Converting Scripts into Reproducible Workflow Research Objects

Lucas A. M. C. Carvalho, Khalid Belhajjame, Claudia Bauzer Medeiros
lucas.carvalho@ic.unicamp.br

Baltimore, Maryland, USA
October 23-26, 2016
Background and Motivation

• Data-Intensive Experiments
  - Collection of scripts, programs and (big) data
Background and Motivation

- Data-Intensive Experiments
  - Collection of scripts, programs and (big) data

How to understand, reproduce or reuse data and models of experiments?
Background and Motivation

• Data-Intensive Experiments
  – Collection of scripts, programs and (big) data

How to understand, reproduce or reuse data and models of experiments?

Manual collection and organization of data provenance
Background and Motivation

• Script-based experiments

Example of script code.

What are the inputs and outputs?

How to change this local program for a similar web service?

Difficult to understand, to reuse, and to reproduce.
Background and Motivation

- Scientific Workflows

Example of Scientific Workflow Management System.
Overview

Create

Understand

Reuse

Reproduce
Overview

Create → Understand → Reuse → Reproduce

+
Related Work

- Script-language specific.
- Workflow-engine specific.
- A new language is needed.
- Outcome is **not** an executable workflow.
- Do not collect provenance data of the conversion process.
Two Kind of Experts

**Scientists**
- Domain experts who understand the experiment, and the script (sometimes called *user*);

**Curators:**
- Scientists who are also familiar with workflow and script programming or;
- Computer scientists who are familiar enough with the domain to be able to implement our methodology;
- Responsible for authoring, documenting and publishing workflows and associated resources.
Requirements

1. Produce workflow-like view of the script.
2. Create an executable workflow and compare execution of workflow and script.
3. Modify the workflow resources.
4. Record provenance data.
5. Aggregate all resources to support Reproducibility and Reuse.
1. Produce workflow-like view of the script.
Requirements

2. Create executable workflow and compare execution of workflow and script.

Executable workflow.

≈

Script-based experiment.
3. Modify the workflow resources.

(a) Local

(b) Algorithm A → Algorithm B
4. Record provenance data

- **Workflow Run**
  - wasAssociatedWith: Lucas
  - used: Sample
  - used: "2012-06-01"
  - wasStartedAt: "2012-06-01"
  - wasGeneratedBy: Activity 1
  - used: Output 1
  - used: Output 2

- **Activity 1**
  - wasGeneratedBy: Activity 2
  - used: Sample
  - used: "2012-06-01"

- **Activity 2**
Requirements

5. Aggregate all resources to support Reproducibility and Reuse.

Provenance

Data

Annotations

Authors

Scripts

Concrete workflows

Abstract workflows

Papers and Reports
Methodology

1. Generate Abstract Workflow
2. Create an executable workflow
3. Refine workflow
4. Annotate and check quality
5. Bundle Resources into a Research Object
Workflow Research Object (WRO)

- Research Objects are semantically rich aggregations of resources that bring together data, methods and people in scientific investigations.

- WROs encapsulate scientific workflows and additional information regarding their context and resources.
Running Example

• Molecular Dynamics Simulations
  – Many branches of material sciences, computational engineering, physics and chemistry.
  – Scripts (shell script), programs (NAMD, VMD, Fortran)
  – **Phases**: set up, simulation and analysis of trajectories.
  – **Inputs**: protein structure, simulation parameters and force field files.
  – **Output**: trajectories and analysis results.
Step 1
Generate Abstract Workflow

```
20 structure = $directory_path"/structure.pdb"
21 protein = $directory_path"/protein.pdb"
22 water = $directory_path"/water.pdb"
23 bglc = $directory_path"/bglc.pdb"
24 egrep -v '(TIP3|BGLC)' $structure > $protein
25 grep TIP3 $structure > $water
26 grep BGLC $structure > $bglc
```

Script code.
Step 1

Generate Abstract Workflow

Manually annotate

Script code.

