A Quantitative Study on the Re-executability of Publicly Shared Scientific Workflows

Rudolf Mayer, Andreas Rauber

SBA Research, Austria

secure
sba-research.org

TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

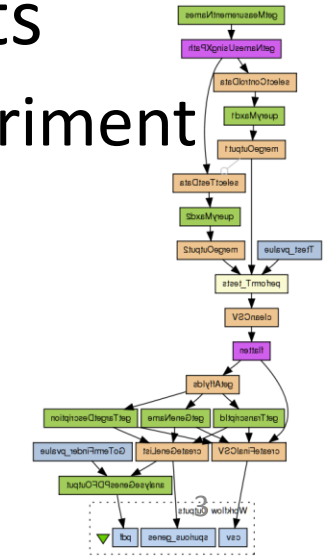
Outline

- Introduction & Motivation
- Data Set & Experiment Setup
- Workflow processors & characteristics
- Results & Recommendations
- Conclusions

Introduction & Motivation

- Repeatibility & Reproducibility cornerstones
  - Computational/eScience shouldn‘t be an exception ...

- Workflows have become a popular mean to share and publish scientific experiments
  - Describe & formalise the steps of an experiment
  - Bundle scripts & code with the workflow definition
  - (More) platform independent
  - Facilitate & *should* enable repeatibility



Introduction & Motivation

- Are workflows really easily repeatable?
  - At least ***re-executable***?
- Prior studies on re-execution / reproducibility
  - In-depth analysis, but small in size: a handful of example workflows
- Need for a larger analysis to see patterns

Introduction & Motivation

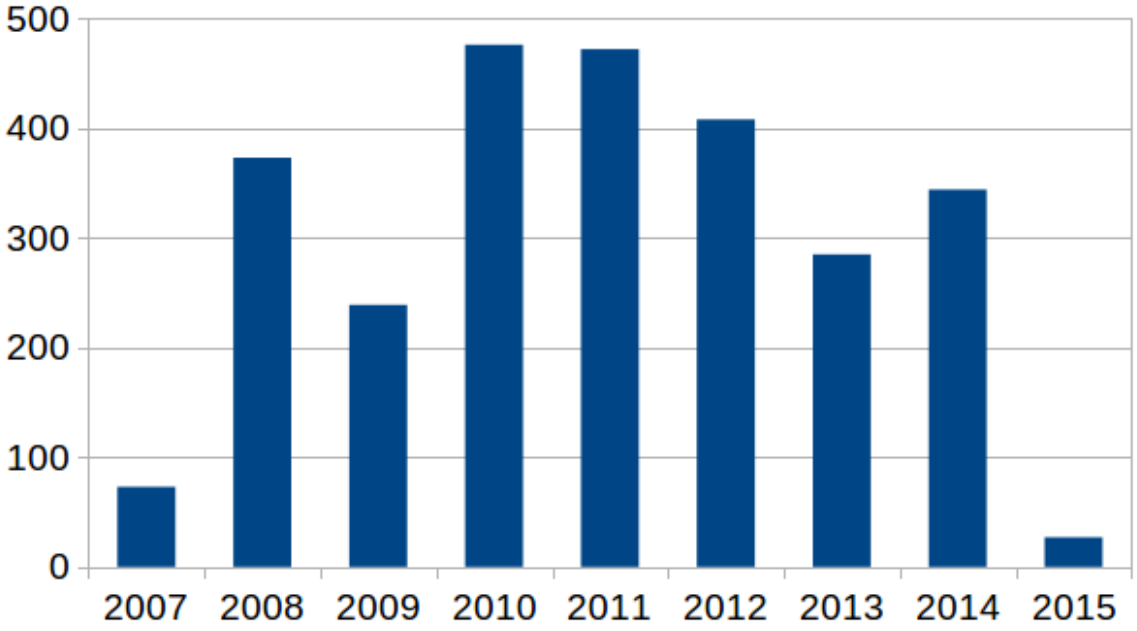
- Our study: large scale quantitative analysis
- Obtain workflows from a public platform dedicated to sharing scientific work myexperiment
  - Published by authors ➔ should be „better quality“
- Try to re-execute the workflows
  - Record data on the reasons for failure along
- Analyse the most common reasons for failures
- Recommendations for enabling better re-executability

Outline

- Introduction & Motivation
- Data Set & Experiment Setup
- Workflow processors & characteristics
- Results & Recommendations
- Conclusions

Data Set

- Source: myExperiment.com (as of March 2015)
  - Available since 2007
  - Data often published by original researchers
  - 300-400 workflows uploaded per year



Data Set

- Obtained workflow definition & meta-data
  - using myExperiment REST API

- As of March 2015:
  - ~ 2,700 workflows shared
  - Small number (92) is *private*
  - 40 different workflow engines
  - Majority for Taverna 2 engine

