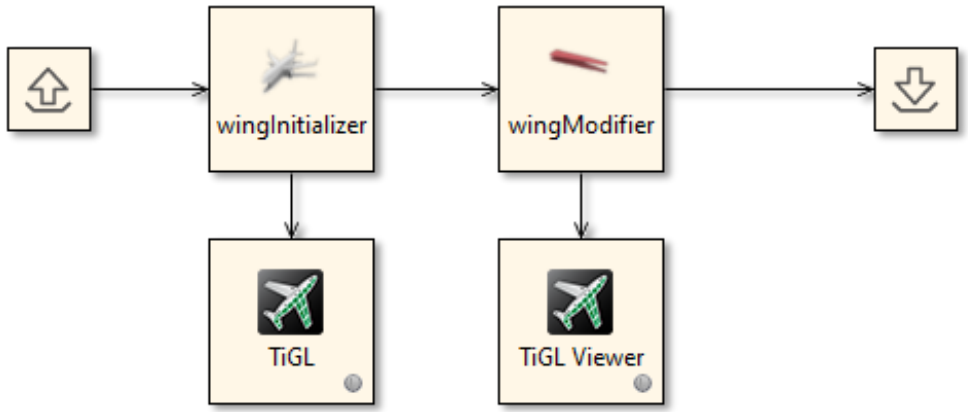
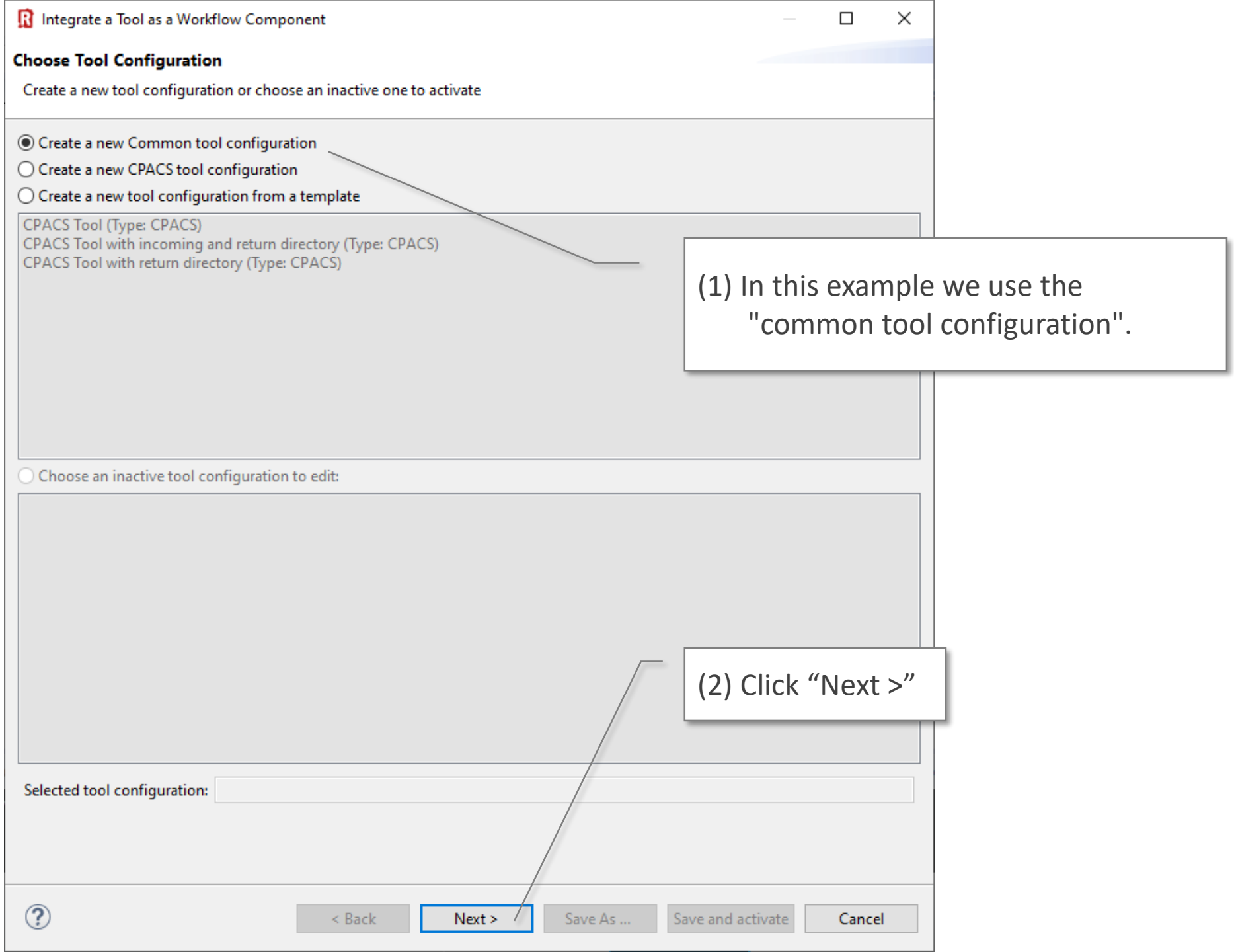
