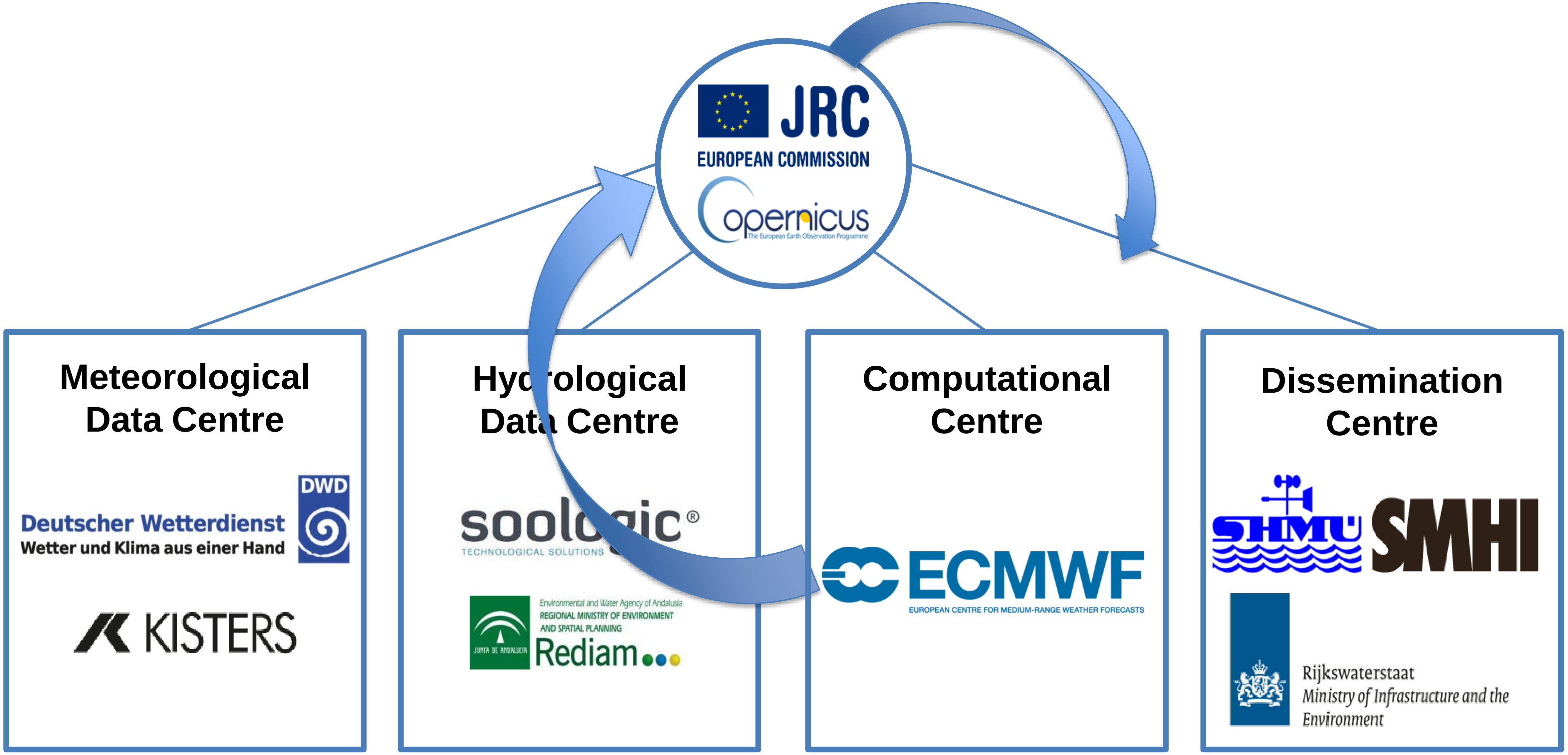
Amid the worst flooding seen in half a millennium, the streets of Passau have been transformed into canals, full of soldiers evacuating pensioners in rubber boats. Though the water is receding, the worst is yet to come with the assessment of damages in the Bavarian town.