Workflow Engine	%
Taverna 2	54.7
Taverna 1	20.9
RapidMiner	10.0
Galaxy	2.0
Others	12.4

Experiment Setup

- Focus on Taverna 2 workflows
  - Account for 55% of the data set
  - Taverna API to analyse & run in batch mode
  - Final data set: ***1,443 workflows***

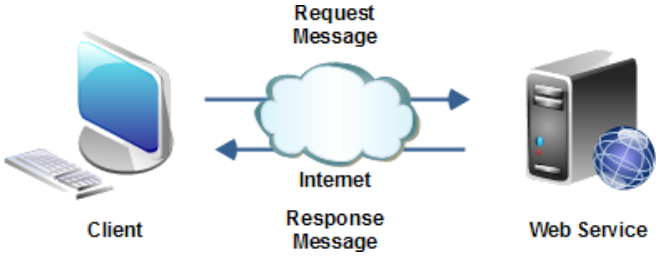
- Static analysis: collect information on types of workflow processing elements
- Re-execution: automatically execute workflows
  - Utilise example values for workflow input parameters
  - Collect execution status, logs & provenance for each workflow

Outline

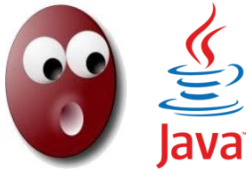
- Introduction & Motivation
- Data Set & Experiment Setup
- Workflow processors & characteristics
- Results & Recommendations
- Conclusions

Workflow Processors & Characteristics

- Types of processing elements in Taverna
  - Web Services
    - WSDL & REST

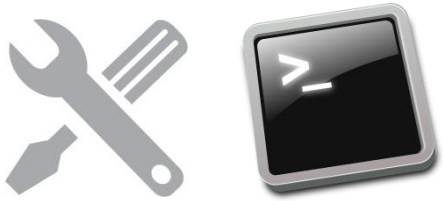


  - Beanshell
    - Execution of Java Code



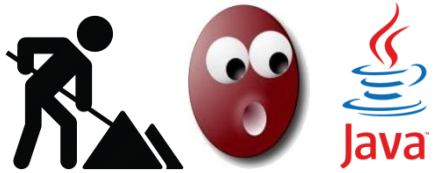
  - Rshell
    - Execution of R Code on RServer Instance

  - Tool
    - Any command available on a ***local*** or ***remote*** system



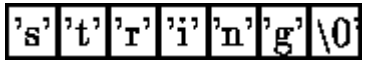
Workflow Processors & Characteristics

- Types of processing elements in Taverna
  - LocalWorker
    - Beanshell with pre-defined functionality
    - E.g. Fetching contents of a URL, Base64 Encoding, ...



  - Xpath queries

  - "Trivial" processors
    - XML Processing
      - Extracting single values / combining to document
    - String Constants



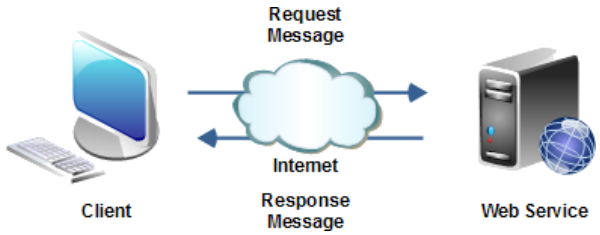
Workflow Processors & Characteristics

- Processor usage in workflows
  - ~12 processor steps per workflow

Processor	%	% nonTrivial
Trivial Processors	> 40	
LocalWorker	18.38	30.4
BeanShell	15.18	25.1
WSDL	4.03	13.2
Tool	3.21	6.7
REST	1.93	5.3
RShell	1.92	3.2
XPath	1.78	3.2

Workflow Processors & Characteristics

- Web Services
  - WSDL & REST

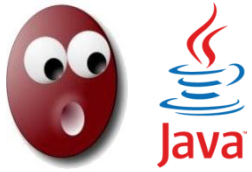


	# WFs	% WFs
WSDL	411	30
REST	172	12.6
Soaplab	10.0	2.8
Any web service	599	42.8

- Vulnerabilities
  - Address not reachable (private INet address)
  - Service not available anymore
  - Service requires authentication
  - Method removed / interface changed

Workflow Processors & Characteristics

- Beanshell
  - Execution of Java code



	# WFs	% WFs
Beanshells	**717**	**49.7**
with Dependencies	**76**	**5.3**

- Vulnerabilities
  - Java version executing the WF is not compatible with code version requirements
  - Dependency not available (not packed with WF)
    - Dependencies in wrong version
  - Beanshell accesses functionality outside the engine