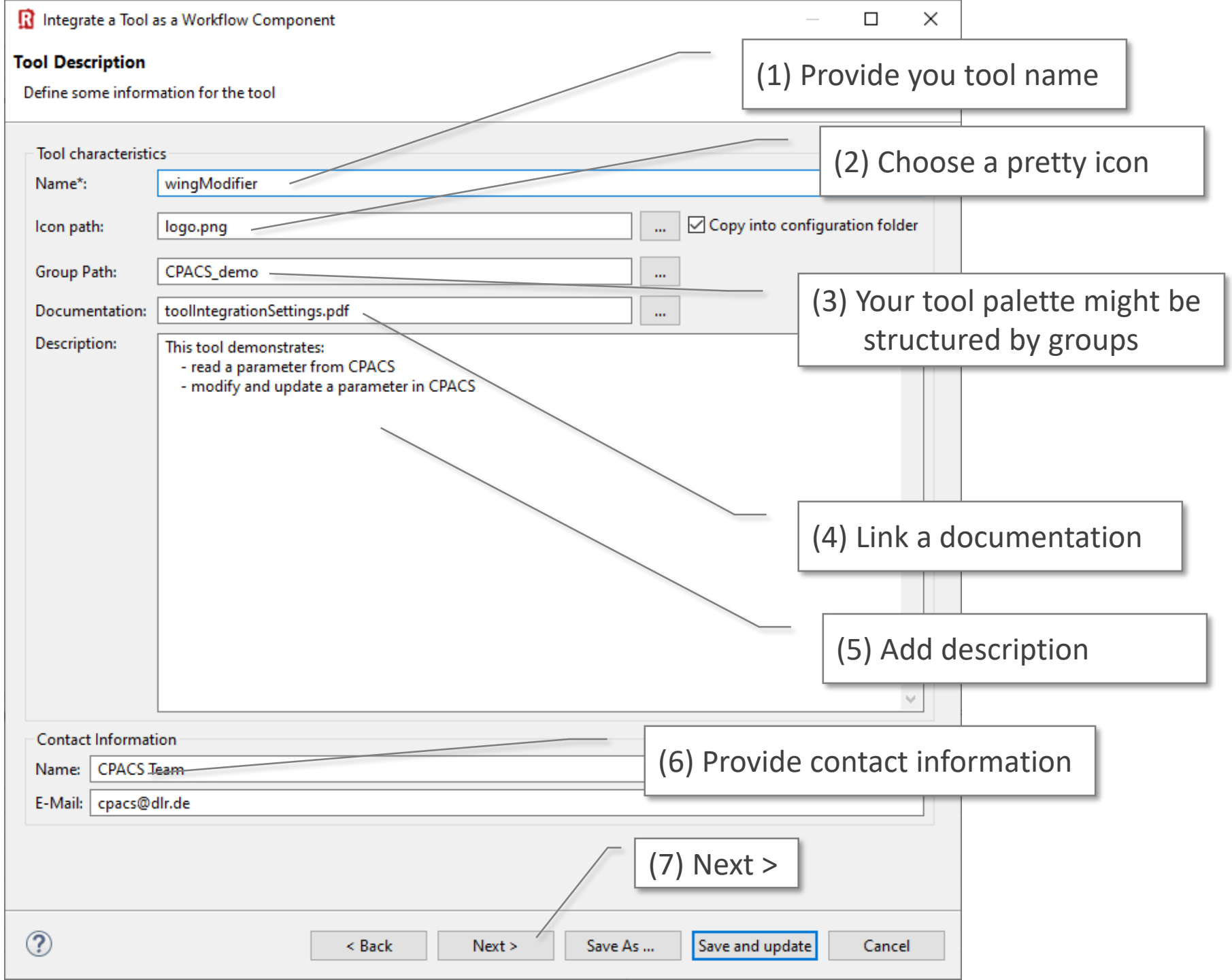