14  # @BEGIN split
15  # @IN initial_structure @URI file:structure.pdb
16  # @IN directory_path @AS directory
17  # @OUT protein_pdb @URI file:{directory}/protein.pdb
18  # @OUT bglc_pdb @URI file:{directory}/bglc.pdb
19  # @OUT water_pdb @URI file:{directory}/water.pdb
20  structure = $directory_path"/structure.pdb"
21  protein = $directory_path"/protein.pdb"
22  water = $directory_path"/water.pdb"
23  bglc = $directory_path"/bglc.pdb"
24  grep –v 'TIP3|BGLC' $structure > $protein
25  grep TIP3 $structure > $water
26  grep BGLC $structure > $bglc
27  # @END split

Annotated script code.
Step 1
Generate Abstract Workflow

Manually annotate

Script code:

```plaintext
20  structure = $directory_path"/structure.pdb"
21  protein = $directory_path"/protein.pdb"
22  water = $directory_path"/water.pdb"
23  bglc = $directory_path"/bglc.pdb"
24  egrep -v '(TIP3|BGLC)' $structure > $protein
25  grep TIP3 $structure > $water
26  grep BGLC $structure > $bglc
```

Script code.

Annotated script code:

```plaintext
14  # @BEGIN split
15  # @IN initial_structure @URI file:structure.pdb
16  # @IN directory_path @AS directory
17  # @OUT protein_pdb @URI file:{directory}/protein.pdb
18  # @OUT bglc_pdb @URI file:{directory}/bglc.pdb
19  # @OUT water_pdb @URI file:{directory}/water.pdb
20  structure = $directory_path"/structure.pdb"
21  protein = $directory_path"/protein.pdb"
22  water = $directory_path"/water.pdb"
23  bglc = $directory_path"/bglc.pdb"
24  egrep -v '(TIP3|BGLC)' $structure > $protein
25  grep TIP3 $structure > $water
26  grep BGLC $structure > $bglc
27  # @END split
```

Annotated script code.

Create workflow-like view

Abstract workflow.
Step 1
Generate Abstract Workflow

YesWorkflow
McPhillips et. al, 2015

- Code comments
- Tags:
  - @begin
  - @end
  - @desc
  - @in
  - @out
  - ...


Annotated script code.

Create Workflow-like view

Abstract workflow.
Step 1

Generate Abstract Workflow

```
# @BEGIN split
# @IN initial_structure @URI file:structure.pdb
# @IN directory_path @AS directory
# @OUT protein_pdb @URI file:{directory}/protein.pdb
# @OUT bglc_pdb @URI file:{directory}/bglc.pdb
# @OUT water_pdb @URI file:{directory}/water.pdb
structure = $directory_path"/structure.pdb"
protein = $directory_path"/protein.pdb"
water = $directory_path"/water.pdb"
bglc = $directory_path"/bglc.pdb"
gegrep -v '(TIP3|BGLC)' $structure > $protein
grep TIP3 $structure > $water
grep BGLC $structure > $bglc
# @END split
```

Annotated script code.

Create Workflow-like view

Abstract workflow.
Step 2
Create an executable workflow

Abstract workflow.

setup

- **split**: input: initial_structure, output: water_pdb, bgic_pdb, protein_pdb, topology_prot, topology_carb
- **psfgen**: input: hyd_psf, hyd_pdb, output: hyd_psf, hyd_pdb
- **solve**: input: wbox_psf, wbox_pdb, output: gh5_psf, gh5_pdb
- **ionize**: input: gh5_psf, gh5_pdb
Step 2
Create an executable workflow

Executable workflow.
Step 2
Create an executable workflow

Create implementation of activities
Copy code blocks from the script.

Executable workflow.
Step (2)
Create an executable workflow

setup

Abstract workflow.

Create implementation of activities

Executable workflow.

```
14  # @BEGIN split
15  # @IN initial_structure @URI file:structure.pdb
16  # @IN directory_path @AS directory
17  # @OUT protein_pdb @URI file:{directory}/protein.pdb
18  # @OUT bglc_pdb @URI file:{directory}/bglc.pdb
19  # @OUT water_pdb @URI file:{directory}/water.pdb
20  # @BEGIN solvate
21  # @BEGIN ionize
22  # @BEGIN constraints_definition
23  # @END split
24  structure = $directory_path"/structure.pdb"
25  protein = $directory_path"/protein.pdb"
26  water = $directory_path"/water.pdb"
27  bglc = $directory_path"/bglc.pdb"
28  grep -v '(TIP3|BGLC)' $structure > $protein
29  grep TIP3 $structure > $water
30  grep RGLC $structure > $bglc
31  # @END split
```
Step 3
Refine executable workflow

Modify resources:
- Algorithms
- Data Sets
- Parallelization
- Web Services
- ...