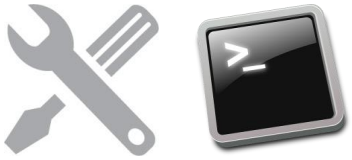
Workflow Processors & Characteristics

- RShell
  - Execution of R code on RServer instance
    - 90 WFs, 337 processors
    - 335 addresses ***local***

	# WFs	% WFs
WFs with Rshell	**90**	**6.2**

- Vulnerabilities
  - Address of RServer not reachable / available
  - Authentication data missing/incorrect
  - Different version of R runtime
  - Custom R package not available / wrong version

Workflow Processors & Characteristics



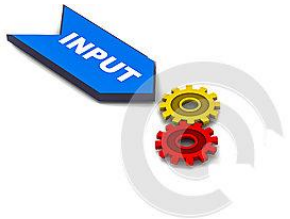
- Tools: local & remote
  - Execution of arbitrary binaries
    - E.g. Image processing
    - Also: Perl / Python scripts

	# WFs	% WFs
Local tools	**240**	**16.6**
Remote tools	**20**	**0.8**

- Vulnerabilities
  - Location not reachable / available
    - Authentication data missing / wrong
  - Tool is not available, or can not be found
  - Wrong version of tool installed

Workflow Processors & Characteristics

- Workflow inputs
  - Runtime parameters for workflow
    - E.g. a data source



	# WFs	% WFs
No input ports	345	23.9
No/some example values	429/97	29.7/6.7
All example values	572	39.6
WF that can be run	917	63.5

- Vulnerabilities
  - No example values provided
  - Example values not correctly formatted
  - Example values not valid anymore

Outline

- Introduction & Motivation
- Data Set & Experiment Setup
- Workflow processors & characteristics
- Results & Recommendations
- Conclusions