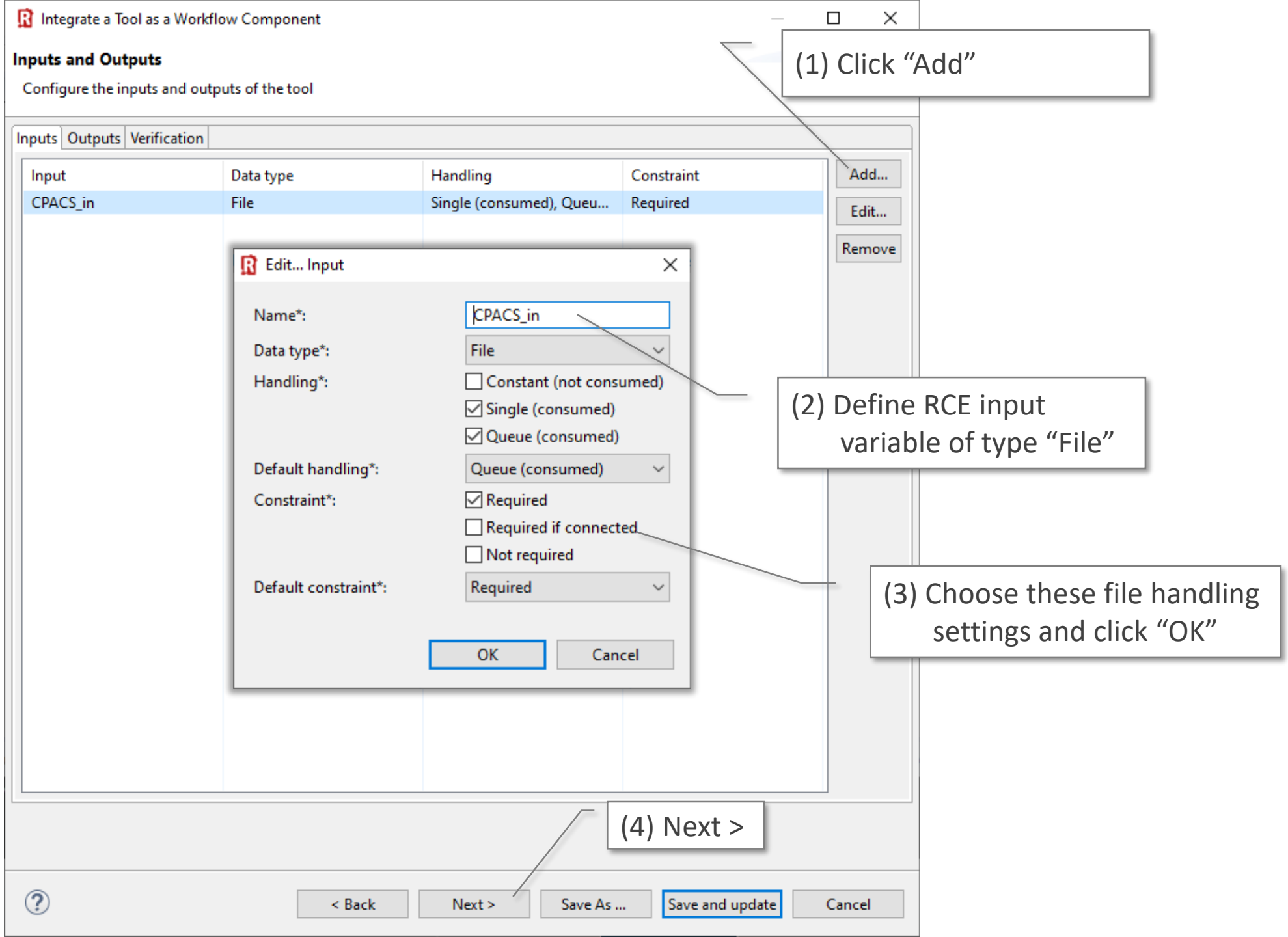
RCE tool integration