By Bjorn Hengst in Passau

Sanitätshaus Fürst

Reproducibility

CEMS-Flood Structure and Governance



Setting the scene: Tight contractual operational constraints

Closed-access data

Resource-efficient

Restricted use of infrastructure

Open-source code

Reproducibility

Fully logged

Adaptive to new soft- and hardware configurations

Automatic recovery-on-failure

Platform-independently Packaged

Setting the scene: Tight contractual operational constraints

Closed-access data

Resource-efficient

Restricted use of infrastructure

Open-source code

Dynamic

Reproducibility

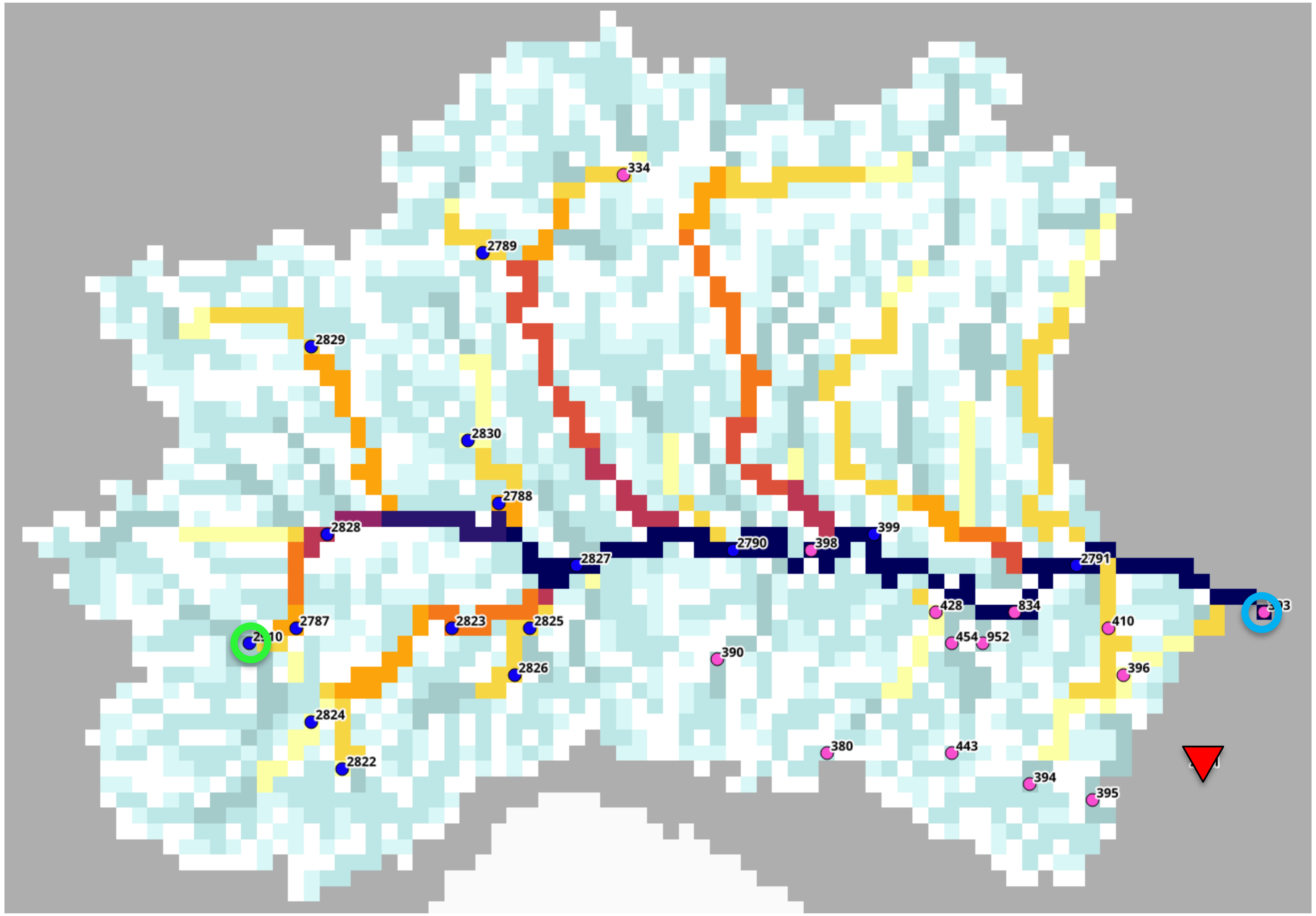
Fully logged

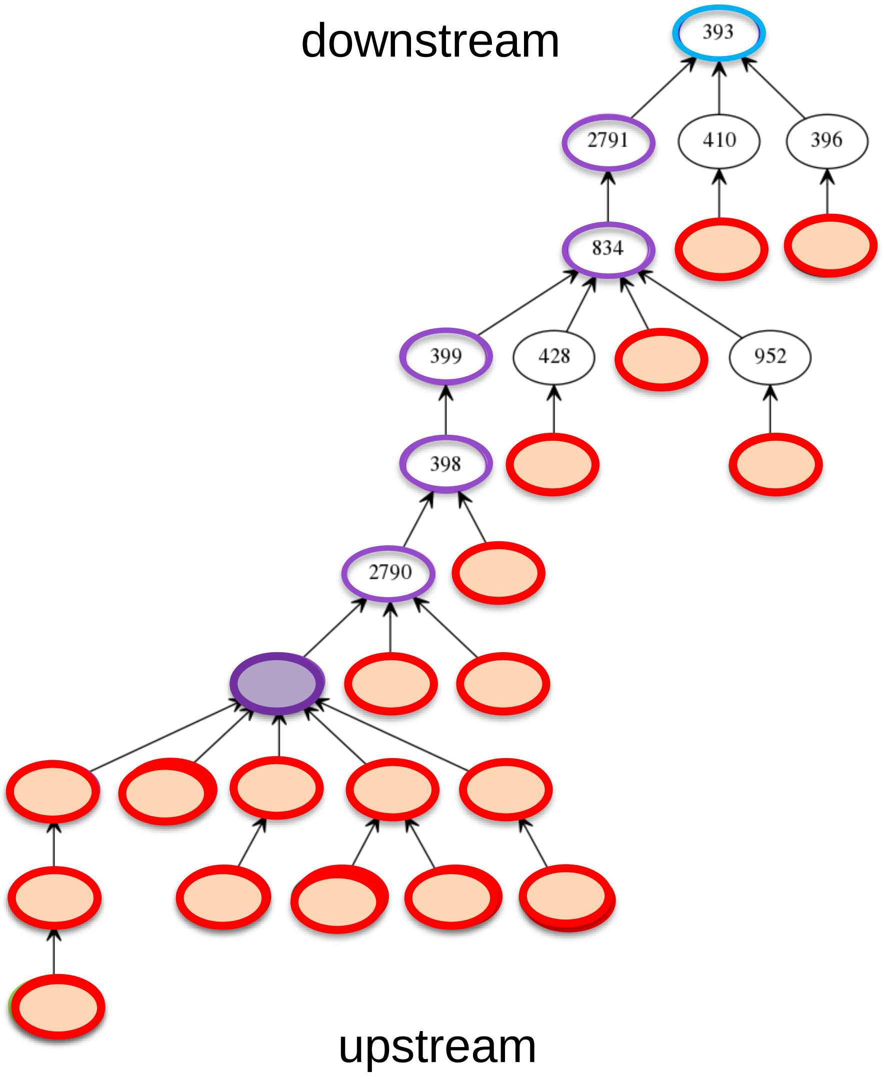
Adaptive to new soft- and hardware configurations

Automatic recovery-on-failure

Platform-independently Packaged

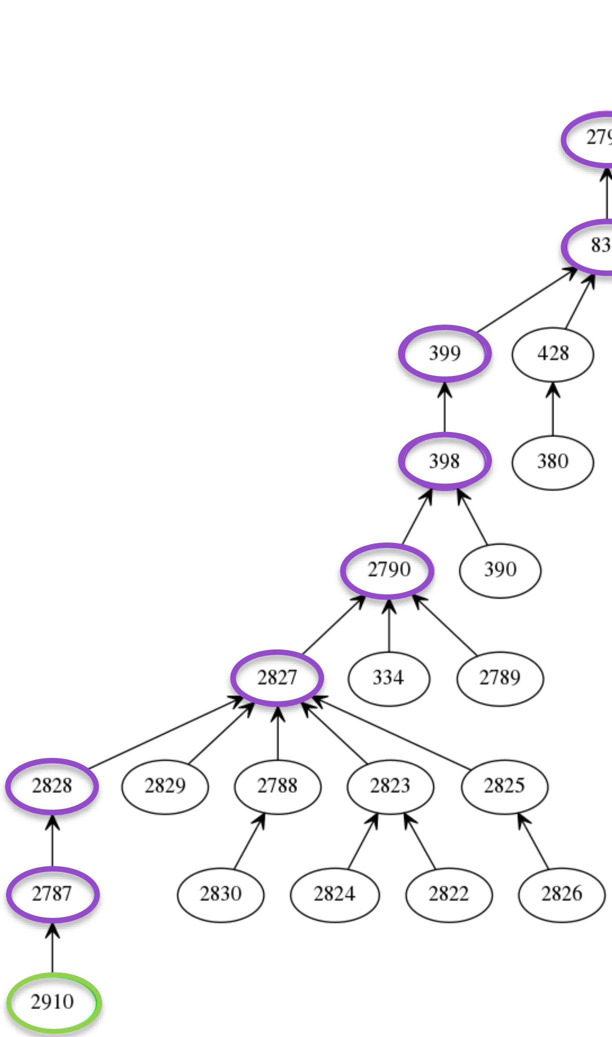
Simultaneously calibrate independent catchments





ECMWF

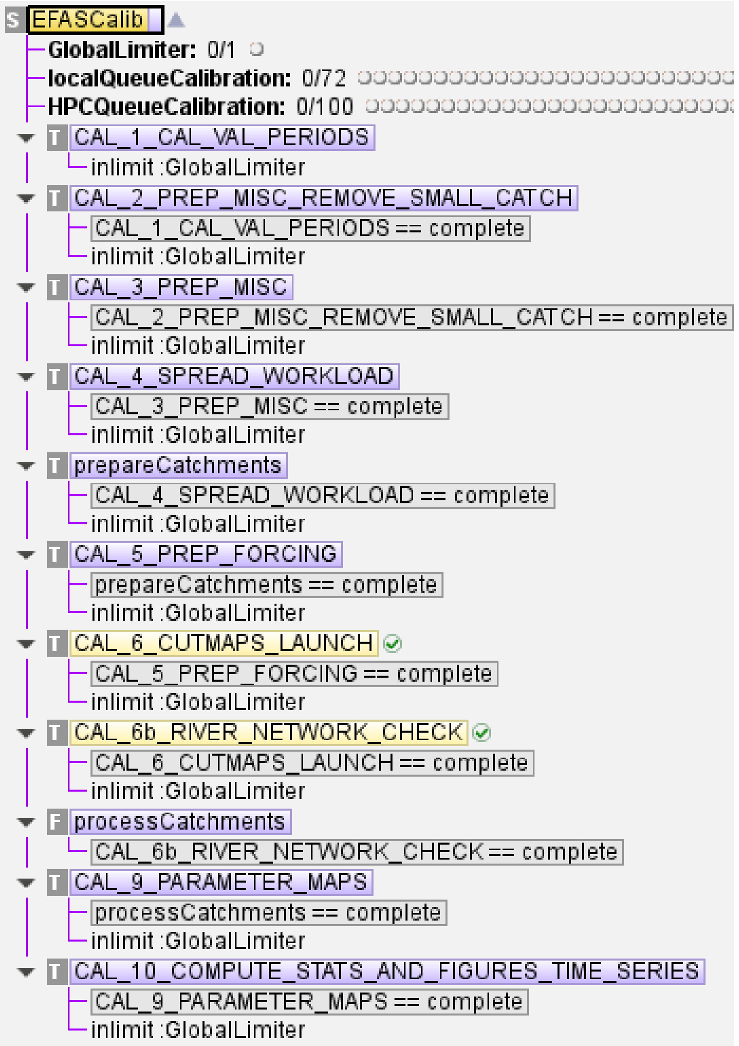
Twisting ecflow to our needs

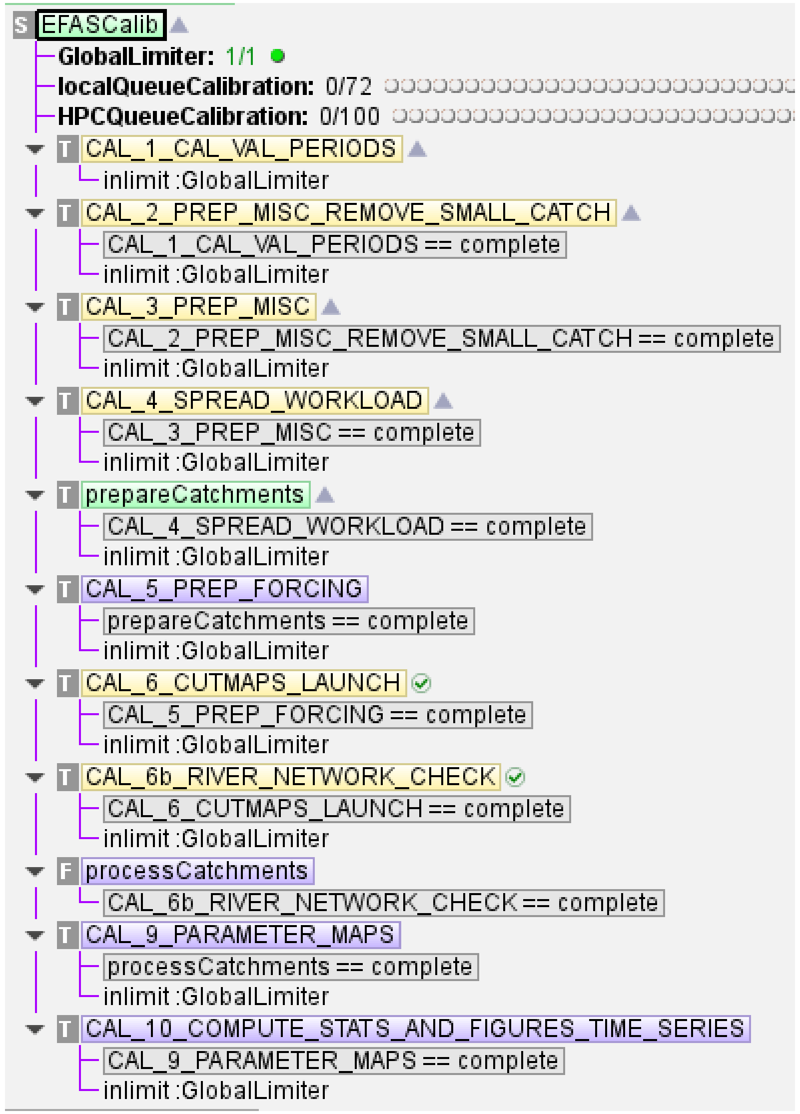


393_Po_Po_NRT_6h

- Inverted tree structure
  ==> Complicated bottom-up triggering
- Recursive suite construction

Dynamic building of the suite

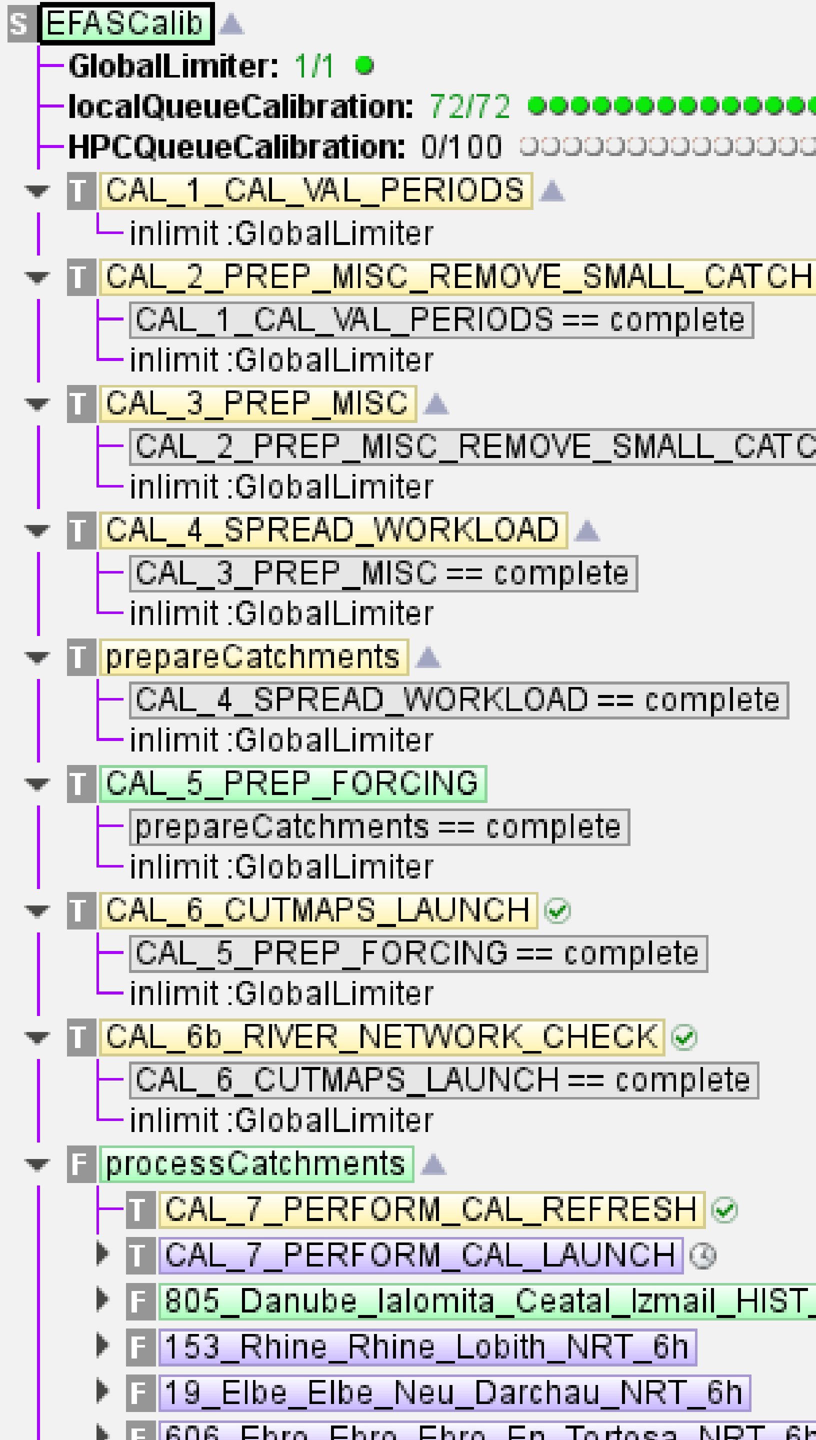




On-the-fly changes

Scientific changes to method during run:

- Interruptible/resumable runs
- Adapt to changes on river network
- Modify the Machine-learning algorithm in real time
- Additional diagnostics



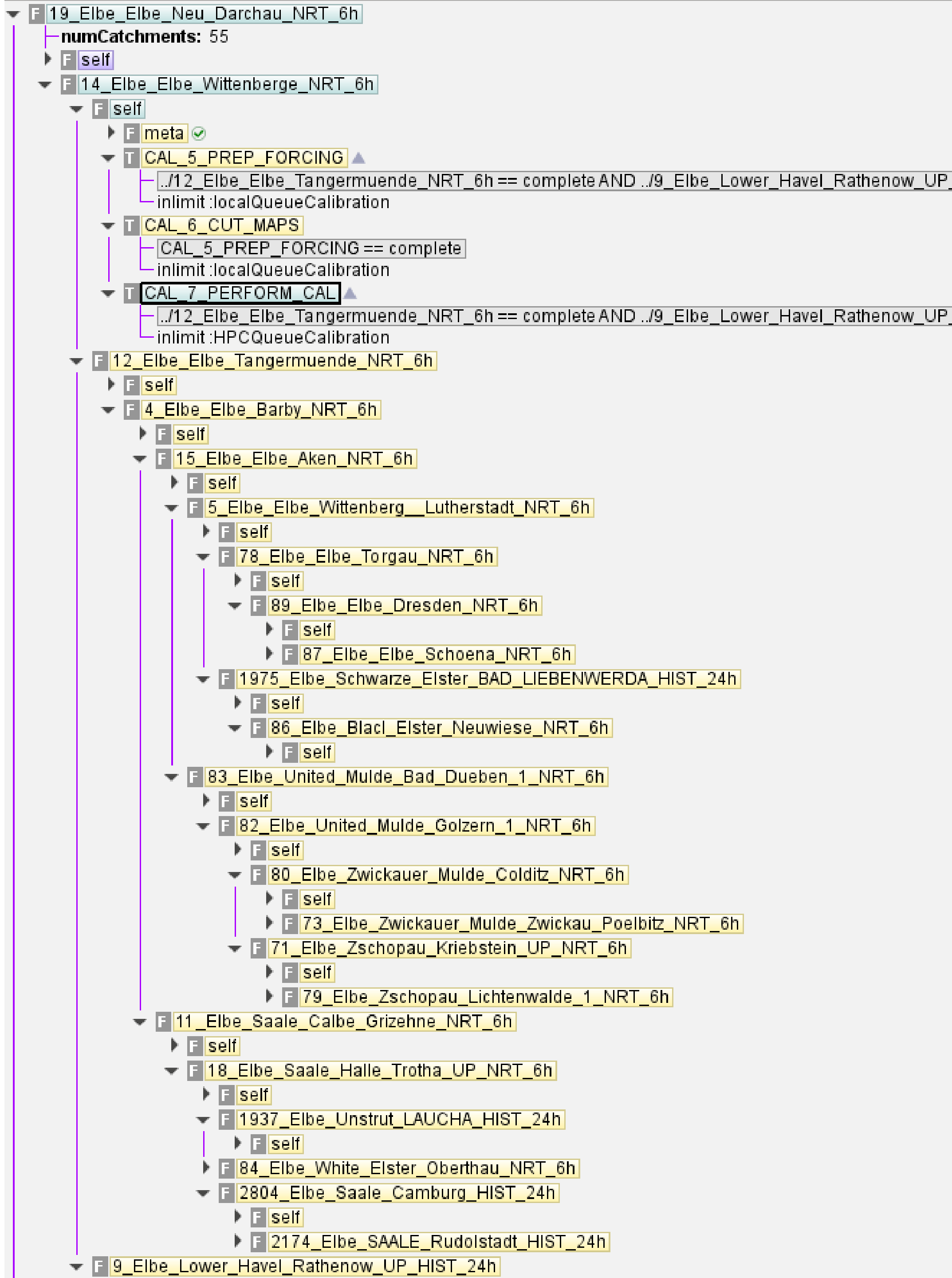
Suite definition

```
#!/usr/bin/env python3.6
import os, errno, sys, csv, re
from ecflow import Defstatus,Defs,Suite,Family,Task,Edit,Trigger,Complete,Event,Meter,Time,Day,Date,Label,RepeatString,RepeatInteger,RepeatDate,InLimit,Limit
defs = Defs(
  Suite(
    "efasCalib",
    Edit(
      {
        "ECF_INCLUDE":inc,
        "ECF_HOME":home,
        "ECF_FILES":ecf,
        "ECF_OUT":log,
        "ECF_SRC":"src",
        "ECF_TRIES":"1",
        "ECF_JOB_CMD":"mkdir -p " + log + "/%SUITE%/%FAMILY:% && mkdir -
        "ECF_KILL_CMD":"export ECF_JOB=\$(basename %ECF_JOB%) && pki
        "ECF_REMOTE_HOST":"cca.ecmwf.int",
        "ECF_REMOTE_OUT":"$TMPDIR/logs",
        "ECF_PYTHON_CMD":"/usr/local/apps/python3.6/bin/python",
      }
    ),
    Limit("localQueueCalibration",${localQueueCalibration}),
    Limit("HPCQueueCalibration",${remoteJobsHPCQueueCalibration}),
    create_tasks()
  )
)
print(defs)
print("Checking job creation: .ecf -> .job0")
print(defs.check_job_creation())
print("Checking trigger expressions")
assert len(defs.check()) == 0,defs.check()
print("Saving definition to file '${isuite}'")
defs.save_as_defs("%s/${isuite}.def" % out)
```

DATE	15.10.2019
DAY	tuesday
DD	15
DOW	2
DOY	288
ECF_CLOCK	tuesday:october:2:288
ECF_DATE	20191015
ECF_FILES	/perm/rd/nedd/EFAS/efasCalib/ecf
ECF_HOME	/perm/rd/nedd/EFAS/efasCalib/output/runscripts
ECF_INCLUDE	/perm/rd/nedd/EFAS/efasCalib/include
ECF_JOB_CMD	mkdir -p /perm/rd/nedd/EFAS/efasCalib/output/logs/%SUITE%/
ECF_JULIAN	2458772
ECF_KILL_CMD	export ECF_JOB=$(basename %ECF_JOB%) && pkill -TERM -
ECF_OUT	/perm/rd/nedd/EFAS/efasCalib/output/logs
ECF_PYTHON_CMD	/scratch/rd/nedd/python/virtualEnv_2.7.12-01/bin/python
ECF_REMOTE_CATCHMENTS	/fws12/sb/work/rd/nedd//EFAS/efasCalib/output/catchments
ECF_REMOTE_HOST	cca.ecmwf.int
ECF_REMOTE_OUT	/fws12/sb/work/rd/nedd/EFAS/efasCalib/output/remote/
ECF_REMOTE_PYTHONPATH	${PYTHONPATH}:/usr/local/apps/pcraster/4.1.0/python/
ECF_REMOTE_PYTHON_CMD	/fws12/sb/work/rd/nedd/python/virt_2.7.12-01/bin/python
ECF_REMOTE_ROOT	/fws12/sb/work/rd/nedd/
ECF_REMOTE_SRC	/fws12/sb/work/rd/nedd/EFAS/efasCalib/src
ECF_SRC	/perm/rd/nedd/EFAS/efasCalib/src/
ECF_TIME	21:25
ECF_TRIES	1
MM	10
MONTH	october
MemPerTask	1692M
SLEEP	300
SUITE	EFASCalib

