# PyTelTools: Python scripts and GUI to automate Telemac post-processing tasks

Luc Duron, Yishu Wang

Compagnie Nationale du Rhône (CNR)

XXIV$^{th}$ TELEMAC-MASCARET User Conference

Graz, Austria, 17-20 October, 2017

Python Telemac Tools

# Table of Contents

# Project initiation

## Compagnie Nationale du Rhône (CNR)

- 1st producer of exclusively renewable energy in France
- 18 hydroelectric facilities on the Rhône River (3000 MW)
- CNR Engineering Departement (for CNR and third party)

# Project initiation

## Compagnie Nationale du Rhône (CNR)

- 1st producer of exclusively renewable energy in France
- 18 hydroelectric facilities on the Rhône River (3000 MW)
- CNR Engineering Departement (for CNR and third party)

## Needs and purpose

- Develop non-existing features, customize it for modeller needs
- Automate and chain Telemac post-processing tasks
- Do not substitute for common post-processing graphical softwares

# Python Telemac Tools

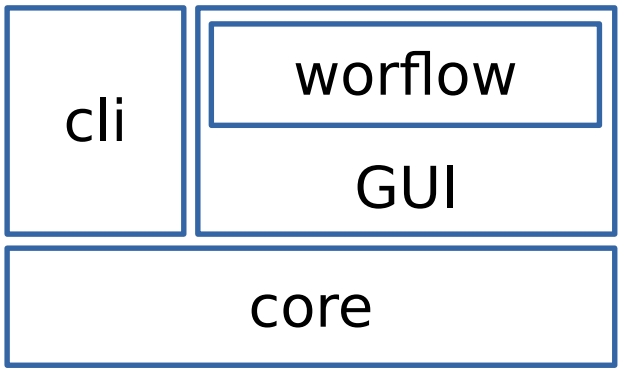
## Developement Guidelines

- extensible
- customizable
- accessible, easy to use
- robustness
- open-source

# Python Telemac Tools

## Developement Guidelines

- extensible
- customizable
- accessible, easy to use
- robustness
- open-source




## Implementation overview

- **core**
  - parsers, base classes, ...
  - mathematical and spatial calculations
- Command Line Interface (**CLI**)
  - multiple arguments
  - easily adapted, snippets (being implemented)
- Graphical User Interface (**GUI**)
  - classic interface
  - **Workflow** : automate, chain and monitor tasks

# Documentation and installation

## Installation

- Prerequisites: Python3
- Packages: see requirements.txt (numpy, matplotlib, PyQt5, ...)
- Installation procedure:

```
# user install
pip install -e git://github.com/CNR-Engineering/PyTelTools.git#egg=PyTelTools --user
# default install (needs to be root)
pip install -e git://github.com/CNR-Engineering/PyTelTools.git#egg=PyTelTools
```

## Documentations

- User: `https://github.com/CNR-Engineering/PyTelTools/wiki`
- Developper: `https://cnr-engineering.github.io/PyTelTools`

# Classification of post-processing tasks

- File conversions (shp, vtk, ...)

# Classification of post-processing tasks

- File conversions (shp, vtk, ...)
- Compute variables

| | ID | Name | Unit |
|---|---|---|---|
| 1 | U | VITESSE U | M/S |
| 2 | V | VITESSE V | M/S |
| 3 | S | SURFACE LIBRE | M |
| 4 | B | FOND | M |
| 5 | W | FROTTEMENT | |

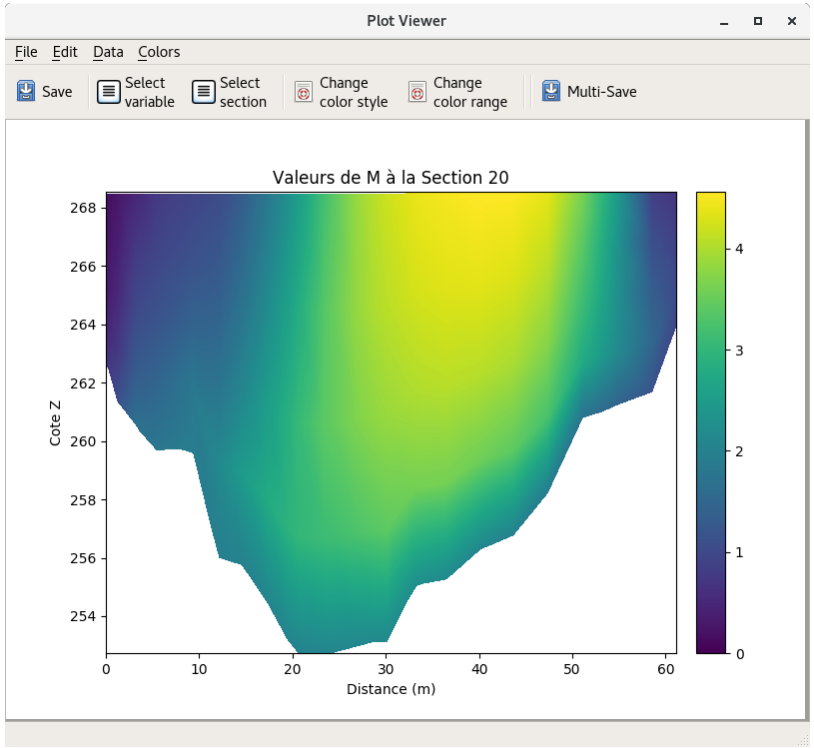
| | ID | Name | Unit |
|---|---|---|---|
| 1 | H | HAUTEUR D'EAU | M |
| 2 | M | VITESSE SCALAIRE | M/S |
| 3 | I | DEBIT SUIVANT X | M2/S |
| 4 | J | DEBIT SUIVANT Y | M2/S |
| 5 | Q | DEBIT SCALAIRE | M2/S |
| 6 | C | CELERITE | M/S |
| 7 | F | FROUDE | |
| 8 | US | VITESSE DE FROT. | M/S |
| 9 | TAU | CONTRAINTE | PASCAL |
| 10 | DMAX | DIAMETRE | MM |

# Classification of post-processing tasks

- File conversions (shp, vtk, ...)
- Compute variables
- Mesh transformation

# Classification of post-processing tasks

- File conversions (shp, vtk, ...)
- Compute variables
- Mesh transformation
- Visualization (temporal, longitunal, cross-section, ...)



# Classification of post-processing tasks

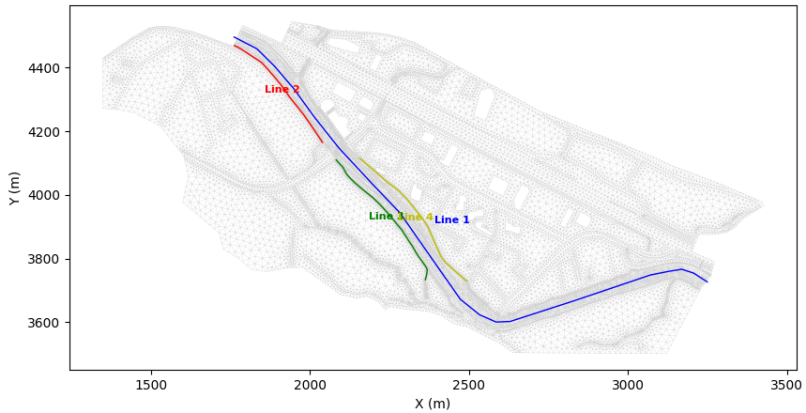
- File conversions (shp, vtk, ...)
- Compute variables
- Mesh transformation
- Visualization (temporal, longitunal, cross-section, ...)
- Temporal analysis

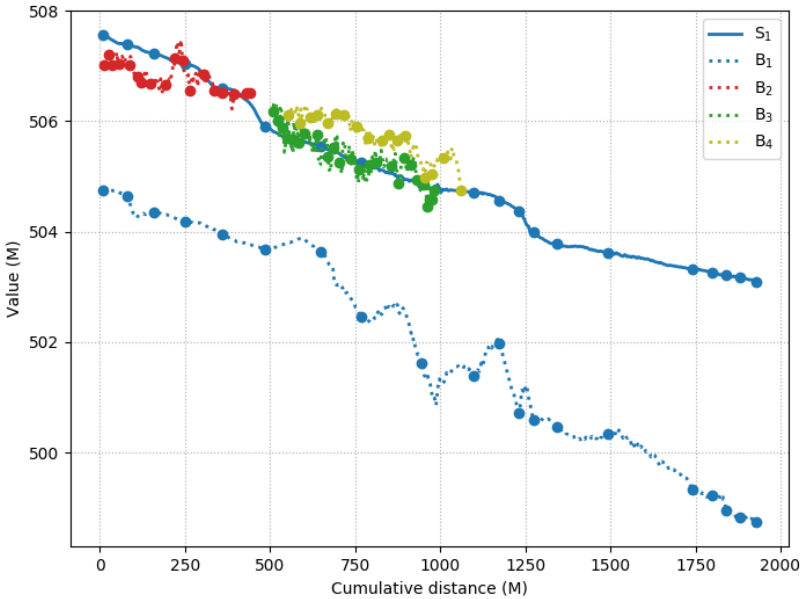
# Classification of post-processing tasks

- File conversions (shp, vtk, ...)
- Compute variables
- Mesh transformation
- Visualization (temporal, longitunal, cross-section, ...)
- Temporal analysis
- Vertical aggregation: depth-averaging, specific layer selection

# Classification of post-processing tasks

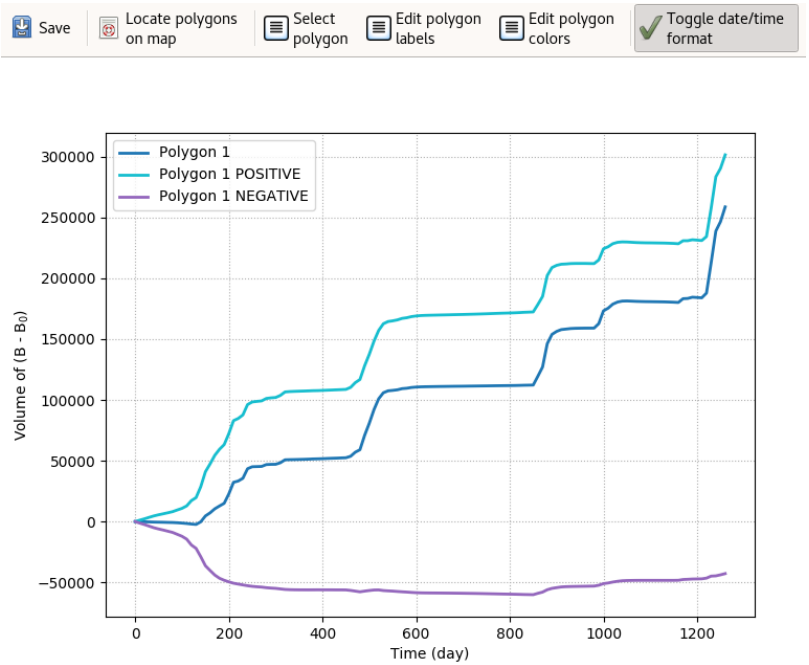
- File conversions (shp, vtk, ...)
- Compute variables
- Mesh transformation
- Visualization (temporal, longitunal, cross-section, ...)
- Temporal analysis
- Vertical aggregation: depth-averaging, specific layer selection
- Interpolate on points, along lines (and project)



# Classification of post-processing tasks

- File conversions (shp, vtk, ...)
- Compute variables
- Mesh transformation
- Visualization (temporal, longitunal, cross-section, ...)
- Temporal analysis
- Vertical aggregation: depth-averaging, specific layer selection
- Interpolate on points, along lines (and project)
- Compute volume, flux



# Classification of post-processing tasks

- File conversions (shp, vtk, ...)
- Compute variables
- Mesh transformation
- Visualization (temporal, longitunal, cross-section, ...)
- Temporal analysis
- Vertical aggregation: depth-averaging, specific layer selection
- Interpolate on points, along lines (and project)
- Compute volume, flux
- Others: comparison criteria (MAD, BSS, ...)

# Philosophy and practice

## Applications

- sensitivity analysis, calibration processes, ...
- write similar post-processing files (slf, CSV)
- export plots, produce maps in mass

# Philosophy and practice

## Applications

- sensitivity analysis, calibration processes, ...
- write similar post-processing files (slf, CSV)
- export plots, produce maps in mass

## Main workflow features

- easy-to-use graphical interface to build pipelines
- visualize progress in real time
- ability to re-use and share pipelines

# Philosophy and practice

## Applications

- sensitivity analysis, calibration processes, ...
- write similar post-processing files (slf, CSV)
- export plots, produce maps in mass

## Main workflow features

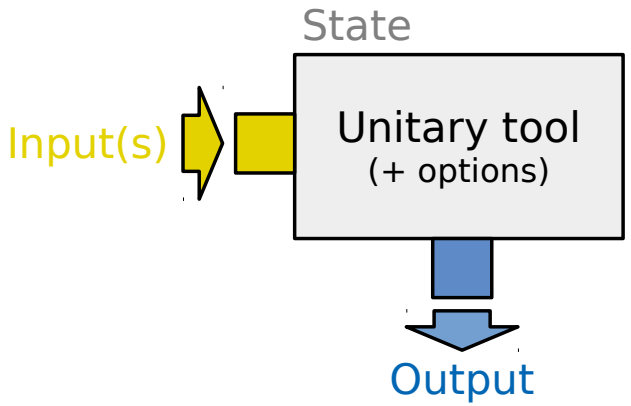
- easy-to-use graphical interface to build pipelines
- visualize progress in real time
- ability to re-use and share pipelines

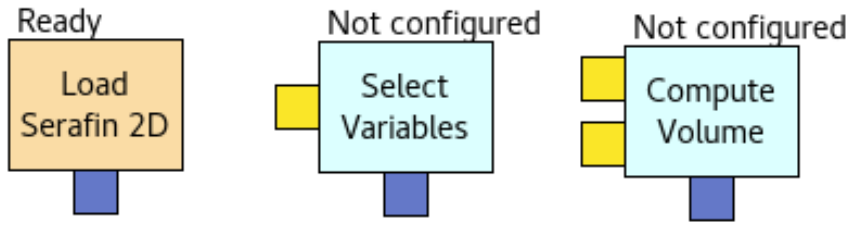
## Two levels of automation

- **Mono**: configure and chain tasks to be run on a single simulation
- **Multi**: run over a set of simulations

# Conceptual model

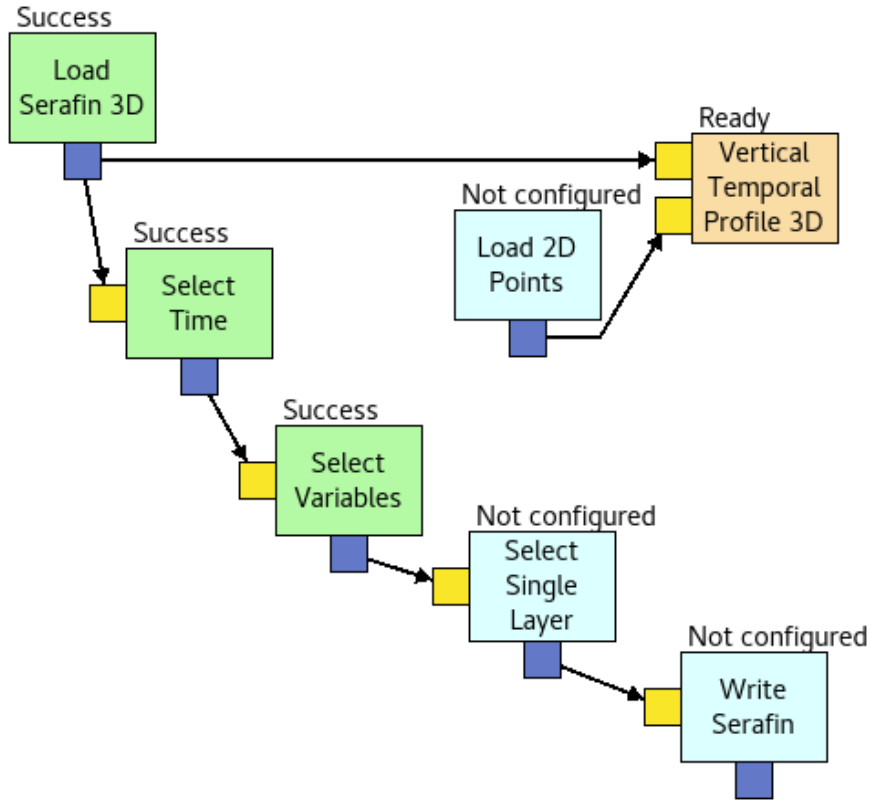
- Each **unitary tool**:
  - State (Not configured, Ready, Sucess)
  - Options (configure)
  - Input(s)/Output