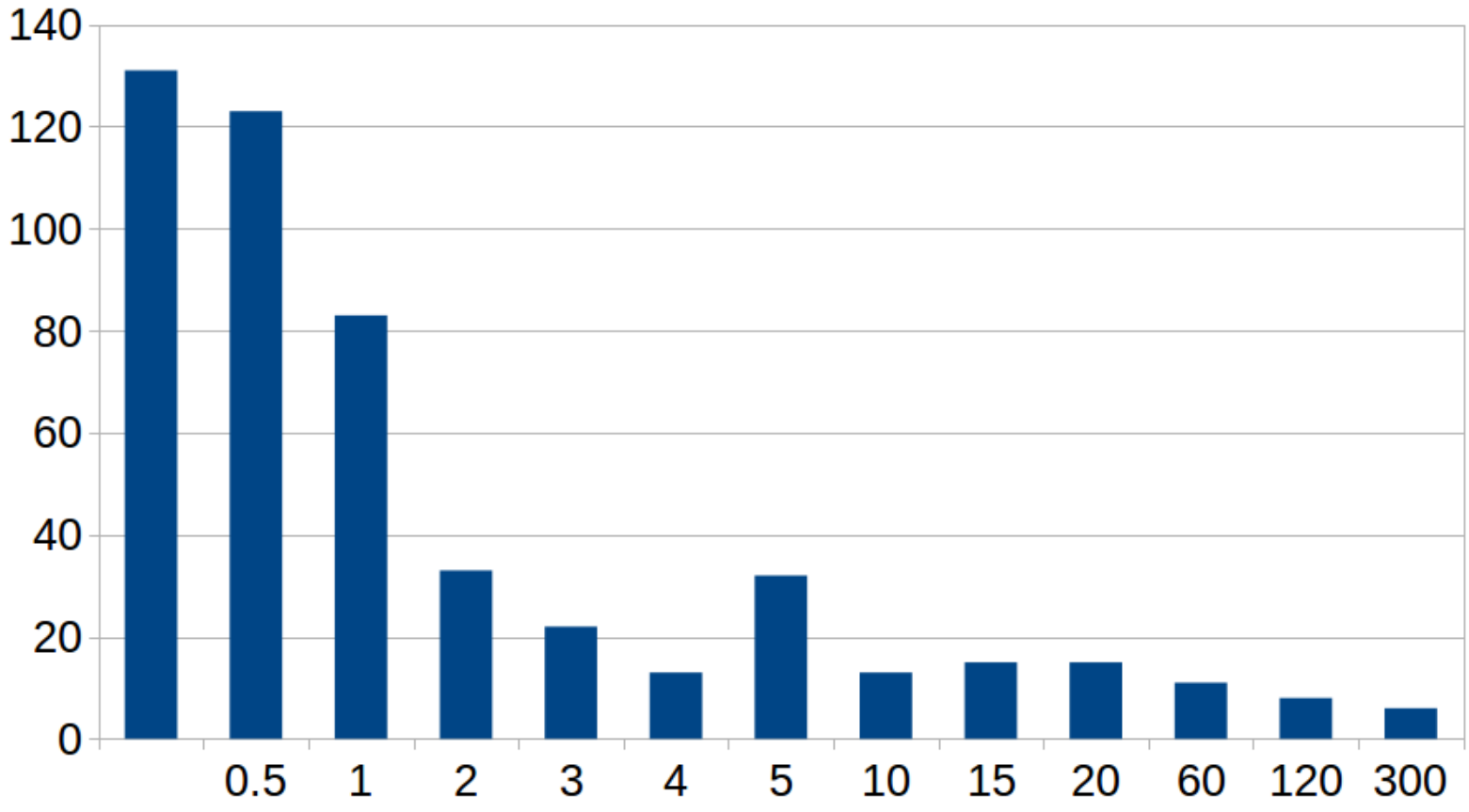
Re-Execution Results

- Data set for re-execution

	# WFs
Original Data Set	**1443**
- Missing input values	**526**
- Disabled processors (WSDL services)	**180**
- Not executable in test environment	**6**
Final Data Set	**731**

Re-Execution Results

- Execution times for workflows (in seconds)



Re-Execution Results

- Execution results

	# WFs	% WFs
Not terminated >48hours	**6**	**0.8**
Execution failed	**384**	**52.5**
Execution successful	**341**	**46.6**

  - Majority of workflows fails
  - No analysis on correctness of results
  - Considering full data set: only 29.2 % are successfully executed

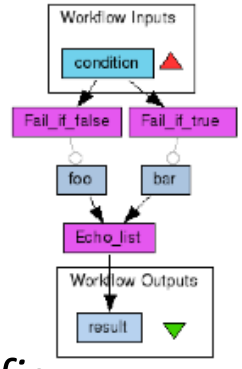
Re-Execution Results

- Reasons for failures
  - Manual analysis of output logs

Reason for Failure	# WFs
REST service unavailable	4
REST service unauthenticated	5
Other unauthenticated (WSDL, Tool)	40
Missing Resources (File, URL)	14
Local/remote Tool unavailable	19

Recommendations

- Capabilities of Workflow engines
  - Provide more scripting languages (e.g. Activiti engine)
    ➔ will reduce number of tool invocations
- Increase expressiveness of WF definition
  - Integrate some aspects from e.g. Research Objects or Context Model [1], e.g. dependency definition
    ➔ will reduce issues with missing/wrong dependencies



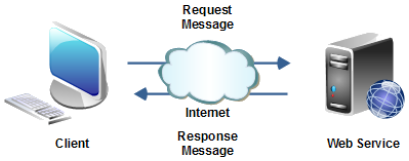
[1] Rudolf Mayer et al. *Ontologies for describing the context of scientific experiment processes*. 10th International Conference on e-Science, Brazil, October 2014

Recommendations

- Tool invocation
  - Dependency definition
  - Allow identification & alternatives for OS-dependent code

  ➔Will reduce number of wrong OS / missing dependencies

- External dependencies
  - Archive & publish utilised WSDL together with workflow ➔ will enable easier substation of service



Recommendations

- Quality check & monitoring
  - Integrated into sharing platform (e.g. myExperiment), similar to WF4Ever project
- Services that check
  - Completeness of definition
    - E.g. all input example parameters are provided
  - Required Java libraries included
- Monitoring for external services to also include REST, RSHell, SSH invocations, ...

Recommendations

- Many aspects can be automated
  - Automatic capturing of process dependencies and required resources [2]
- Verification of ***correct*** execution to be addressed separately [3]

[2] Johannes Binder et al. P*rocess migration framework -- virtualising and documenting business processes*. Workshop Proceedings of the 18th IEEE International EDOC Conference, 2014

[3] Tomasz Miksa et al. *VPlan -- ontology for collection of process verification data*. 11th International Conference on Digital Preservation, 2014

Outline

- Introduction & Motivation
- Data Set & Experiment Setup
- Workflow processors & characteristics
- Results & Recommendations
- Conclusions

Conclusions & Future work

- Workflows should foster reproducibility
- Many shared workflows fail at ***re-execution***
  - "Trivial" aspects (e.g. lack of example input data)
  - Publishing/packaging resources, external dependencies
  - Local dependencies (tool executions)

- Remedies: better documentation, dependency management, monitoring of external services
- Via improvements in workflow engines, workflow definition & monitoring

Thank you!

Rudolf Mayer, *SBA Research, Austria*
rmayer@sba-research.org

http://www.sba-research.org/digital-preservation
http://ifs.tuwien.ac.at/dp/process

secure sba-research.org

dexhelpp

TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology