# A new blueprint for next generation land-surface and hydrological predictions

## Conceptual & Technical Overview of the Modelling Framework

**Hallouin T.**, Ellis R., Lawrence B.N., Clark D., Blyth E., Polcher J., Hughes A., Lister G. and Dadson S.

thibault.hallouin@ncas.ac.uk github.com/ThibHlln twitter.com/ThibHlln

11th September 2020
JULES Open Science Meeting
Online

Hydro-JULES
UK Centre for Ecology & Hydrology
British Geological Survey
National Centre for Atmospheric Science
NATURAL ENVIRONMENT RESEARCH COUNCIL

# Current limitations in the modelling of hydrological processes in JULES

JULES was historically developed as a lower boundary condition to UM:

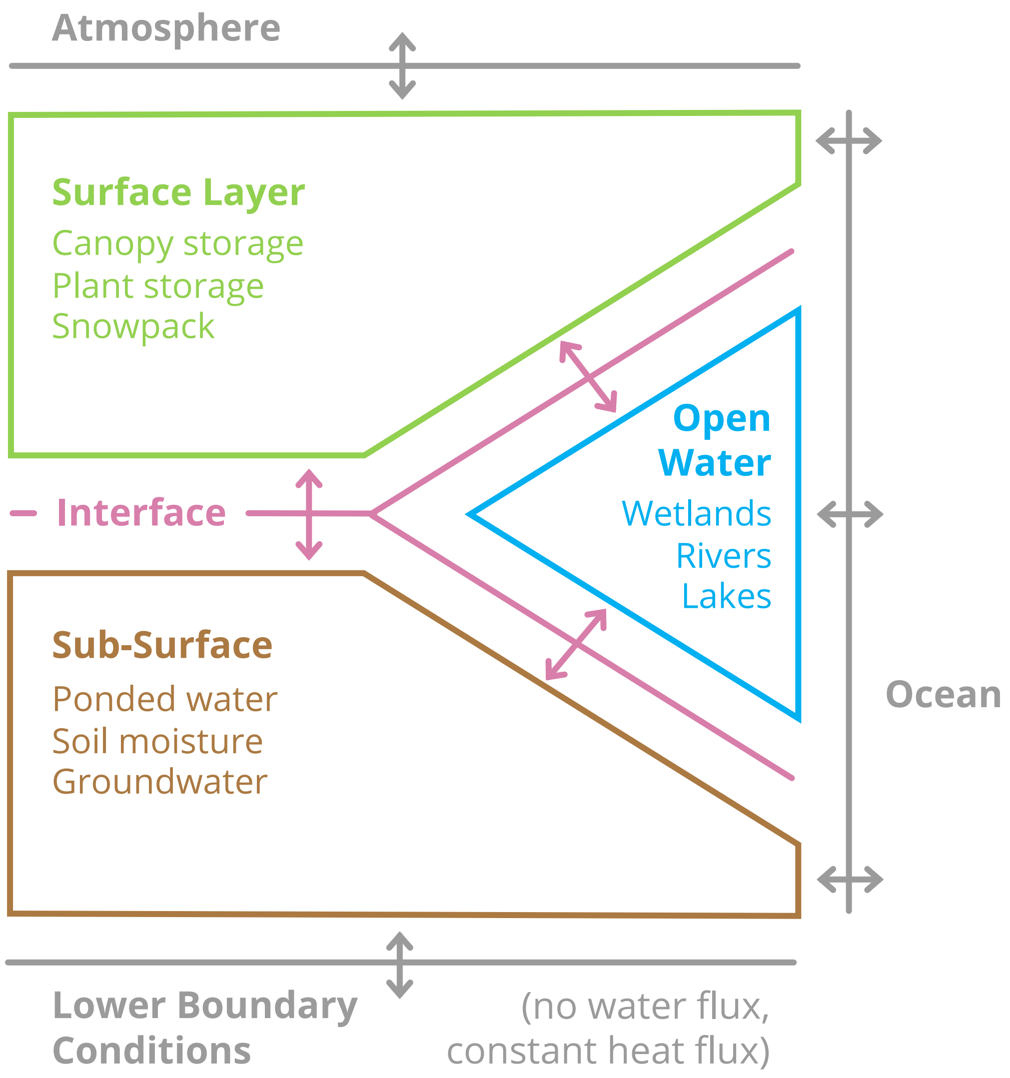
- inherited the resolution of the atmospheric grid
  - ↘ land heterogeneity and hydrological structures within the atmospheric grid accommodated for by using a large variety of sub-grid approaches
- inherited a column-based approach (i.e. vertical exchanges)
  - ↘ absence of lateral flows between grid cells
- overlooked the two-way interaction between the land and the ocean
  - ↘ absence of ocean feedbacks such as tides, storm surges, etc.

## Hydro-JULES contributions to these limitations

The Hydro-JULES aims to tackle these limitations by providing:

- a new modelling framework for the terrestrial water cycle
  - ↘ a modular representation of the water cycle
  - ↘ interchangeable modules (referred to as components)
  - ↘ possible two-way communication with other models (climate, ocean)
- a repository of components, including:
  - ↘ a modular version of JULES
  - ↘ new groundwater models
  - ↘ CaMa-Flood

# A modular representation of the terrestrial water cycle



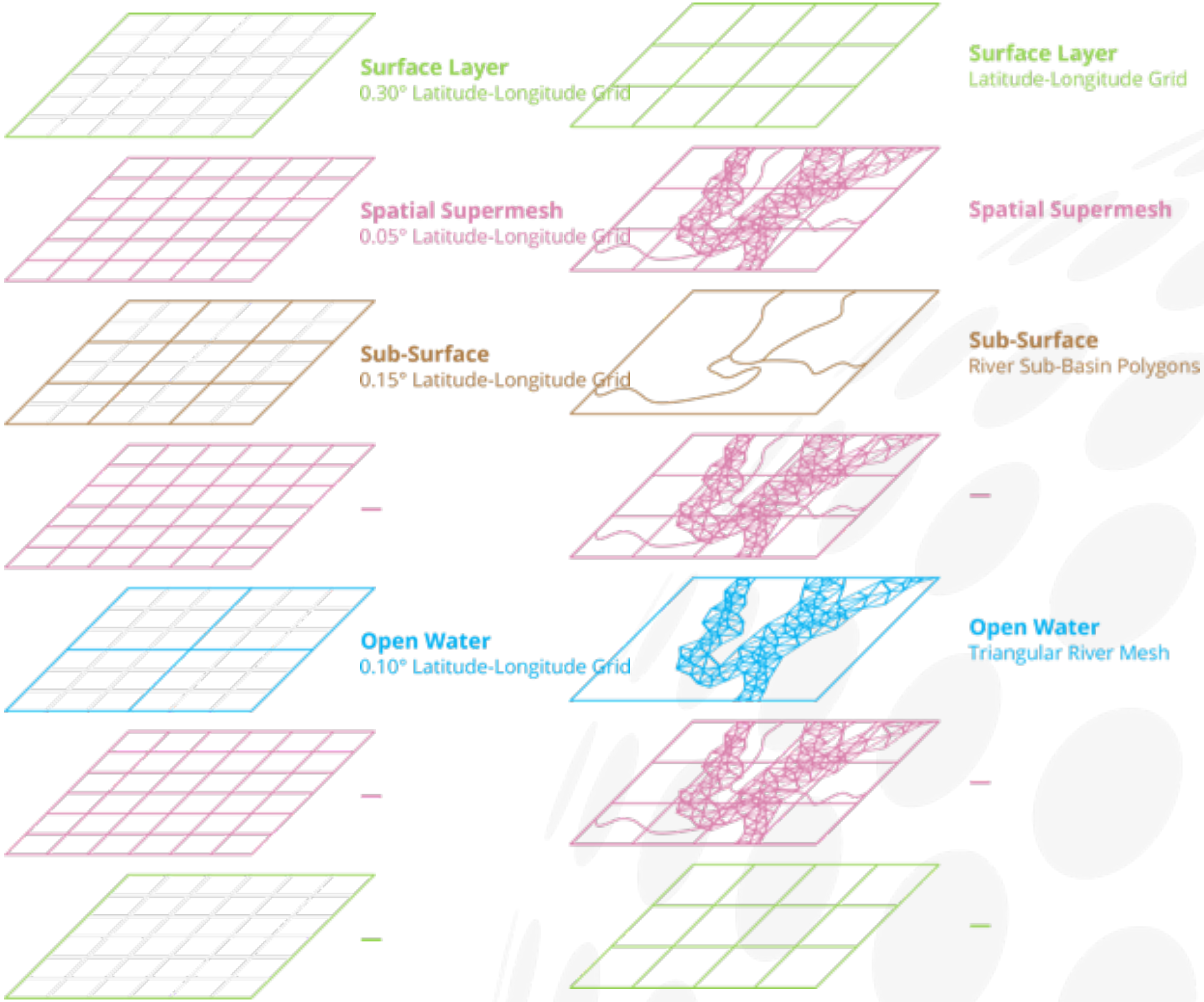

*Note, this is a draft version of the framework subdivisions.*

- subdivide the land system into components whose resolutions are adapted to the equations they aim to solve:
  - spatial resolution can be adapted to the dominant structures of heterogeneity
  - temporal resolution can be adapted to the timescale of the dominant processes
- for each component, all processes governing the energy, water and biogeochemical cycles are treated within a common numerical framework
- each scientific community (and their respective models and expertise) should map onto one or more components

# Flexibility in the resolutions of the components

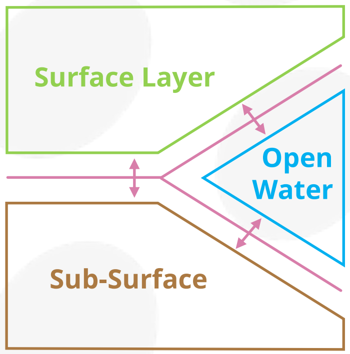
## Space

- each component can feature its own spatial discretization
- a spatial "supermesh" is determined from the resolution of the components to preserve the continuity equations
- the grid cells/polygons can communicate with their neighbours (i.e. lateral flow, not only vertical)



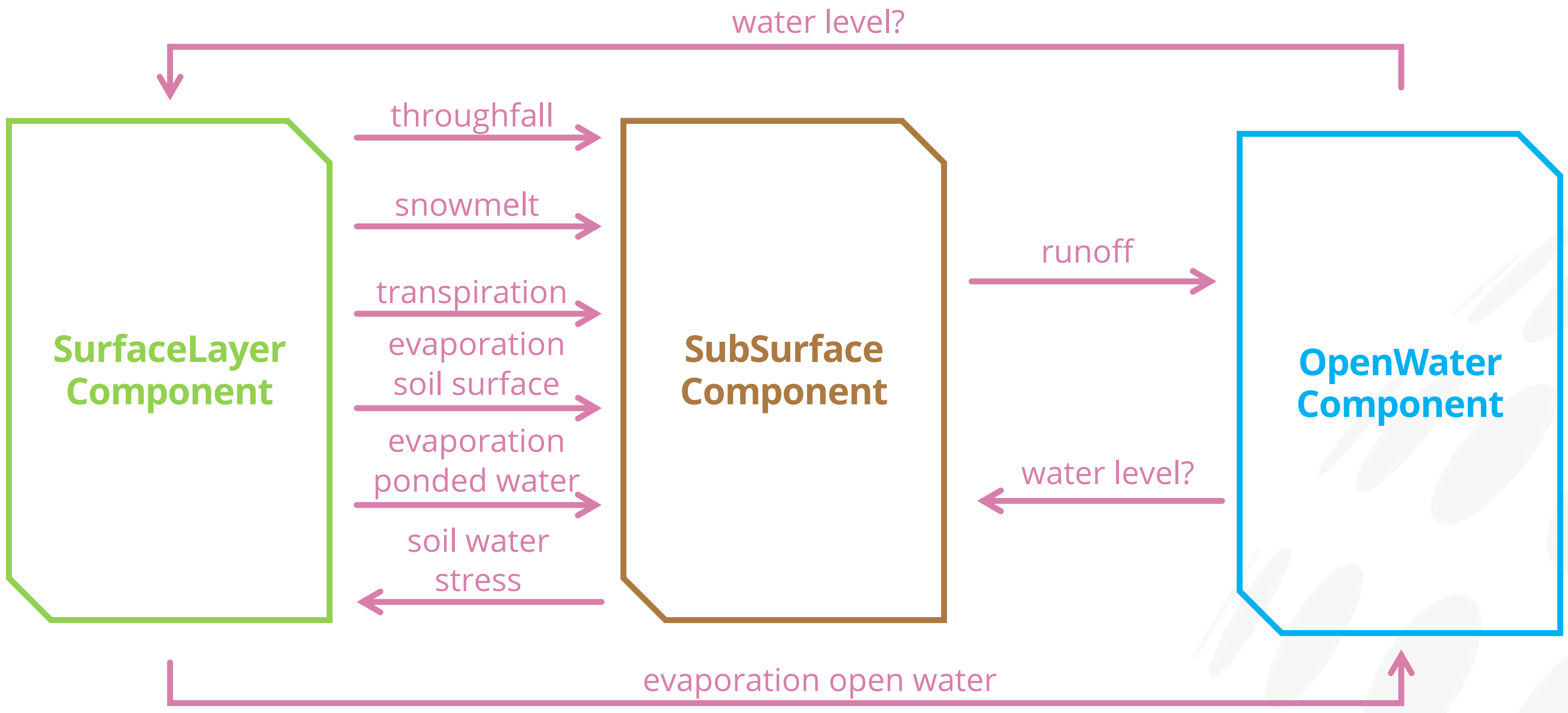

## Time

- each component can feature its own regular temporal resolution

# A fixed interface of transfers between components

With an initial focus on the water cycle



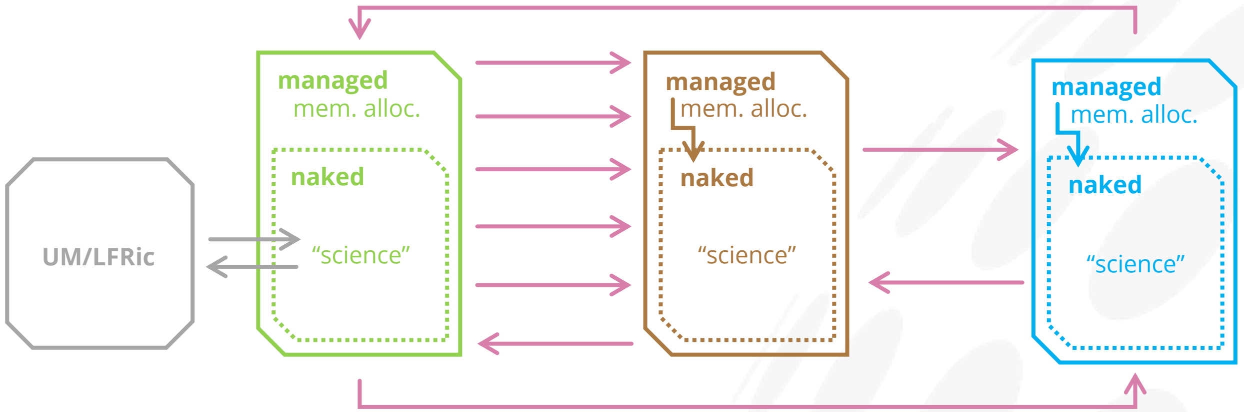

- each component must comply with a fixed interface (i.e. information to produce, information to incorporate)
- the information to exchange (i.e. transfers) can be fluxes, ratios (e.g. soil water stress), and maybe states (water level)?
- the interface may feature a set of transform functions to convert component specific information to the set interface information?

*Note, this is only a draft version of the interface.*

# Allowing for communication with external models

**The modelling framework will implement a “nested API” for each Component (i.e. a ”managed API” around a “naked API”)**
(as per UK Met Office concept for JULES: https://code.metoffice.gov.uk/trac/jules/wiki/NestedAPIDiscussion)

The underlying objective being to allow **two manners of handling the memory** that Components require to sustain information between modelling time steps:




“Naked API”

- memory allocated and handled **outside of the Component** (i.e. by the calling model)

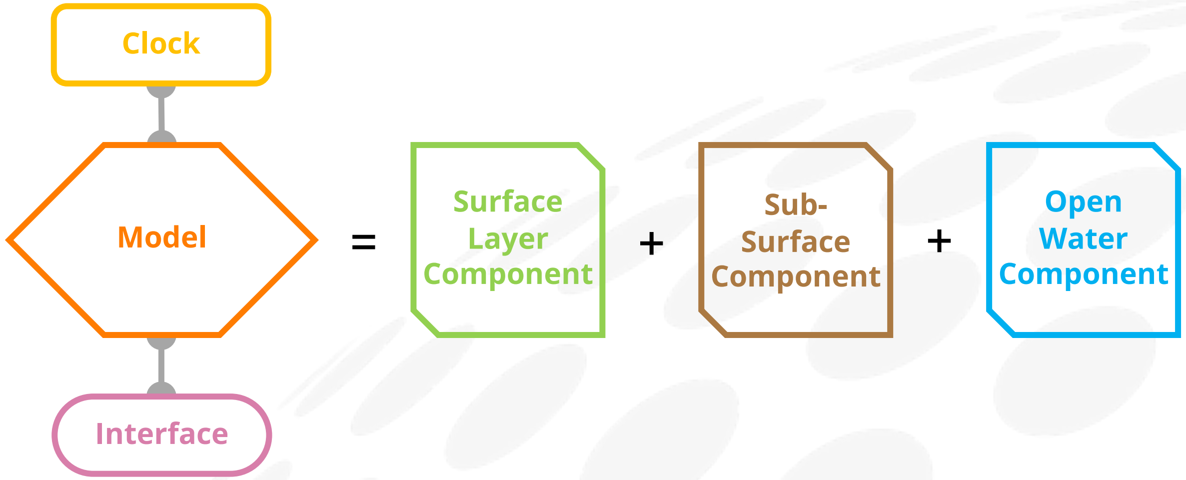
”Managed API”

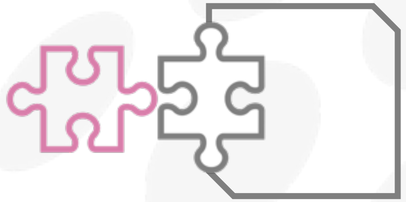
- memory allocated and handled **inside the Component** (i.e. by the Component itself)

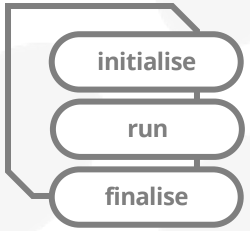
# A Python package as a first implementation of the framework blueprint

## Core concepts of the Python implementation

- each Component is an object (of 3 different types: SurfaceLayer, SubSurface, OpenWater)
- a Model object is allowing the communication between each Component, it namely:
  - features an Interface (responsible for the exchange of information and the remapping)
  - features a Clock (responsible for the time-stepping)
- each Component must comply with the fixed Interface (information in, information out)
- each Component must be implemented following the "initialise-run-finalise" paradigm






## Technical aspects of the Python implementation

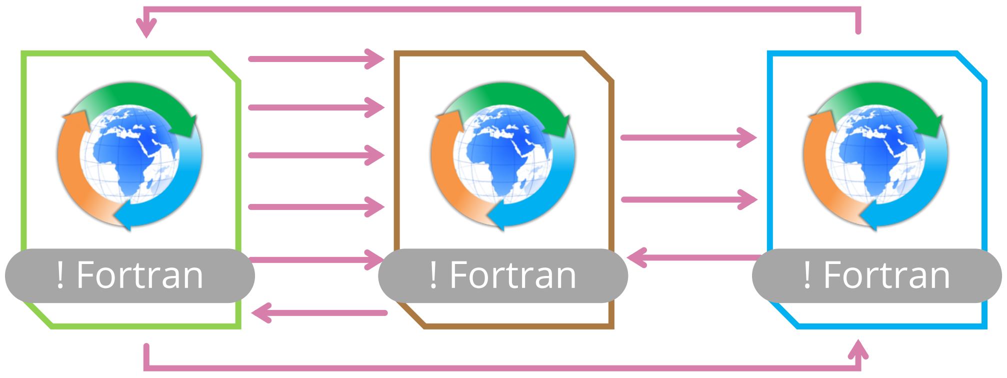
- able to run simulation in parallel (using MPI protocol)

# Supporting Component's initialise-run-finalise to be in Python-C-Fortran

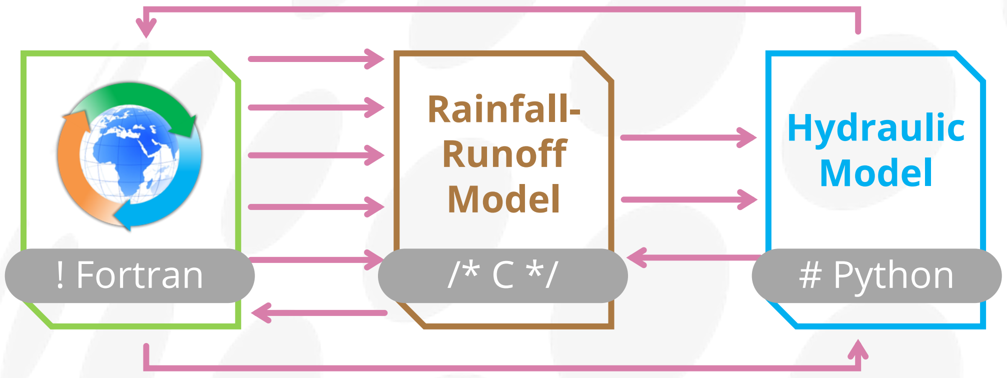
**Three languages are supported by the Python implementation:**

- Python (trivial)
- Fortran using numpy.f2py to compile the Fortran code
- C using NumPy C-API and *e.g.* Cython to compile the C code

**example 1**
running the framework with all the components from JULES




**example 2**
combining JULES with a rainfall-runoff model and a hydraulic model



# Summary

Hydro-JULES aim to provide:

- a new blueprint subdividing the terrestrial water cycle into components whose spatial and temporal scales are adapted to the equations of the dominant hydrological processes they are trying to solve
- a first implementation of this blueprint as a Python package (with possibility for Fortran and/or C extensions)
- the complete incorporation of JULES in the framework as distinct components
- a framework that can communicate in a two-way fashion with atmospheric and ocean models (while allowing flexibility on memory allocation ownership)
- a framework that supports and promotes the development and comparison of model components

github.com/hydro-jules

twitter.com/HydroJules

Hydro-JULES

UK Centre for Ecology & Hydrology

National Centre for Atmospheric Science
NATURAL ENVIRONMENT RESEARCH COUNCIL

British Geological Survey