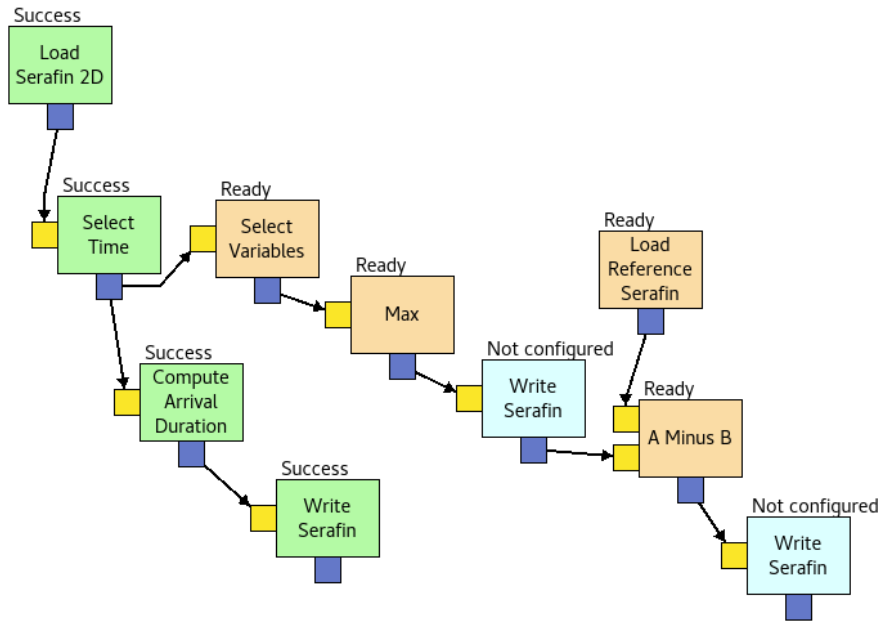
# Conceptual model

- Each **unitary tool**:
  - State (Not configured, Ready, Sucess)
  - Options (configure)
  - Input(s)/Output
- **Combine them** in a **Directed Acyclic Graph** (DAG)
  - data-type correctness
  - junction
  - data flows downward
  - filters
  - commutativity is sometimes possible (flexibility)

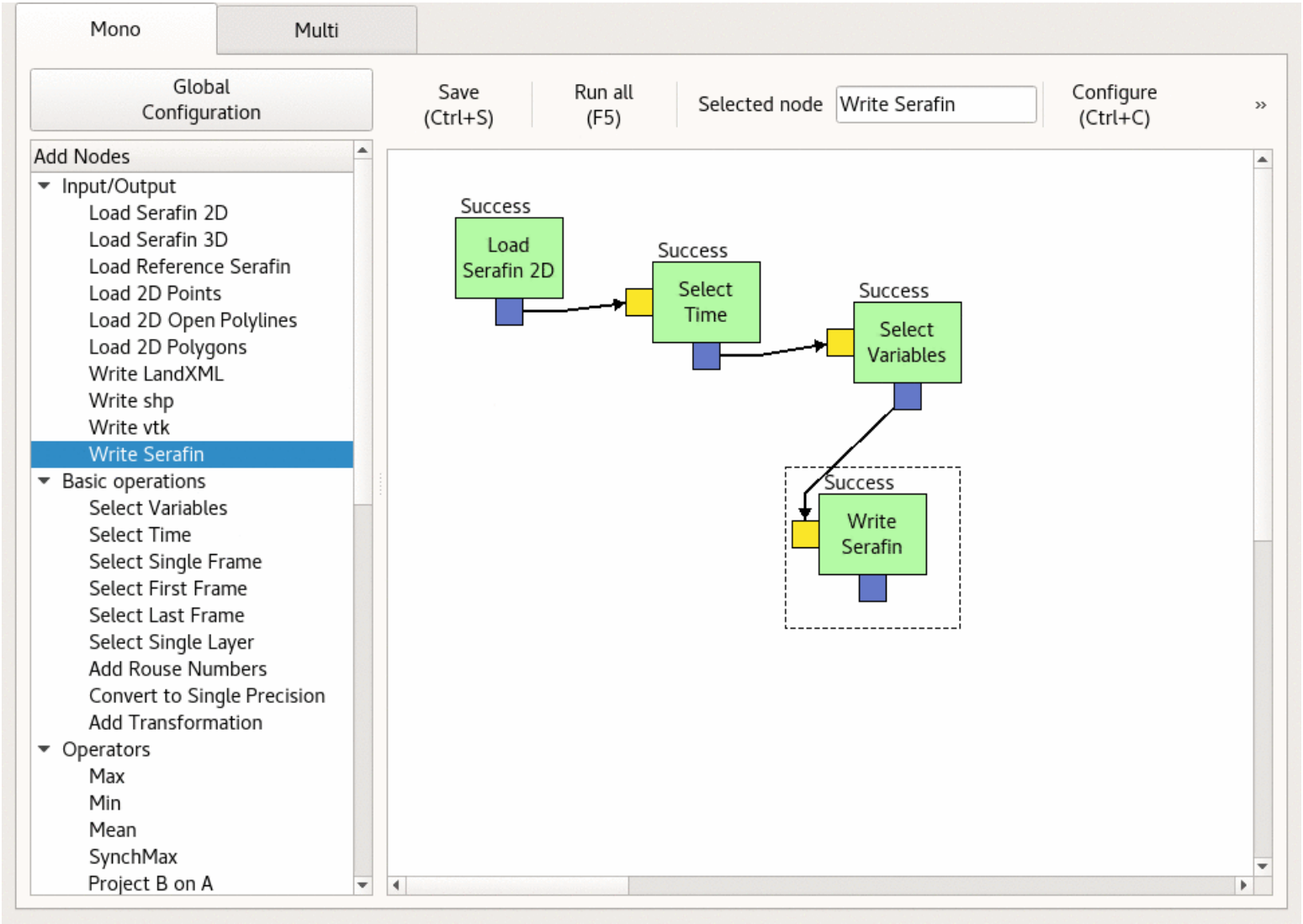


# Conceptual model

- Each **unitary tool**:
  - State (Not configured, Ready, Sucess)
  - Options (configure)
  - Input(s)/Output
- **Combine them** in a **Directed Acyclic Graph** (DAG)
  - data-type correctness
  - junction
  - data flows downward
  - filters
  - commutativity is sometimes possible (flexibility)
  - reproductability (project file)
  - reentrancy (intermediate files)



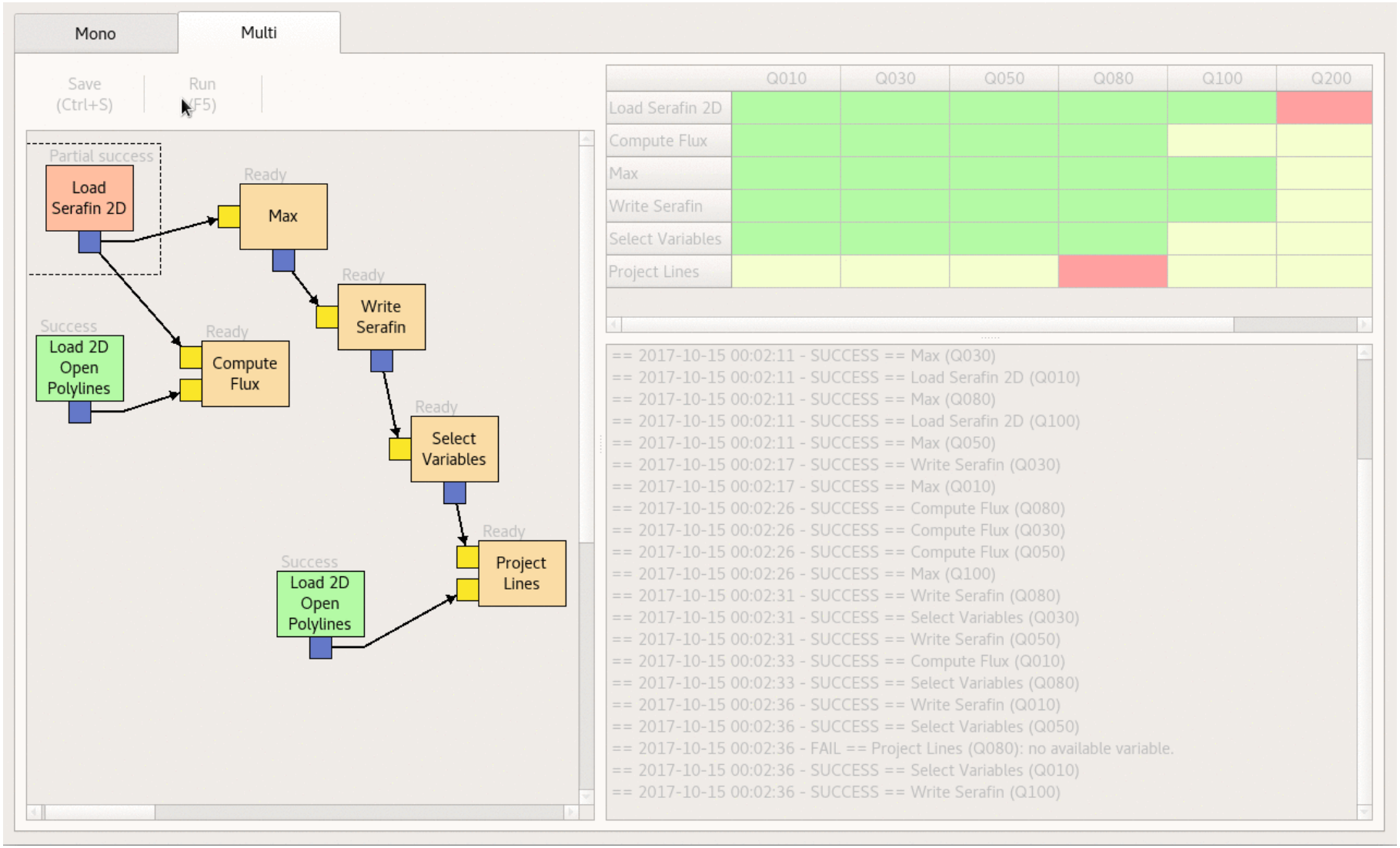
# Demo on Mono view



# Repeat over multiple simulation

- Prerequisites
  - the graph is already configured
  - input data are mostly identical and follows a naming convention
- repeat over this serie of simulations
- parallelize on multiple CPU
- monitoring: status table, log events

# Demo on Multi view



# Conclusion

- Python scripts developped from scratch for Telemac post-processing
- accessible through a GUI
- open-source, modest contribution
- different levels of automation (workflow)
- Some **improvements**:
  - compare to measurements, implementation, ...
  - Unify more Mono and Multi implementation
- Coming **new features** depends on needs (mostly new plots)

# Thank you

```
try:
    PyTelTools.presentation()
    print('Thank your for your attention!')
except QuestionException:
    print('I will do it now!')
print('My work is done')
```

# References

Luc Duron and Yishu Wang.

Equations integrated in PyTelTools.

August 2017.

URL: https://github.com/CNR-Engineering/PyTelTools_media/raw/master/latex/equations.pdf.

Luc Duron and Yishu Wang.

Mathematical definitions for PyTelTools.

August 2017.

URL: https://github.com/CNR-Engineering/PyTelTools_media/raw/master/latex/mathematical_definitions.pdf.

Yishu Wang.

PyTelTools: workflow for water flow, August 2017.

CNR ENGINEERING