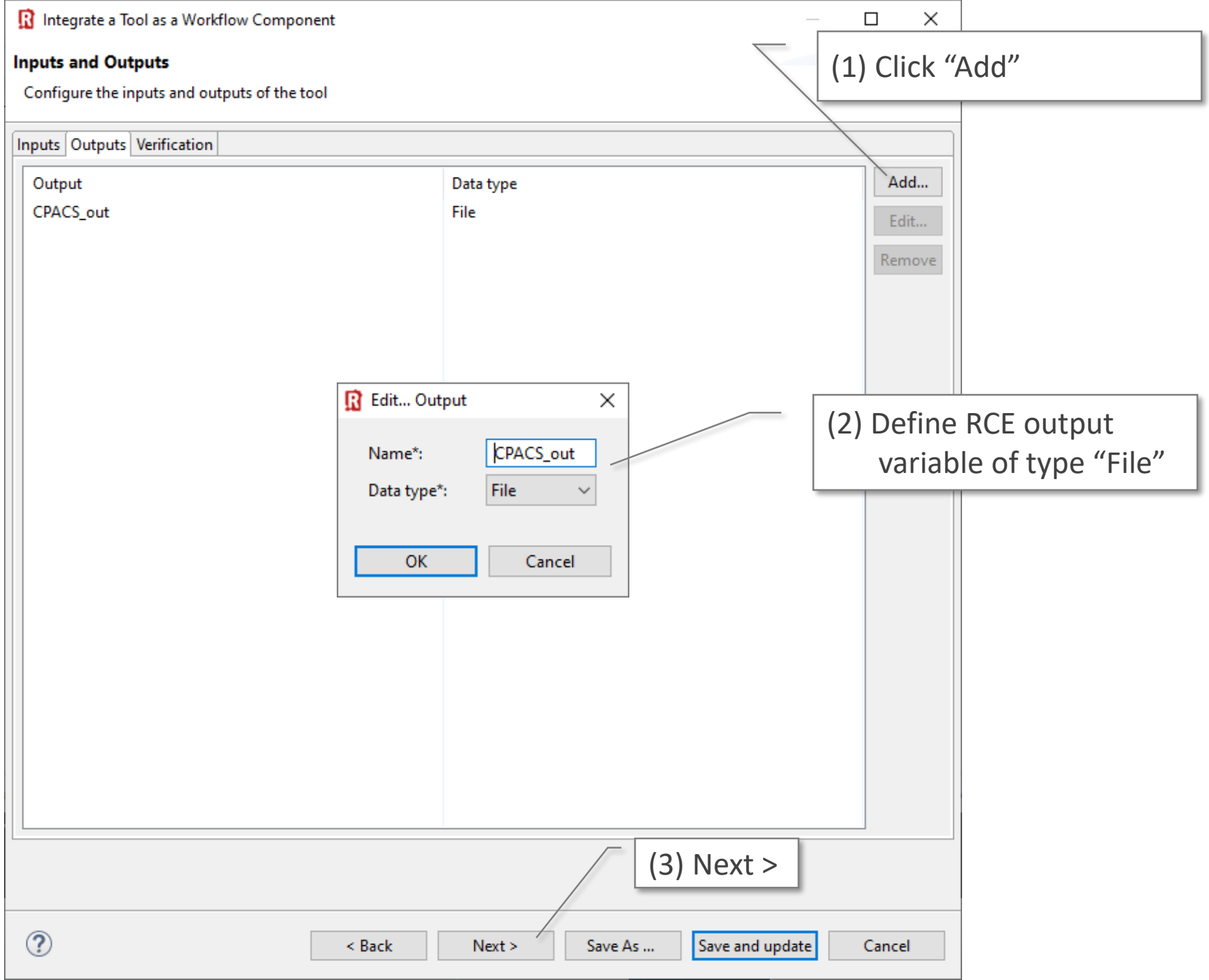
wingModifier

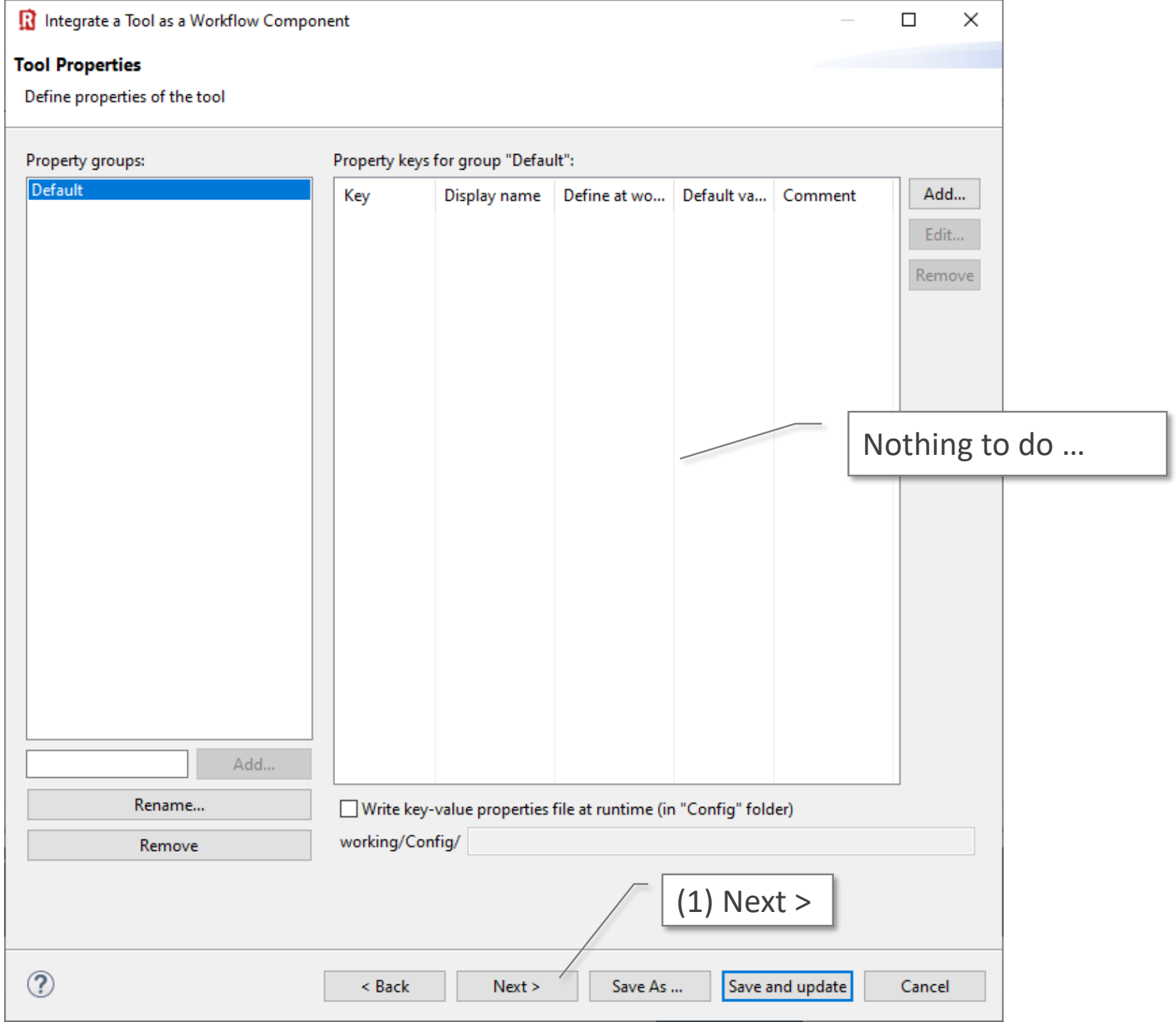


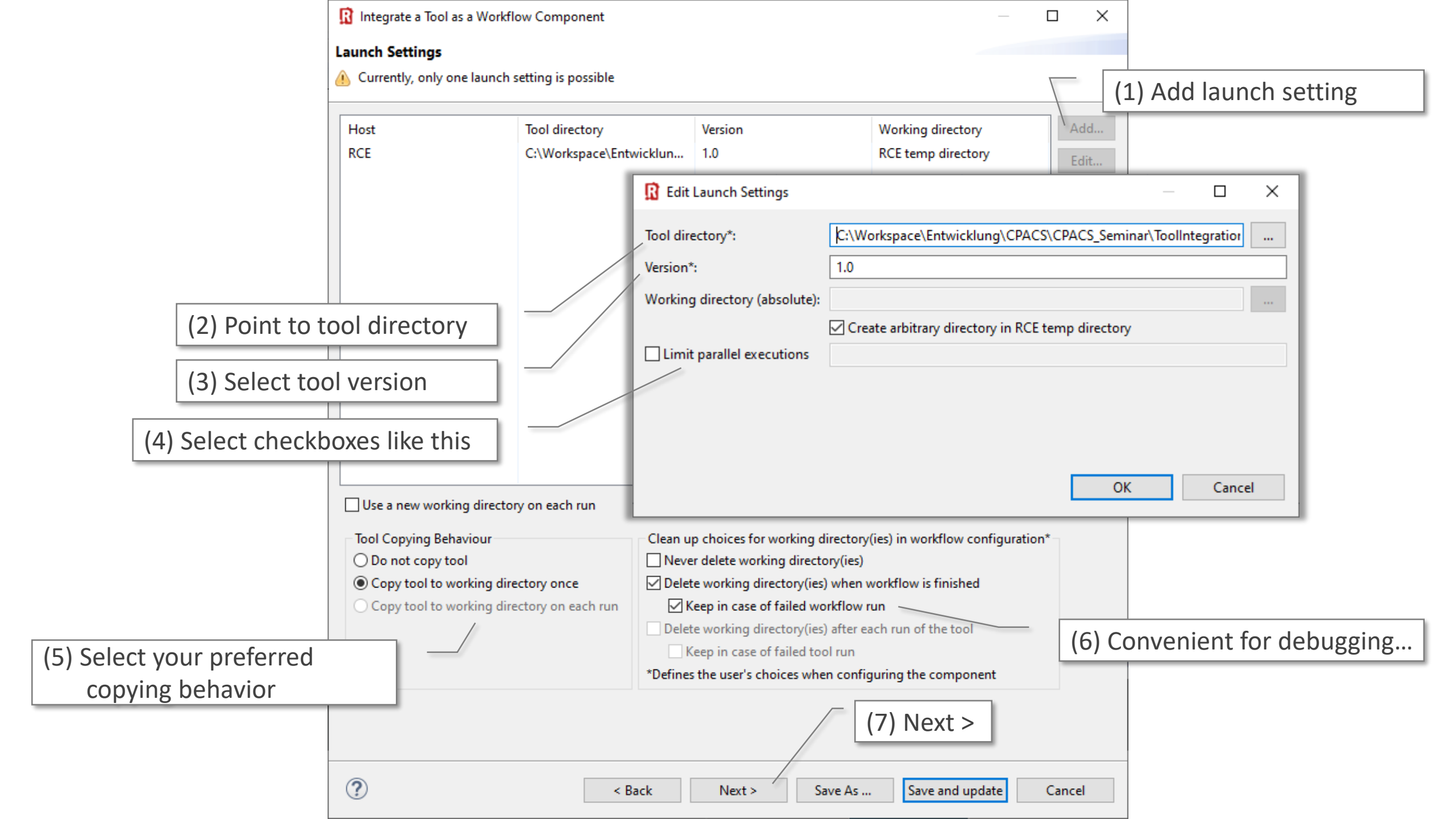

Integrate a Tool as a Workflow Component
Choose Tool Configuration
Create a new tool configuration or choose an inactive one to activate
Create a new Common tool configuration
Create a new CPACS tool configuration
Create a new tool configuration from a template
CPACS Tool (Type: CPACS)
CPACS Tool with incoming and return directory (Type: CPACS)
CPACS Tool with return directory (Type: CPACS)
Choose an inactive tool configuration to edit:
Selected tool configuration:
< Back
Next >
Save As ...
Save and activate
Cancel
(1) In this example we use the "common tool configuration".
(2) Click “Next >”


Integrate a Tool as a Workflow Component
Tool Description
Define some information for the tool
Tool characteristics
Name*:
wingModifier
Icon path:
logo.png
...
Copy into configuration folder
Group Path:
CPACS_demo