Pythonesque suite components

```
def createTasks():
  allFamilies = []
  allFamilies += [
    Family [
      "CAL_RHINE“,
      addTask("CAL_7_PERFORM_CAL")
  ]
  return allFamilies

def addTask(thisTask):
  return [
        Task(
          InLimit("HPCQueueCalibration"),
          Edit(
                {
                  "settingsFile":“%remoteJobsCalibSrc%/settings_calibration.txt",
                  "MemPerTask":"1440M",
                  "remoteJobsHPCNumCores":72
                }
          ),
          thisTask,
          triggers
        )
  ]
```



Any ecf script

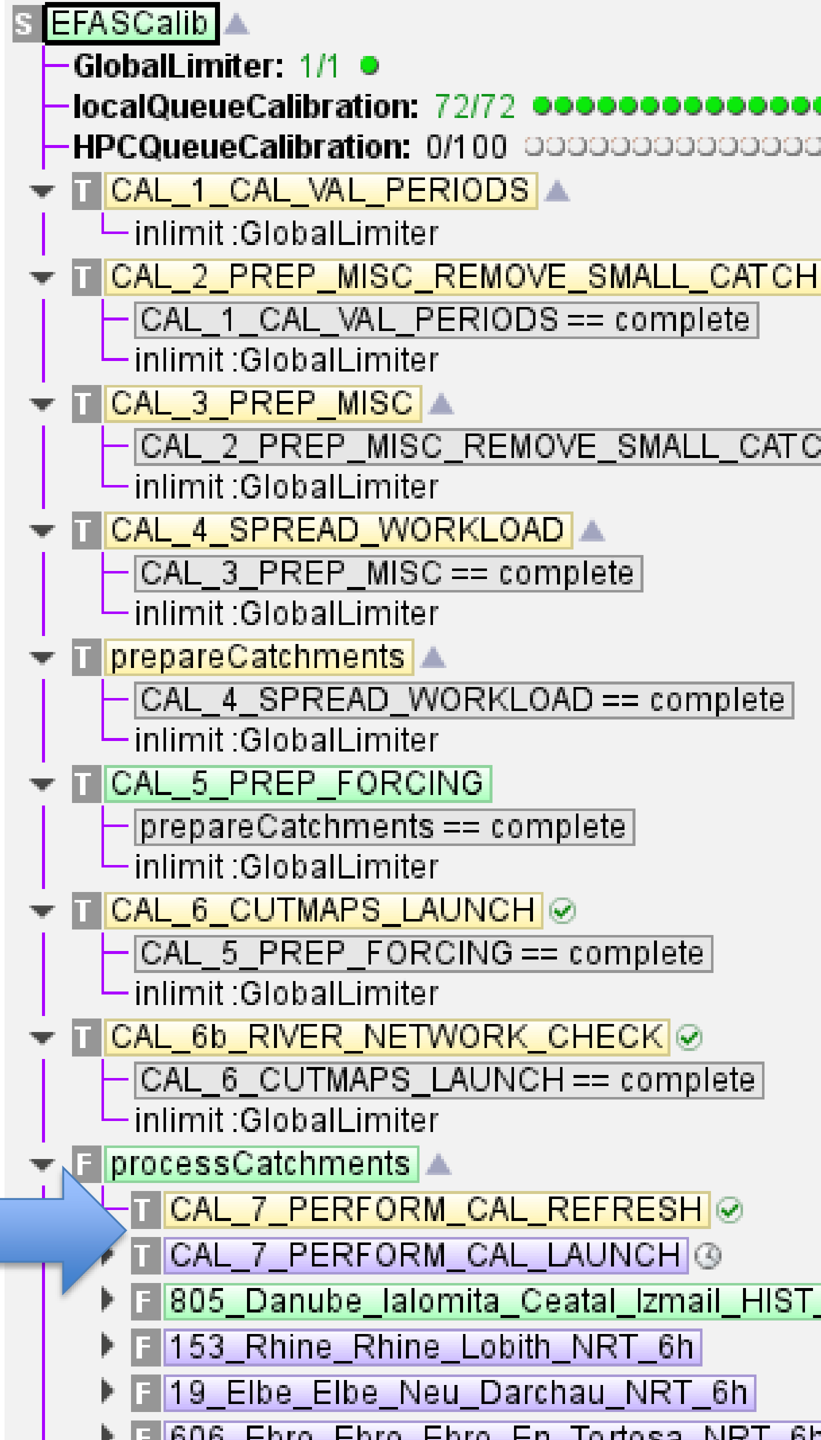
```
#!/bin/ksh

%include <head.h>

%ECF_PYTHON_CMD% "%pyFile%" %settingsFile%

%include <tail.h>
```

Job file

```
#!/bin/ksh

# Tell ecFlow we have started
ecflow_client --init=\$\$ 2>/dev/null
…
module load gdal/2.1.1
module load pcraster/4.1.0

/usr/local/apps/python3.6/bin/python /perm/rd/nedd/EFAS/efasCalib/src/CAL_7_PERFORM_CAL.py /perm/rd/nedd/EFAS/efasCalib/src/settings.txt

# Tell ecFlow the task is done
ecflow_client --complete 2>/dev/null  # Notify ecFlow of a normal end
…
exit 0
```