Executable workflow.
Step 3
Refine executable workflow

Create new version

Executable workflow.

Modify resources:
- Algorithms
- Data Sets
- Parallelization
- Web Services
- ...

New workflow version.
Steps 2 3

Record provenance data: execution traces.

W3C PROV

Executable workflow.
Steps 

Record provenance data: conversion process.

```plaintext
# @BEGIN split
# @IN initial_structure @URI file:structure.pdb
# @IN directory_path @AS directory
# @OUT protein_pdb @URI file:{directory}/protein.pdb
# @OUT bglc_pdb @URI file:{directory}/bglc.pdb
# @OUT water_pdb @URI file:{directory}/water.pdb
structure = $directory_path"/structure.pdb"
protein = $directory_path"/protein.pdb"
water = $directory_path"/water.pdb"
bglc = $directory_path"/bglc.pdb"
egrep -v '(TIP3|BGLC)' $structure > $protein
grep TIP3 $structure > $water
grep BGLC $structure > $bglc
# @END split
```

New workflow version.

W3C PROV

Executable workflow.

Was Derived From

Was Derived From

Was Derived From

Was Associated With

Curator
Step 4
Annotate and check quality

- Annotations describing the workflow.
- Use provenance data
  - To check the quality of the conversion process.
- Run checks to verify the soundness of the workflow.
Step 4
Annotate and check quality

14  # @BEGIN split
15  # @IN initial_structure @URI file:structure.pdb
16  # @IN directory_path @AS directory
17  # @OUT protein_pdb @URI file:{directory}/protein.pdb
18  # @OUT bglc_pdb @URI file:{directory}/bglc.pdb
19  # @OUT water_pdb @URI file:{directory}/water.pdb
20  structure = $directory_path"/structure.pdb"
21  protein = $directory_path"/protein.pdb"
22  water = $directory_path"/water.pdb"
23  bglc = $directory_path"/bglc.pdb"
24  egrep -v '(TIP3|BGLC)' $structure > $protein
25  grep TIP3 $structure > $water
26  grep BGLC $structure > $bglc
27  # @END split

Script code.

Executable workflow.
Step 4
Annotate and check quality

Initial Executable workflow.
Step 4
Annotate and check quality

- Common mistakes during the conversion:
  - not clearly identified the main logical processing units in the script;
  - a mistake when migrating script code into the corresponding activity;
  - not provided the correct input files and parameters;
  - the coding of the workflow itself contained errors.
Step 5
Bundle Resources into a Research Object

Provenance Data
Annotations
Attributions

Script
Concrete workflow(s)
Paper
Abstract workflow
Contributions

- A methodology that guides curators in a principled manner to transform scripts into reproducible and reusable WRO;
- This addresses an important issue in the area of script provenance;
Conclusions

- We addressed issues wrt understanding, reuse and reproducibility of script-based experiments.
- The methodology created was:
  - elaborated based on requirements;
  - showcased via a real world use case from the field of Molecular Dynamics;
- We exploited tools and standards from the scientific community:
  - Scientific Workflows, YesWorkflow, Research Objects, the W3C PROV recommendations and the Web Annotation Data Model.
- The bundle is available at http://w3id.org/w2share/s2rwro/
Next Steps

- Evaluation using other case studies;
- Evaluation of the cost of the effectiveness of our methodology;
- Extension of YesWorkflow to support the semantic annotation of blocks;
- Implementation of tools.
Acknowledgments

- FAPESP (grant # 2014/23861-4)
- CCES/CEPID (grant # 2013/08293-7)
  - Center for Computational Engineering & Sciences
- LIS (Laboratory of Information Systems)
- Prof. Munir Skaf and his group from Institute of Chemistry - Unicamp.
Converting Scripts into Reproducible Workflow Research Objects

Lucas A. M. C. Carvalho, Khalid Belhajjame, Claudia Bauzer Medeiros
lucas.carvalho@ic.unicamp.br

12th eScience 2016
IEEE International Conference
Baltimore, Maryland, USA
October 23-26, 2016