...
Documentation:
toolIntegrationSettings.pdf
...
Description:
This tool demonstrates:
- read a parameter from CPACS
- modify and update a parameter in CPACS
Contact Information
Name:
CPACS Team
E-Mail:
cpacs@dlr.de
< Back
Next >
Save As ...
Save and update
Cancel
(1) Provide you tool name
(2) Choose a pretty icon
(3) Your tool palette might be structured by groups
(4) Link a documentation
(5) Add description
(6) Provide contact information
(7) Next >


Integrate a Tool as a Workflow Component
Inputs and Outputs
Configure the inputs and outputs of the tool
Inputs
Outputs
Verification
Input
Data type
Handling
Constraint
CPACS_in
File
Single (consumed), Queu...
Required
Add...
Edit...
Remove
Edit... Input
Name*:
CPACS_in
Data type*:
File
Handling*:
Constant (not consumed)
Single (consumed)
Queue (consumed)
Default handling*:
Queue (consumed)
Constraint*:
Required
Required if connected
Not required
Default constraint*:
Required
OK
Cancel
(1) Click “Add”
(2) Define RCE input variable of type “File”
(3) Choose these file handling settings and click “OK”
(4) Next >
< Back
Next >
Save As ...
Save and update
Cancel


Integrate a Tool as a Workflow Component
Inputs and Outputs
Configure the inputs and outputs of the tool
Inputs
Outputs
Verification
Output
Data type
CPACS_out
File
Add...
Edit...
Remove
Edit... Output
Name*:
CPACS_out
Data type*:
File
OK
Cancel
(1) Click “Add”
(2) Define RCE output variable of type “File”
(3) Next >
< Back
Next >
Save As ...
Save and update
Cancel


Integrate a Tool as a Workflow Component
Tool Properties
Define properties of the tool
Property groups:
Default
Add...
Rename...
Remove
Property keys for group "Default":
Key
Display name
Define at wo...
Default va...
Comment
Add...
Edit...
Remove
Write key-value properties file at runtime (in "Config" folder)
working/Config/
Nothing to do ...
(1) Next >
< Back
Next >
Save As ...
Save and update
Cancel

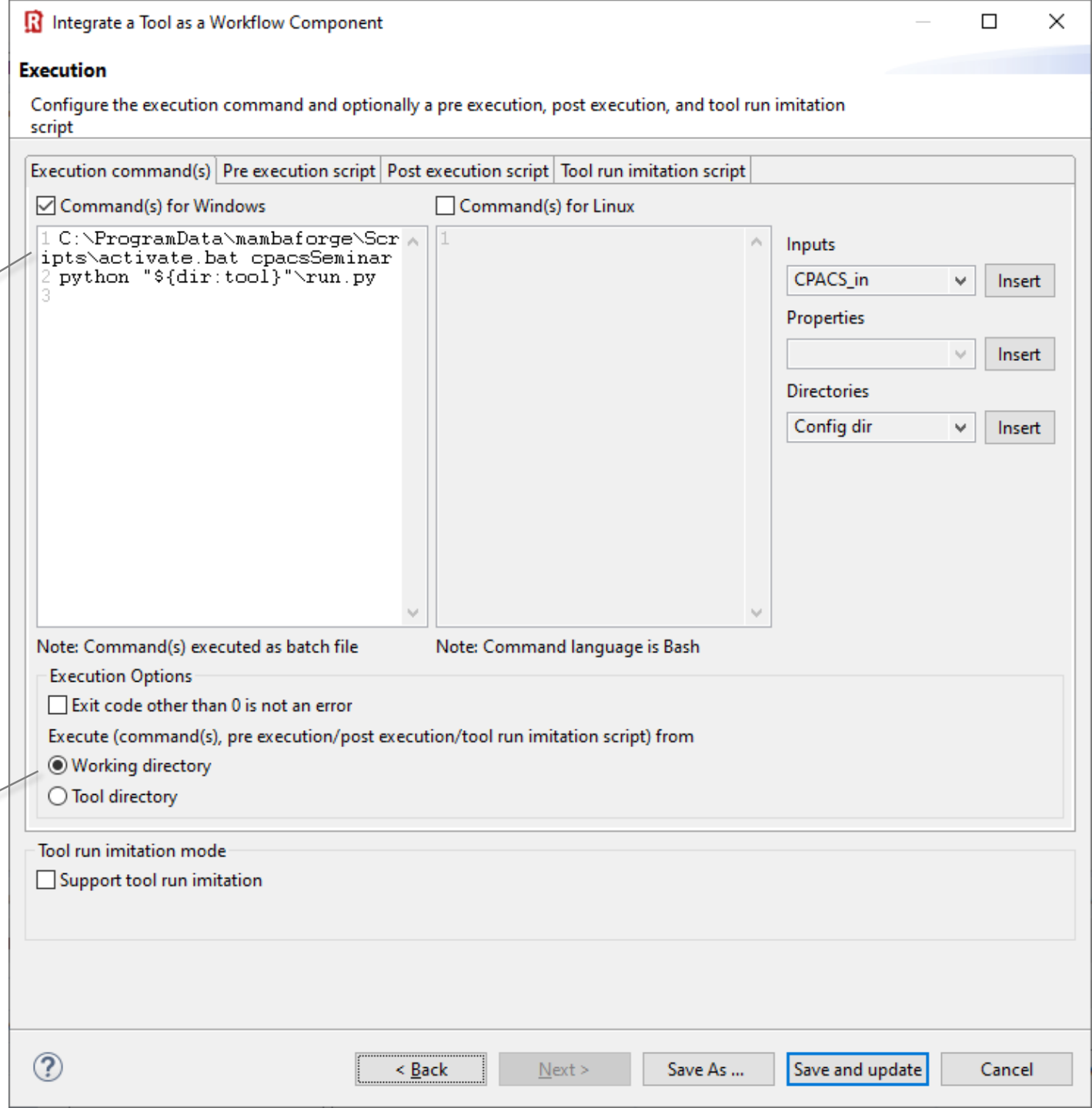

Integrate a Tool as a Workflow Component
Launch Settings
Currently, only one launch setting is possible
Host
Tool directory
Version
Working directory
RCE
C:\Workspace\Entwicklun...
1.0
RCE temp directory
Add...
Edit...
Edit Launch Settings
Tool directory*:
C:\Workspace\Entwicklung\CPACS\CPACS_Seminar\ToolIntegratior
Version*:
1.0
Working directory (absolute):
Create arbitrary directory in RCE temp directory
Limit parallel executions
OK
Cancel
Use a new working directory on each run
Tool Copying Behaviour
Do not copy tool
Copy tool to working directory once
Copy tool to working directory on each run
Clean up choices for working directory(ies) in workflow configuration*
Never delete working directory(ies)
Delete working directory(ies) when workflow is finished
Keep in case of failed workflow run
Delete working directory(ies) after each run of the tool
Keep in case of failed tool run
*Defines the user's choices when configuring the component
< Back
Next >
Save As ...
Save and update
Cancel
(1) Add launch setting
(2) Point to tool directory
(3) Select tool version
(4) Select checkboxes like this
(5) Select your preferred copying behavior
(6) Convenient for debugging...
(7) Next >

```
C:\ProgramData\mambaforge\Scripts\activate.bat
cpacsSeminar
python "${dir:tool}"\run.py
```

(1) Specify execution commands for Windows and/or Linux

ⓘ This example is written in Python, so we need to activate the correct interpreter first.

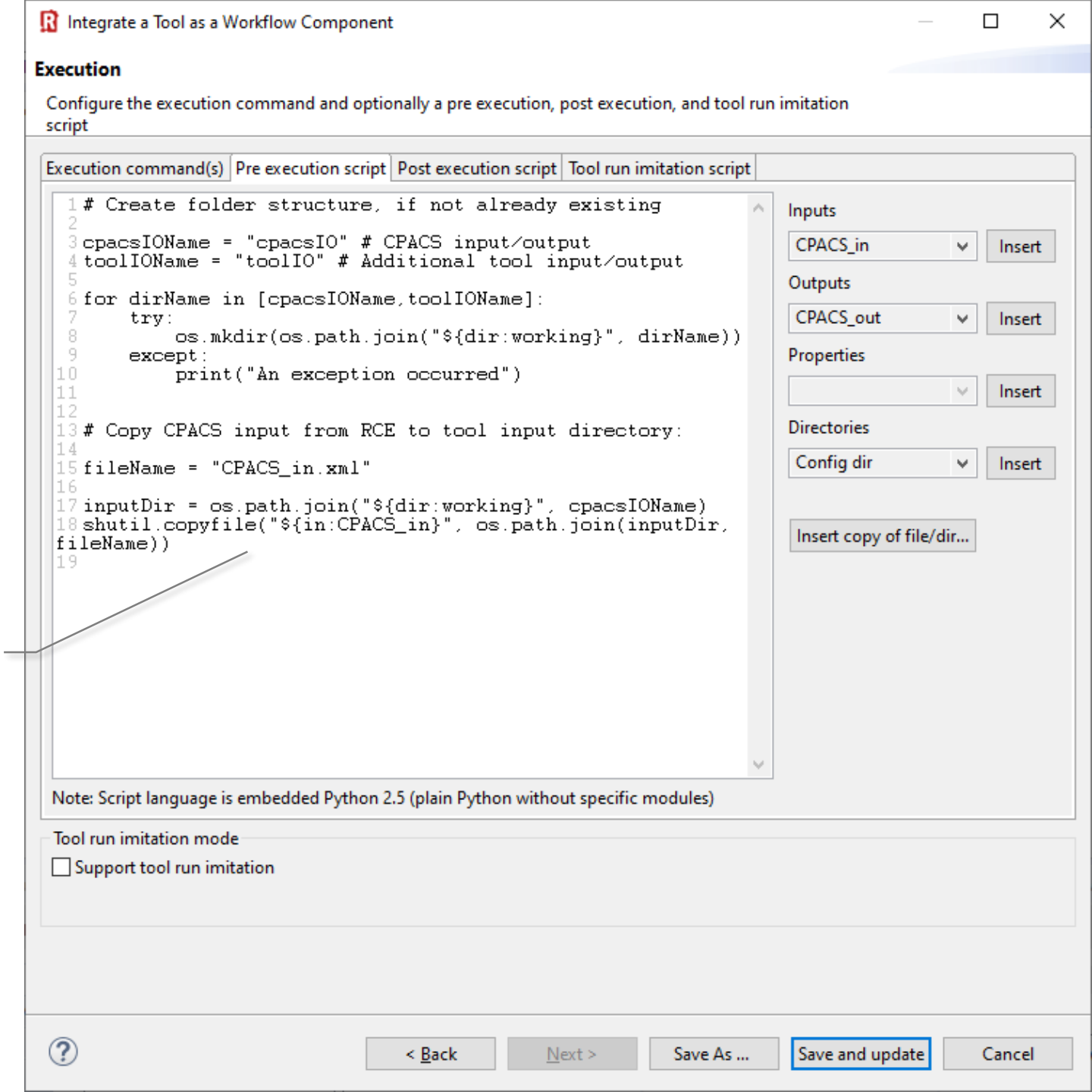
(2) Select “Working directory”



```
# Create folder structure, if not already existing

inputDirName = "ToolInput"
outputDirName = "ToolOutput"

for dirName in [inputDirName,outputDirName]:
    try:
        os.mkdir(os.path.join("${dir:working}", dirName))
    except:
        print("An exception occurred")


# Copy CPACS input from RCE to tool input directory:

fileName = "CPACS_in.xml"

inputDir = os.path.join("${dir:working}", inputDirName)
shutil.copyfile("${in:CPACS input}", os.path.join(inputDir,
fileName))
```

(1) Insert pre-execution script in Python

ⓘ Pre-execution: What happens before the actual tool is activated. Use this to create the required folder structure, if necessary, and copy the input file into it.



```
dirName = "ToolOutput"
fileName = "CPACS_out.xml"

${out:CPACS output} = os.path.join("${dir:working}",
dirName, fileName)
```

ⓘ Post-execution: What happens after the actual tool has finished. Here we're telling RCE where to find the output file.

(1) Insert post-execution script in Python