

# COMMON WORKFLOW DESCRIPTION AN EXPERIMENTAL VIEWPOINT

S. Brinckmann, F. Chen, T. Dahmen, T. Hickel, L. Huber, S. Menon,  
J. Murugan, J. Neugebauer, R. Schwaiger, H. Tsybenko, O. Waseda

IMD-1: Microstructure and Properties of Materials

# ALLOW FOR COMPARISON AND REUSE OF WORKFLOWS

## Experiments 1

1. Prepare Al sample
2. Measure sample
3. Tensile test
4. Analyse curves
5. Calculate Modulus

# ALLOW FOR COMPARISON AND REUSE OF WORKFLOWS

## Experiments 1

1. Prepare Al sample
2. Measure sample
3. Tensile test
4. Analyse curves
5. Calculate Modulus

## Experiments 2

1. Prepare polymer
2. Measure sample
3. Tensile test
4. Analyse curves
5. Calculate Modulus

# ALLOW FOR COMPARISON AND REUSE OF WORKFLOWS

## Experiments 1

1. Prepare Al sample
2. Measure sample
3. Tensile test
4. Analyse curves
5. Calculate Modulus

## Experiments 2

1. Prepare polymer
2. Measure sample
3. Tensile test
4. Analyse curves
5. Calculate Modulus

## Simulations

1. Build model
2. Apply forces
3. Calculate
4. Analyse curves
5. Calculate Modulus

# MANY WORKFLOW LANGUAGES EXIST

Dedicated workflow languages

- Common workflow language
- Snakemake
- Nextflow
- Nexus
- Kadi4Mat - json

...

Programming languages:

- C
- python
- bash
- ...

**TASK: PICK ONE AND USE FOR  
SIMULATIONS AND EXPERIMENTS**

# FORMAL WORKFLOW DESCRIPTION EXISTS

## BUSINESS PROCESS MODELING

- Visual representation
- Process analysis
- Communication and documentation
- Simulation and automation



Two layers of information:

- Top level: short description
- Bottom level: details

# FORMAL WORKFLOW DESCRIPTION EXISTS

## BUSINESS PROCESS MODELING

- Visual representation
- Process analysis
- Communication and documentation
- Simulation and automation

# GOAL: WORKFLOW STANDARDIZATION

Experiments have two layers:

- Workflow step incl. description
- Standard Operating Procedure

Simulations have two layers:

- Function calls
- Each function is a graph node



# WORKFLOW EXAMPLE

```
from common_workflow_description import Storage, Sample, step
from analysis_steps import plot_curves, calc_E
```

Header

```
wf = Workflow('Sandia Fracture Challenge 3')
proceduresLibrary = urlparse('https://...')
storage=Storage(proceduresLibrary)
```

Define workflow

```
sample = Sample('AM_NA_05')
wf.step1 = step(storage, sample, 'metallography', {})
wf.step2 = step(storage, sample, 'light microscopy', {})
```

Experiments

```
...
file_name = [v for k,v in list(wf....items())...][0][1]
wf.step6 = plot_curves(file_name, 'Strain (Gauge0)', 'Engr. Stress')
wf.step7 = calc_E(file_name, 'Strain (Gauge0)', 'Engr. Stress')
```

Analysis

```
print('Output: ', '\n '.join([f"{k}: {v}" for k,v in list(wf...)]))
wf.draw().render(filename="io_demo", format="png", cleanup=True,
```

Footer

same as pyiron-workflow style

Member of the Helmholtz Association

# PROCEDURE IN MARKDOWN+

```
# Zeiss SEM  
- Vent  
- Open chamber  
- Mount sample and close chamber  
- Use it at |voltage|20| kV  
- Use a aperture of |aperture|3|
```

```
Author: Steffen Brinckmann, IMD-1, FZJ
```

```
<!--- version:1.0 |filename|| -->
```

# EXAMPLE: INSTRUCTIONS

Instructions can have:

- no forms
- only ask for a file

Common workflow description

Execute the following action with sample: AM

**Metallography**

```
## Grinding
• rounding all corners
• semi auto grinding: 500 -> 800 -> 1000 -> 2400 (30s) -> 4
  - significant pressure, two directions, last paper 3-4 dir
  - starting at #1000: rpm = 100
  - cleaning after each step (sample: with much water >> eth
  - Check the scratch which was made at the previous grindin

## Auto polishing
• mount sample on sample holder for auto polisher
• cleaning all clothes & polishing sample holder with water
• 3 um diamond suspension (1 spray every 1 min)
  - 30N (force reduction) / 10min 2 step (total time: 20 min

• 1 um diamond suspension (1 spray every 1 min)
  - 1st step: 15N (force reduction) / 10min / 150 rpm
  - 2nd step: 20N (force reduction) / 5min / 150 rpm
  - blue lubricant : stage 8

• cleaning after each step
  - sample: rinsing with ethanol, ultra sonic cleaning, rins
  - holder: cleaning with water

• check the surface in OM. If the surface is not good enough

## Final polishing
```