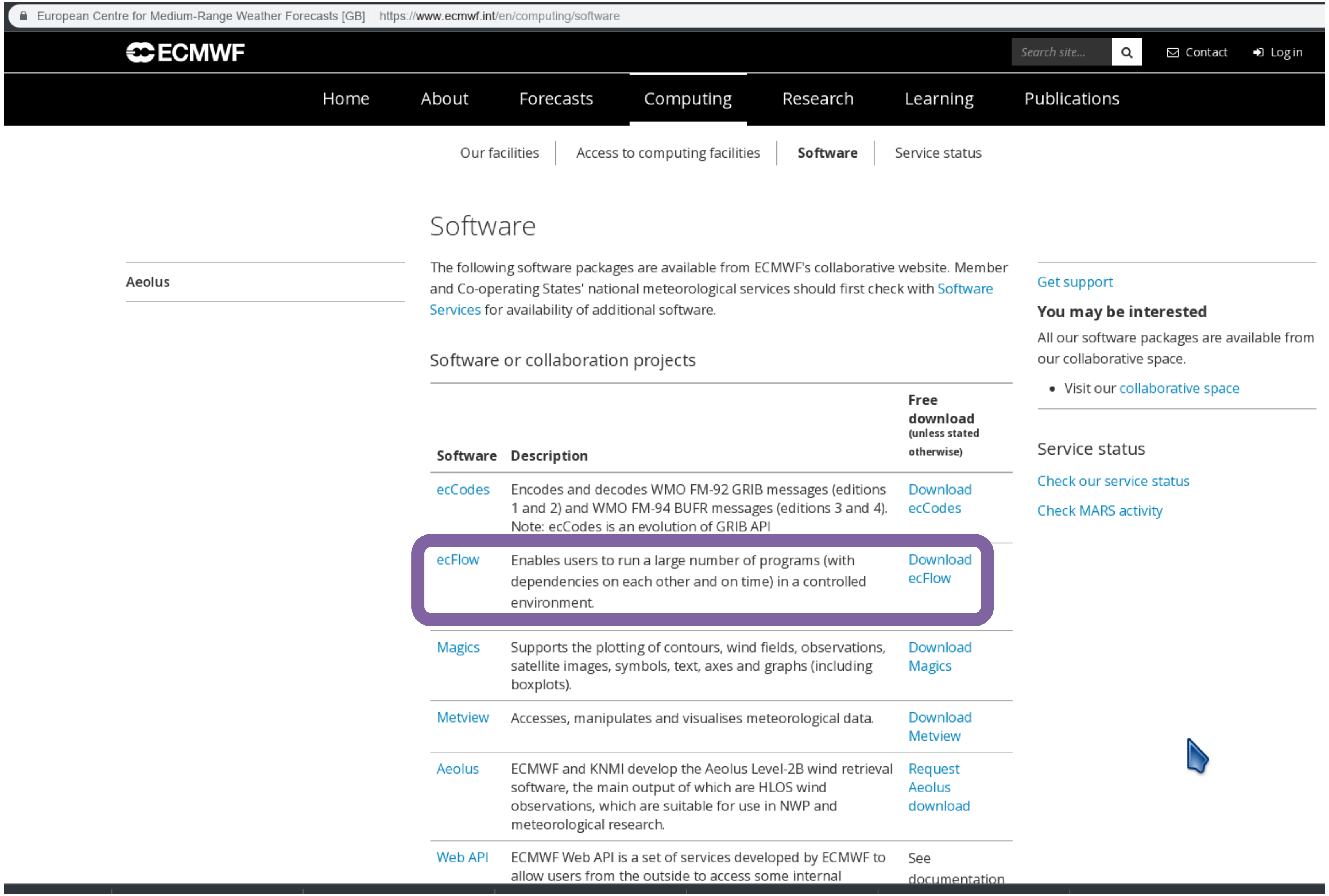
Practical complications

- HPC compute nodes isolated from network:
  - Mechanism to package jobs to run remotely…
  - … and to retrieve the results
  - Efficient use of nodes ➔ load-balancing system
  - Monitoring: check running state every n mins
  - Automatic switching to different computer on failure
  - Restore state after network or server crashes



Summary

Download from https://www.ecmwf.int/en/computing/software



Prerequisites:

- Qt
- cmake
- boost if using Python API

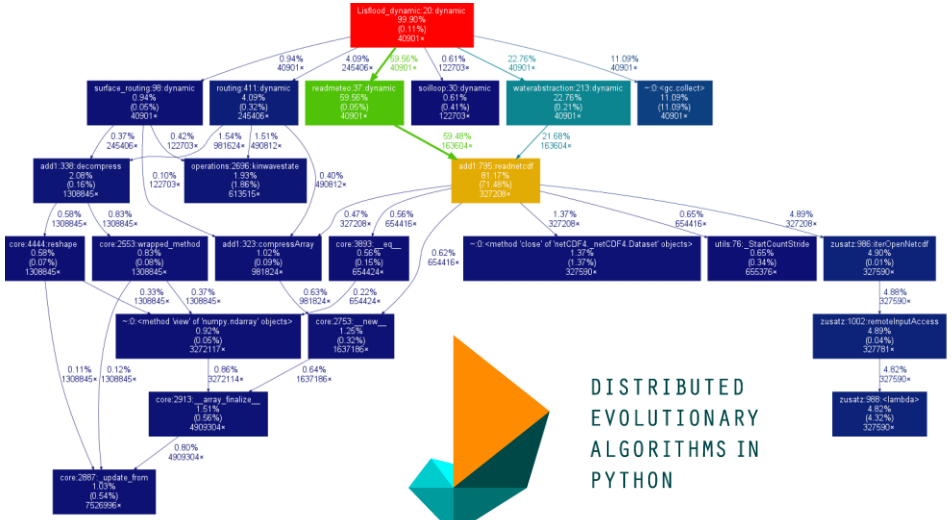
Summary

- Simplicity is key to monitor and maintain, use simple constructions
- Tailored to the needs with minimal footprint
- Shareable workflows are crucial
- Flexibility
- Identify restrictions on code, data, etc.

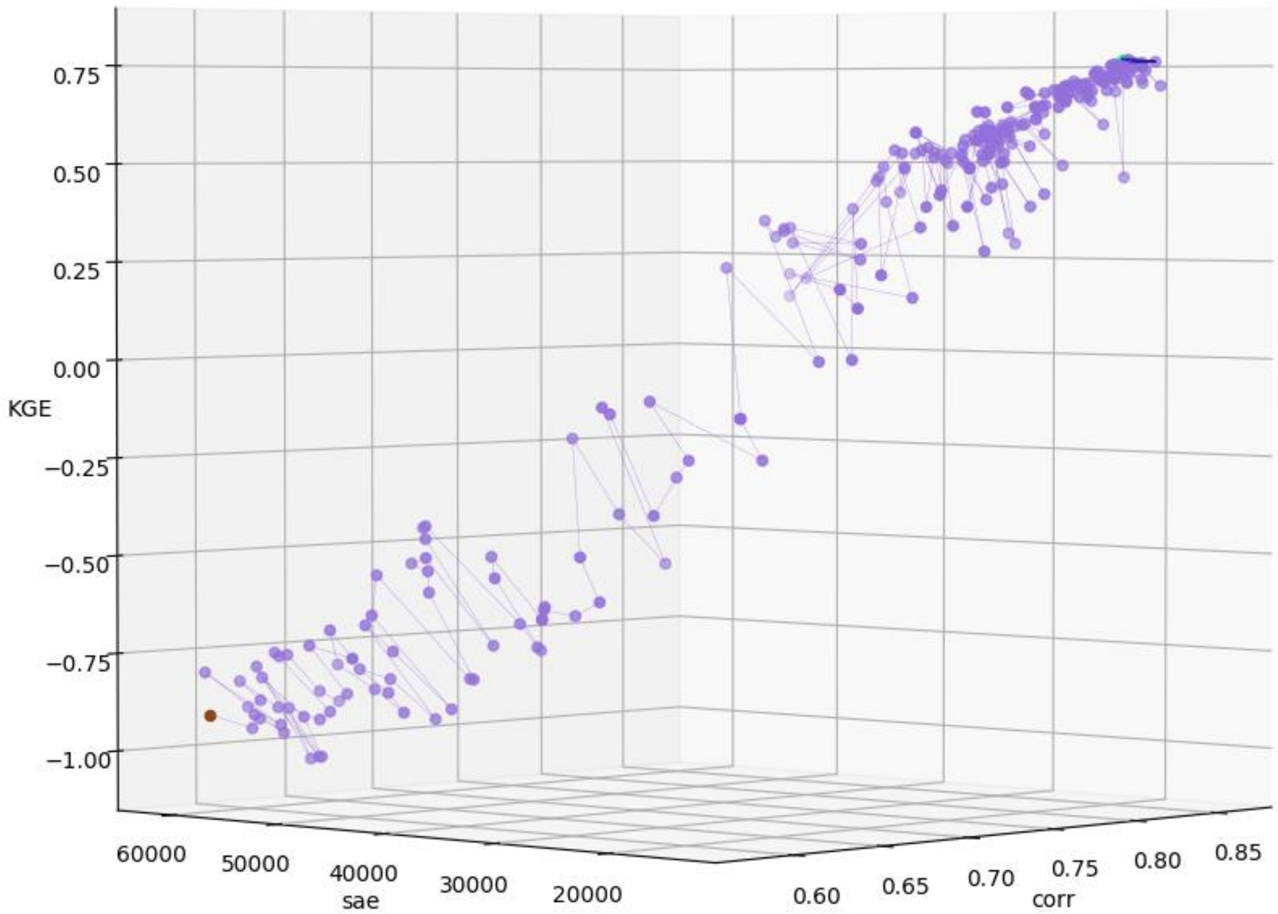
- For the future:
  - Higher-level suite constructions?
  - Keep abstraction manageable!

Methodology

- Bulky stand-alone code not fit to be run 400 to 700 times
- Profiling revealed 60% of time spent on I/O
  ➔ Rewrite I/O to load input data in memory once
- Processing in load-balanced paired batches to optimise HPC usage



DISTRIBUTED EVOLUTIONARY ALGORITHMS IN PYTHON



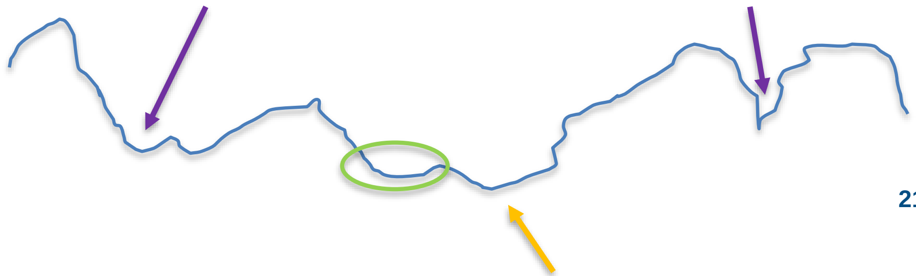
- Cost function:

$$KGE' = 1 - \sqrt{(r-1)^2 + (\beta - 1)^2 + (\gamma - 1)^2}$$

$$\beta = \frac{\mu_s}{\mu_0}$$

$$\gamma = \frac{CV_s}{CV_o} = \frac{\sigma_s/\mu_s}{\sigma_o/\mu_o}$$

- 13 calibration parameters
- Genetic Machine-Learning algorithm
- Uneven real weights
- Lack of hydrologic sense ➔ smart filtering



head.h

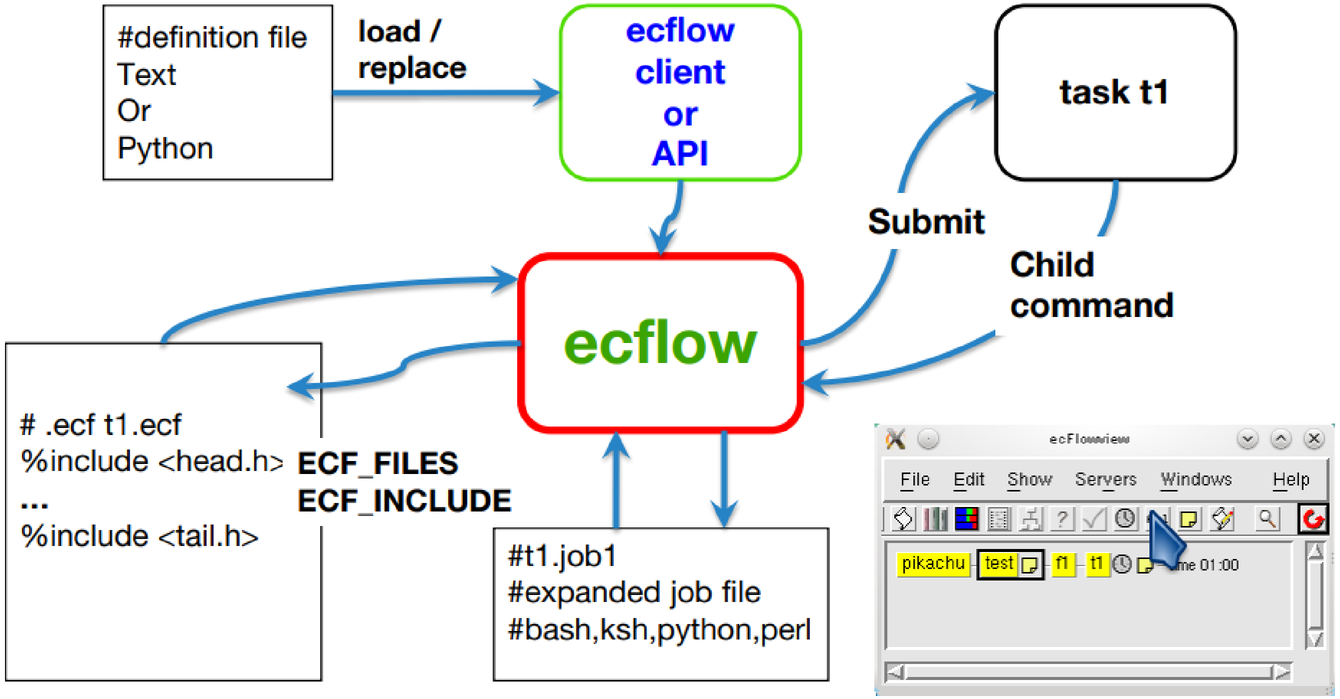
```
# Make a kick-arse prompt which gives many more useful details during running
PS4='   <-- \\\$(date +\|%g%m%d\|%H:%M:%S\|)\\\${SECONDS}\|\\\${0}:L\\\${LINENO} -->'

# Tell ecFlow we have started
ecflow_client --init=\$\$ 2>/dev/null

# Define a error handler
ERROR() {
  set +e                      # Clear -e flag, so we don't fail
  wait                        # wait for background process to stop
  ecflow_client --abort=trap  2>/dev/null # Notify ecFlow that something went wrong, using 'trap' as the reason
  if [[ "\${HOSTNAME: }" == "%ECF_HOST%" ]]; then
    echo FAIL
  else
    # Then copy the log back to the server's job output files so we can consult them directly in ecflow
    sleep 5; rsync -av %ECF_REMOTE_JOBOUT% %ECF_HOST%:%ECF_JOBOUT%.%ECF_TRYNO%.log
  fi
  trap 0                      # Remove the trap
  exit 0                      # End the script
}
# Trap any signal that may cause the script to fail
trap '{ echo "Killed by a signal"; ERROR ; }' 1 2 3 4 5 6 7 8 10 12 13 15

module load gdal/2.1.1
module load pcraster/4.1.0
```

tail.h

```
set +e
wait                      # wait for background process to stop
set +u
ecflow_client --complete 2>/dev/null  # Notify ecFlow of a normal end
if [[ "\${HOSTNAME: }" != "%ECF_HOST%" ]]; then
   # Copy the log back to the server's job output files so we can consult them directly in ecflow
   sleep 5; rsync –av %ECF_REMOTE_JOBOUT% %ECF_HOST%:%ECF_JOBOUT%%ECF_TRYNO%
fi
set –u
trap 0                    # Remove all traps
exit 0                    # End the shell
```

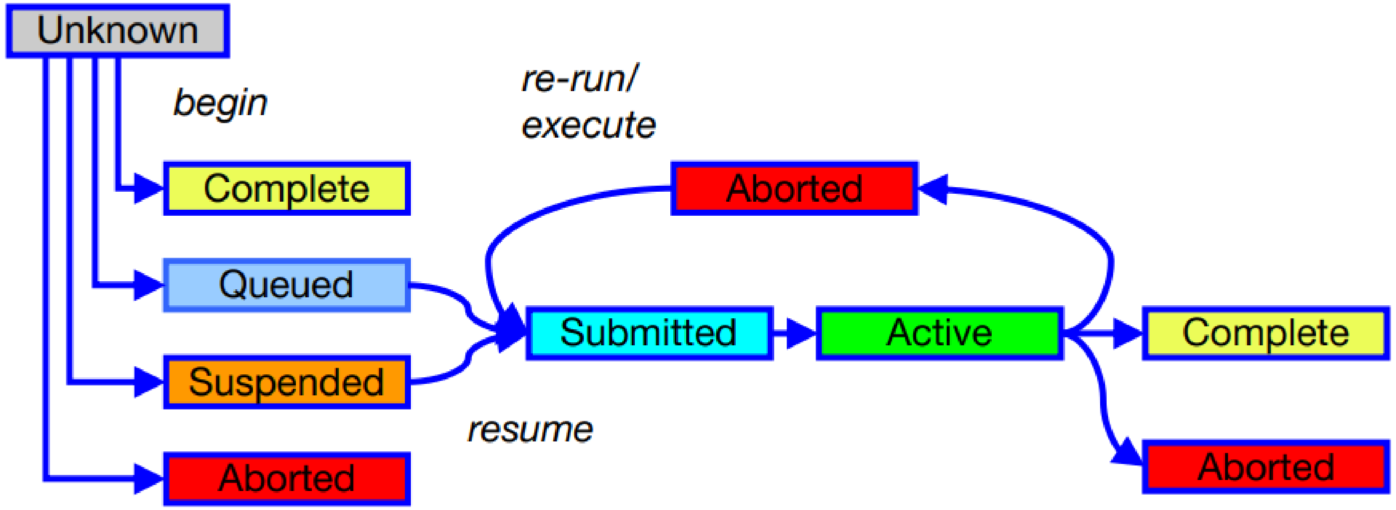
ecFlow to the rescue

- Work-flow manager consisting of:
  - ecFlow server (C++/boost ASIO) to receive requests from clients
  - Client software interfaces through:
    - GUI
    - Command line tools
    - Python API
- Stores task states
- Handles relationships between tasks
- Platform independent
- Embedded suite definition checking
- Manages deadlocks and zombies
- On-the-fly suite updates



- Manual & Training material available at https://confluence.ecmwf.int/display/ECFLOW/Documentation

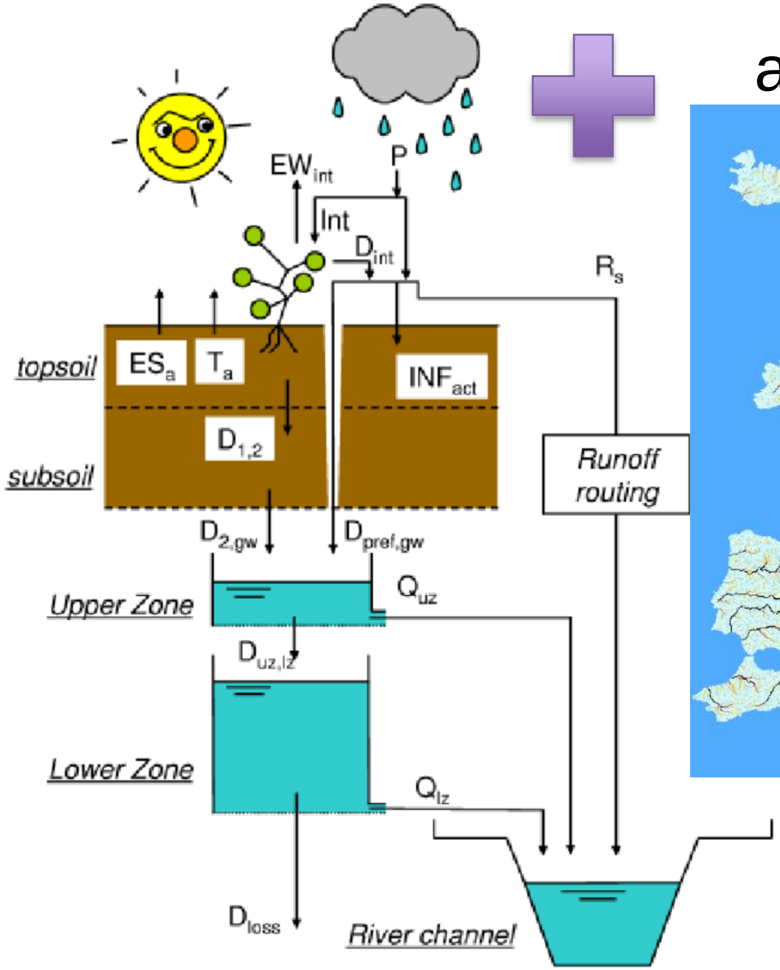
State driven work-flow management



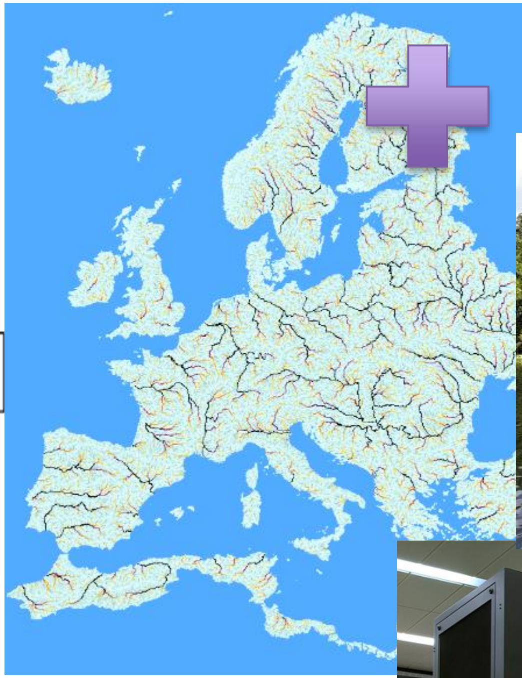
Fascinating story

Reproducible workflow

Imperfect hydrologic models



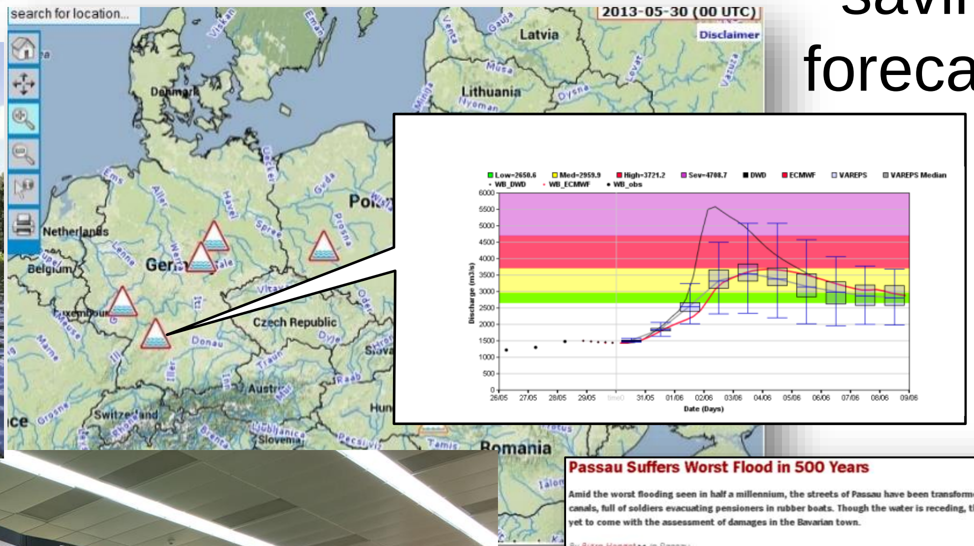
and river network



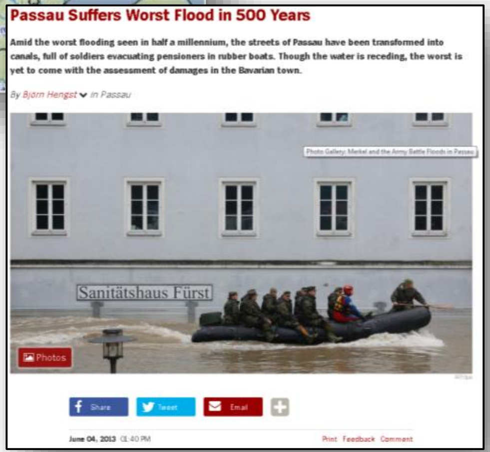
calibration

obs

Life-saving forecasts



on HPC


Passau Suffers Worst Flood in 500 Years

Amid the worst flooding seen in half a millennium, the streets of Passau have been transformed into canals, full of soldiers evacuating pensioners in rubber boats. Though the water is receding, the worst is yet to come with the assessment of damages in the Bavarian town.

By Bjorn Hengst in Passau

June 04, 2013 01:40 PM