done

Common workflow description

Execute the following action with sample: AM

**Confocal Image Acquisition with the Olympus**

- Turn on the PC
- Turn on the LEXT OLS4000 microscope
- Start the LEXT OLS4000 software
- Select the "Imaging" mode
- Select the objective lens with the smallest setting of 5
- Press the button "Move to load position" in the "Map scre
- The objective lens will be moved upwards to avoid contact
- Place the sample on the center of the stage
- In the "Map screen" window, press the button "Move to ori
- The sample will be moved underneath the microscope under
- Move the focus knob counterclockwise to adjust the hight
- Lock the focus knob to avoid further movement
- Set the lower height limit by clicking the "Advanced sett
- Ensure that the "Imaging" mode and the "Laser Microscope"
- Search for the area of interest on the sample surface by
- Increase the magnification step-wise by clicking on the o
- Select the desired objective lens magnification from the
- After changing the objective lens, adjust height to reach
- Click on the "Laser" button. The mode will change from th
- Only the in-focus sections will be visible in the "Laser
- Click the "Focus" tab to access the focus controls
- Click the arrow buttons to raise or lower the objective l
- Set the origin height by clicking the "Advanced settings"
- Click the "Brightness" tab to change the brightness setti
- Use the "Brightness" slider or click the "Auto" button to
- Adjust the brightness settings to be as high as possible

filename

select file

done

# INSTRUCTION OPTIONS

Instructions can include parameters

- geometry

- real process parameter

prescribed flow rate: 16

real flow rate: 10

- experimental metadata that is not in output file

- Setup controll box at instrument
  - General information
  - buttons F1..F3 correspond to three symbols above
  - PC-mode: use for control by PC
  - turn knob to get there and then press ESC to get move th
- Start Software and hope that LAN connection works
  - otherwise retry
- Button Bike "Travel order"
  - distance-controlled, relative: 100mm with 100mm/min
  - direction clear
- Measure the sample:
  - |width||
  - |thickness||
- Mount sample: top first and then move into bottom
- Button Setup experiment: Hammer button
  - "Test procedure, speeds", "End of test" most important
  - Switch from F0->F1 confusing
  - choose F0 as max force
  - Example speeds:
    - metals=0.5mm/min
    - polymers=5mm/min
  - Adopt path: "Store data to"
- all 3 tare-buttons should be pressed
- Button Traffic-Light: start experiments
- Button Diagram (one line)
  - open & close: rescale graph

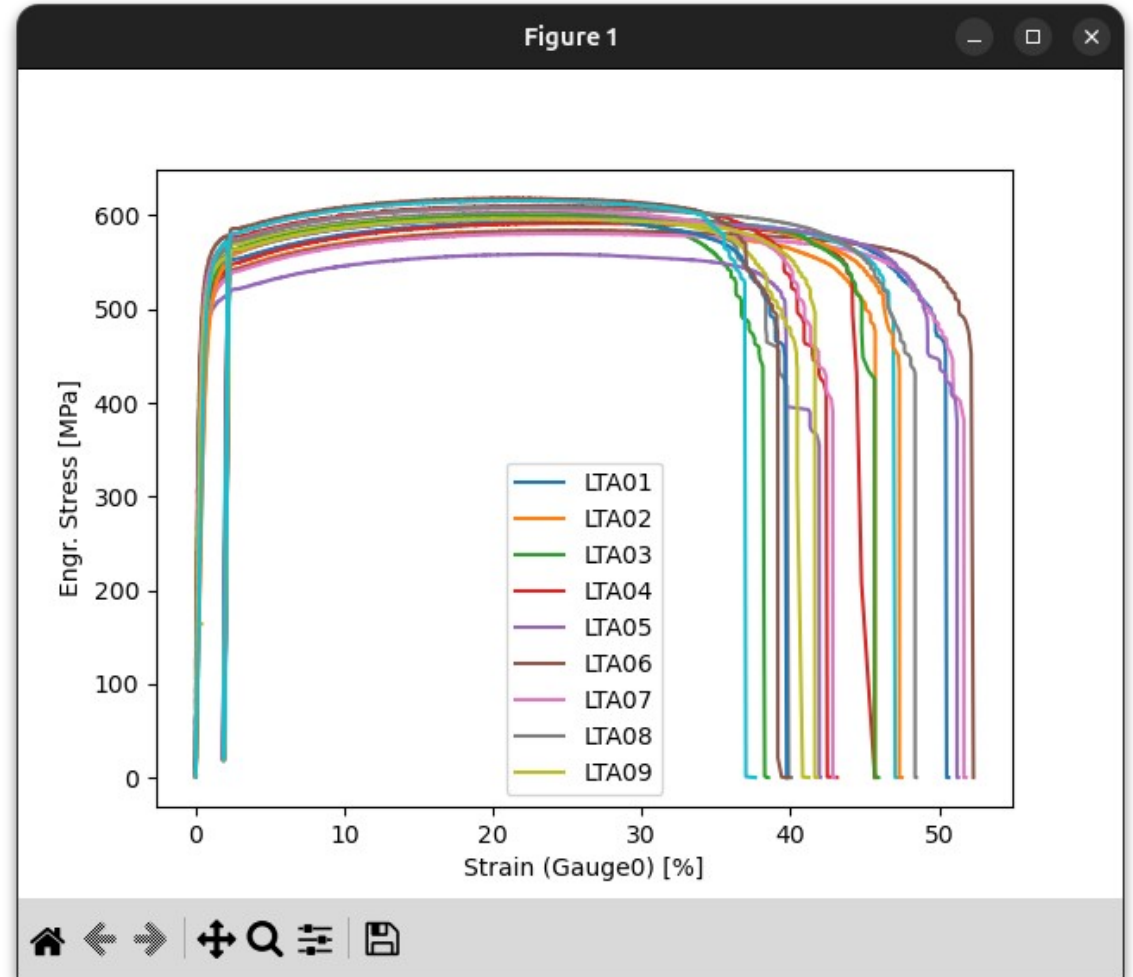
width	<input type="text" value="1.016"/>
thickness	<input type="text" value="1.027"/>
filename	<input type="button" value="select file"/>
	<input type="button" value="done"/>

# EXAMPLE: ANALYSIS STEP

All possibilities exist for:

- graphical plotting (tensile curves)
- calculating material properties
- simulation steps

Young's modulus in GPa:  
average=159.82  
std-dev=9.05



# OUTPUT AND LOGFILE

Started minimalistic workflow engine

Output:

```
step1: [Name: AM05, '', {}]  
step2: [Name: AM05, '', {'magnification': '5'}]  
step3: [Name: AM05, '....xlsx', {'width': '1.016'}]  
step4: [Name: AM05, '', {'magnification': '5'}]  
step5a: [Name: AM05, '', {'voltage': '30', 'aperture': '3'}]  
step5b: [Name: AM05, '', {'voltage': '30', 'aperture': '2'}]
```

```
08-22 10:52:39|INFO:Start workflow  
08-22 10:56:31|INFO:Start step  
    sample:{AM_NA_05}  
    procedure-name:{tensile test}  
    sha256:{ae9c351e9fcdbd41fc39d0e3831d9dc8d3388bd7e714869aa050b870dc21ac341}  
    Parameters: {  
08-22 10:57:40|INFO:Save step  
    file-name:...LongitudinalTensileOverall.xlsx  
    metadata: {"width": "1.016"}  
08-22 10:57:40|INFO:End step
```

# TWO ENGINES EXIST TO EXECUTE WORKFLOWS

Pyiron-workflow:

- Many features: graphics, super-computer...
- Allow for adoptive of workflows

**Same output and  
logging files**

Minimalistic engine:

- Can only run common-workflow-description
- No installation required

# A COMMON WORKFLOW DESCRIPTION EXISTS FOR EXPERIMENTAL AND NUMERICAL MATERIALS SCIENCE

- Most basic version exists
  - creates a receipt of all experimental steps
  - unified way to write procedures
- You can use pyiron-workflow to adopt it
- **GUI version with drag-drop will exist**

**Looking for scientists feedback: What is missing?**





# NON UNIQUE DEFINITION: WHAT IS “A STEP” (same in simulations)

Smallest step that make scientific sense to execute individually  
(might depend on others)

- SEM
- EBSD (requirement of SEM before)
- EDX (requirement of SEM before)

Issue: requirements can only be resolved if there is step types (SEM procedures, ...) → much more complicated (second version)

# KNOWN SHORTCOMING

Validation of experimental input:

“What happens if I enter a non-nonsensical value?”

Issue: How to integrate that into procedure that it is easy to write?

- allowed values
- only integers can be entered
- regex needed?

# NOT A SHORTCOMING

Allow for comment / remark in each step?

- one can enter unforeseeable events

Very easy fix: just add “|comment|” at end of file