SCAPE
Scalable Preservation Environments

Design of provenance component

Authors
David Withers, Norman Paton (University of Manchester)

September 2012

This work was partially supported by the SCAPE Project. The SCAPE project is co-funded by the European Union under FP7 ICT-2009.4.1 (Grant Agreement number 270137).

Executive Summary

The provenance model provides detailed information about the actions performed and the data produced by the workflow. This document provides a detailed description of the provenance model. An example workflow explains how the provenance model concepts relate to actual workflow executions and how the provenance requirements are satisfied.

Table of Contents

1 Introduction

The process management work package will produce tools to manage complex data processing workflows. These tools will provide:

- A workflow design environment for the specification of complex data preservation workflows.
- A workflow execution environment allowing the use of workflows to enact preservation processes.
- A data provenance management component to record detailed traces of workflow runs.
- Search and discovery facilities for preservation components developed during the project.
- An infrastructure for managing workflows as community assets, facilitating the sharing and reuse of workflows within the digital preservation community.

The scope of this document is the design of the workflow provenance management component. It identifies the information about a workflow execution that should be captured and describes the provenance model that will be implemented in order to capture the behaviour of preservation workflows.

1.1 Role

The role of the provenance component is to provide information about entities and activities used in a workflow execution to produce a piece of data. The provenance information can be used to make assessments about the data's quality, reliability or trustworthiness, as well as simply documenting how a data product was produced.

The provenance component will extend the existing Taverna[1] provenance manager, both in the model and in its implementation architecture, to accommodate the data provenance requirements of the SCAPE project. Detailed provenance traces are generated for each workflow execution, and stored persistently for post mortem analysis of the effects of preservation actions.

The extensions to the existing Taverna provenance manager will:

- Ensure that sufficient information is captured to fulfil the provenance requirements
- Improve to performance of the provenance collection
- Improve the provenance export functionality to produce provenance information in a format compatible with the SCAPE Data Publication Platform

1.2 Context

The SCAPE project is concerned with the preservation of digital objects and provenance plays an important role. The decision to instigate the migration of digital objects in a repository will be influenced by factors such as information provided by the watch component (being developed by the

[1] http://www.taverna.org.uk/

Planning and Watch sub-project), institutional policies and the availability of resources. All of this information is part of the provenance of the digital preservation process.

The provenance component described by this document fits into the overall preservation provenance by providing detailed information about actions performed as part of executing a Taverna workflow. The workflow provenance will be stored in the SCAPE Data Publication Platform (as described in SCAPE Deliverable D2.2).

Workflow provenance will be used within the SCAPE project to:

- Provide the planning component with information on runtime performance and resource requirements of workflows
- Annotate data products with documentation on how they are produced

The SCAPE Execution Platform will execute workflows to preserve the digital objects stored in the repository so it is important that the workflows (themselves digital objects) are also preserved for the long term, as they are part of the overall preservation provenance. This aspect is being dealt with by the related FP7 project Wf4Ever: Advanced Workflow Preservation Technologies[2].

1.3 Out of scope

The provenance component does not provide information about the decisions made, or information considered, that lead to a particular workflow being executed. However, the provenance model described in Section 3 is capable of expressing such information.

Also out of scope for this document is the provenance of the workflow, i.e. who designed it, the version history, where the workflow or its components came from. The SCAPE Component Catalogue, based on myExperiment[3], may record some of this information.

[2] http://www.wf4ever-project.org/

[3] http://www.myexperiment.org/

2 Requirements

A document listing the requirements for the information to be provided by the provenance component was produced for SCAPE checkpoint CP046[4]. These requirements were based on the Taverna provenance requirements originally identified in the Wf4Ever project.

The requirements were also presented[5] and discussed at the Second SCAPE All-Staff Meeting at The Hague. The SCAPE specific requirements resulting from the discussions and feedback are presented here.

The requirements specify the information about a workflow execution that should be provided by the provenance component. The requirements are divided into three levels

- Level 1 – Minimum provenance information that must be provided for all workflow executions.
- Level 2 – Highly desirable provenance information that should be provided for workflow executions
- Level 3 – Additional provenance information that may be provided for some workflow executions

The level of provenance information available for a workflow execution will depend on factors such as the execution environment or the services used within the workflow. The execution environments used in the SCAPE project are the Taverna Workflow Engine and the Parallel Preservation Execution Platform (being developed by TUB for WP6). It is expected that the Taverna Workflow Engine will be able to produce provenance information at least levels 1 and 2. The provenance capabilities of the Parallel Preservation Execution Platform are as yet unknown.

2.1 Provenance Terminology

The provenance component requirements use the following terminology when describing the information that each level provides:

- Workflow – a Taverna workflow
- Entity – a data value used or generated by a process or workflow; either the actual data value or a reference to the data value
- Role – the Taverna port that an entity is associated with

The key words " MUST", " SHOULD", " SHOULD NOT" and "MAY" are to be interpreted as described in RFC2119[6]

[4] http://wiki.opf-labs.org/display/SP/PT.WP.4+Task+2+CP046+Requirements+documents+for+provenance+component
[5] https://portal.ait.ac.at/sites/Scape/Shared%20Documents/Meetings/Second%20All-Staff%20Meeting/Presentations/SCAPE-PT.WP.4-TavernaProvenance-2012-02-28.pptx

2.2 Level 1 - minimum provenance information

The provenance component MUST be able to provide the following information, where appropriate identified uniquely at least within the scope of a related workflow execution. Globally accessible resources SHOULD be identified with a globally unique identifier.

- Entities used by a workflow - at least identifier/URL
- Entities generated by a workflow - at least identifier/URL
- For each process execution, a description of what entities it used (at least identifier/URL), and in what role
- For each process execution, a description of what entities it generated (at least identifier/URL), and in what role
- For each process execution which failed, an indication of the error occurring. This could be as direct information about the failure and also an indication if the generated entities are non-data-carrying errors (e.g. Taverna's error documents)

The minimum provenance information specifies the information required to construct a provenance trace from workflow inputs via process executions for all entities created by a workflow, including intermediate entities. Each output entity should be traceable to the initial workflow inputs and/or initial non-input process executions.

The actual data values passed through the workflow are not included in the minimum requirements, but all entities should have identifiers. A second export of the same workflow execution provenance SHOULD use the same entity identifiers. A second workflow execution with the same byte-wise input values SHOULD NOT have the same identifiers for *generated* workflow and process entities, but SHOULD have the same identifiers for workflow input entities if they are coming from the same source (e.g. a file).

2.3 Level 2 - Highly desirable provenance information

The provenance component SHOULD be able to provide the following information for any workflow execution

- A reference to the workflow instance that defines a specific workflow execution (including inputs, options, etc.). Note that this is distinct from a workflow template that can be instantiated with different inputs by different workflow instances. If a workflow is repeated, there may be several workflow executions defined by a workflow instance.
- A unique identifier, or information from which a globally unique identifier can be constructed for any workflow execution (e.g. host name and timestamp for workflow execution)
- Date, time and time zone at which the workflow execution started
- Date, time and time zone at which the workflow execution completed
- Identification of the person or other agent that initiated the workflow execution
- Identification of the host (or cluster, or hosts) on which the workflow was run
- Software or service(s) used to perform each process execution (possibly by reference to an element in the corresponding workflow instance)

[6] http://www.ietf.org/rfc/rfc2119.txt

- Date, time and time zone at which any external services were invoked, or any software image was retrieved and loaded (if materially different from the time at which the process execution was started)
- Date, time and time zone at which each process execution started
- Date, time and time zone at which each process execution completed
- Identification of the host (or cluster, or hosts) on which each process execution was run (WSDL endpoint, SSH node, REST URL)
- For each process execution (output) which failed, details about the error, such as a message and stack trace
- For each entity which can be represented as a binary object, a checksum (SHA-1 or SHA-256) of the binary value
- For each entity, the date, time and time zone it was created
- For each entity used in a process execution, the date, time and time zone it was accessed by that process execution. (This is particularly relevant for external entities.)
- For each external entity accessed via the Web, the URI from which it was retrieved
- For each workflow input/output entity, the actual values as a binary object
- Details of entity list/collection memberships (list of entities as another entity)
- Resources (memory, CPU time, transient disk space, special compute resources) used by each process execution

Note for entity checksums: not all (intermediate process) entities may be representable as a binary object, e.g. a reference to a JVM object. Some entities may not have a uniform binary representation (e.g. a table in Galaxy). The actual value may be inaccessible because it is large, secured or in a different system such as a data repository - but should in these cases generally still have an URI reference.

2.4 Level 3 - Additional provenance information

The provenance component MAY be able to provide the following information:

- User comments and annotations about a particular execution
- For each entity used or generated, information that can be used to verify its integrity (e.g. checksum, details of web retrieval transaction, etc.)
- For each software or system used in a process execution, information that can be used to verify its integrity
- For each software or system used, including the workflow engine, version, path and compile information (32/64-bit, OS, etc.)
- For each external resource used (entity, software or system), links to human- and machine-readable information about that resource (this may be in the workflow instance).
- For each process execution, an indication of whether it is a "shim" process, or something more substantial
- For each process execution (output) which gave a warning, details about this warning
- For external processes - metadata about the service at time of invocation, e.g. copy of WSDL and XSD, HTTP headers from server, etc.
- For external processes - a full protocol-level trace of the invocation - e.g. HTTP headers and body of request(s) and response(s)

- For each entity retrieved from external sources, the protocol-level trace of its retrieval e.g. HTTP headers of request and response
- For each workflow input/output entity, the actual values as a binary object
- For each process input/output entity, the actual values as a binary object

3 Provenance Model

3.1 Requirements for the workflow provenance model

The requirements considered when choosing the workflow provenance were that it must:

- Be a standard provenance format
- Be able to be expressed as RDF triples
- Have support from, and be used by, the wider provenance community
- Have existing tooling that can be utilized
- Meet the requirements specified in Section 2

Based on these criteria OPM[7] and PROV[8] models were investigated. The PROV model was chosen as it was found to best capture the provenance information as specified by the requirements.

3.2 The PROV provenance model

PROV is a provenance data model for building representations of the entities, people and processes involved in producing a piece of data or thing in the world. The core types of the PROV data model are:

- **Entity** – a physical, digital, conceptual, or other kind of thing. For example, an entity could be a document or an image in a repository, or it could be a document that will be generated by and activity at some point in the future.
- **Activity** – something that generates new entities, usually making use of exiting entities. For example, an activity may generate a new version of an existing document in a different format.
- **Agent** – something or someone that was responsible for or otherwise associated with what happened in an activity. An agent could be a person, a piece of software, or an organization. For example, an agent may be a piece of software that converts a document into PDF format.

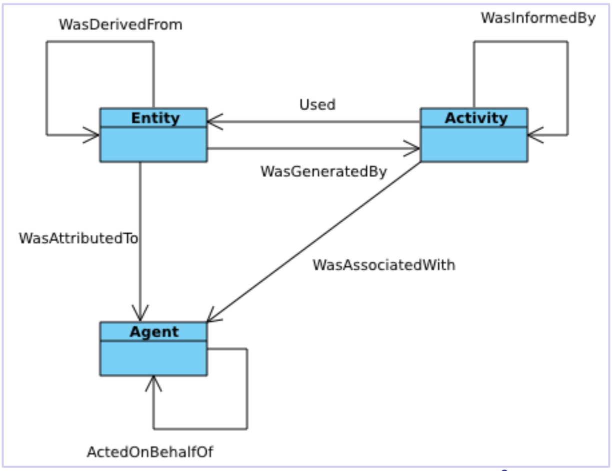


Figure 1 Core PROV types and their relationships[9]

[7] http://openprovenance.org/
[8] http://www.w3.org/TR/prov-primer/
[9] http://www.w3.org/TR/prov-dm/#prov-core-structures-top

Other main concepts of the PROV data model are:

- **Role** – describes the function or the part that an entity played in an activity.
- **Derivation** – describes how entities are derived from other entities.
- **Plan** – pre-defined procedure that an agent follows when executing an activity. For example, a plan could be a set of instructions or a workflow.
- **Time** – the timing of significant events, including when an entity was generated or used, or when an activity started and finished. For example, when a new version of a document was created (generation time), or when a document was migrated (start and end of the migration activity).

3.3 The wfprov and tavernaprov PROV extensions

For the purpose of describing provenance of workflow executions in a high-level perspective, the Wf4Ever project has developed the *wfprov*[10] model. This includes descriptions of the abstract workflow using the *wfdesc* model, and can be mapped as an extension of PROV-O. Here we propose using *wfprov* mainly to distinguish between workflow runs and process runs. A workflow run describes the activity of running a workflow. A process run represents a step of the workflow execution. Process runs have dependencies on inputs, modelled with the relationship *wfprov:usedInput*, and generate results, linked by the relationship *wfprov:wasOutputFrom*.

tavernaprov[11] is an ontology extending PROV-O and wfprov, and which has been developed together with the Wf4Ever project in order to describe any Taverna-specific behaviour not covered by the general models, such as error documents and iterations. Here we propose to reuse *tavernaprov* for such edge cases, as detailed in later sections.

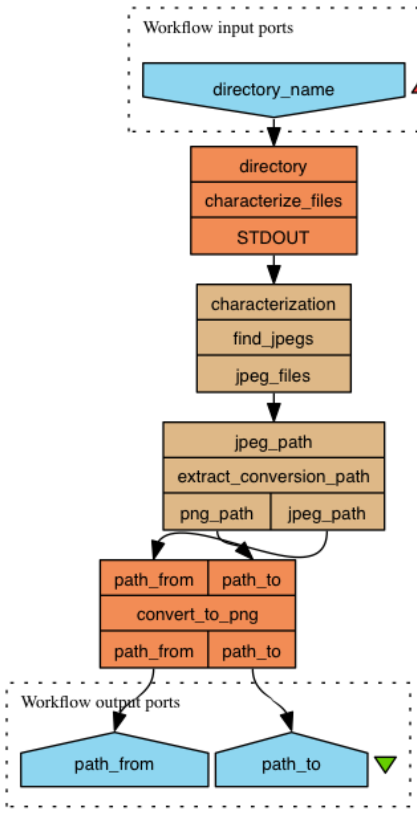
3.4 Example PROV export from a Taverna workflow execution

The following example further describes the concepts of the PROV data model and how they relate to the provenance information for a Taverna workflow execution.

3.4.1 Example workflow

The example workflow takes a path to a directory, characterizes the files in the directory, discovers images files encoded in JPEG format and converts the JPEG files to PNG format. The actual workflow inputs and outputs are references to the data (file paths for the example workflow execution).

[10] http://purl.org/wf4ever/model
[11] http://ns.taverna.org.uk/2012/tavernaprov/



3.4.2 Example provenance export

The examples shown in the provenance export are based on a sample run of the example workflow. The workflow was run using the Taverna Workbench 2.4[12]. The PROV provenance export was produced using a prototype Taverna PROV plugin[13] produced as part of the Wf4Ever project.

The PROV samples use the formal PROV ontology (PROV-O[14]) to represent the PROV descriptions as RDF triples (shown using the Turtle[15] notation). The namespace prefix *prov:* denotes terms from the PROV ontology. Examples are given for each of the PROV concepts and also for the requirements from Section 2.

3.4.2.1 Entities

Entities in a workflow execution are the data that is used or generated. The **entity** could be the data value itself or a reference to the data, such as a file or an entry in a repository.

[12] http://www.taverna.org.uk/download/workbench/2-4/
[13] http://wf4ever.github.com/taverna-prov/
[14] http://www.w3.org/TR/prov-o/
[15] http://www.w3.org/TR/2011/WD-turtle-20110809/

The example workflow has a single input port *directory_name*. The value of the workflow input has been saved in a file, *directory_name.txt*, and this is modelled as an **entity**.

```
<directory_name.txt> a prov:Entity .
```

There are two workflow output ports, *path_from* and *path_to*. The outputs are lists of values and have been saved as directories containing a file for each element in the list. The name of the file indicates the position of the element in the list. Each list and list element is modelled as an **entity**.

```
<path_from/> a prov:Entity, prov:Collection .
<path_from/0.txt> a prov:Entity .
<path_from/1.txt> a prov:Entity .
<path_from/2.txt> a prov:Entity .
…
<path_to/> a prov:Entity, prov:Collection .
<path_to/0.txt> a prov:Entity .
<path_to/1.txt> a prov:Entity .
<path_to/2.txt> a prov:Entity .
…
```

The PROV model also has the concept of alternative views of the same **entity**. The following **entity** is a value stored in the Taverna reference manager that has also been saved to the file *path_from/2.txt*

```
<http://ns.taverna.org.uk/data/run1/ref/0c3dc264-6dcc-4b2d > a prov:Entity ;
   prov:alternateOf <path_from/2.txt> .
```

The intermediate values were not exported for this example so there are only **entities** for the data reference. The **activity** that generated the entity is also shown.

```
<http://ns.taverna.org.uk/data/run1/ref/175ab497-2fd9-4bde> a prov:Entity ;
   prov:wasGeneratedBy <http://ns.taverna.org.uk/2011/run/run1/process/p11/> .
```

3.4.2.2 Activities

Activities generate new **entities** and change entity attributes to create new **entities**. Activities also make use of existing **entities**. In a workflow the invocation of a processor is an **activity**.

For example, a process invocation, *p15*, is an **activity**,

```
<http://ns.taverna.org.uk/2011/run/run1/process/p15/> a prov:Activity .
```

that used **entities**

```
<http://ns.taverna.org.uk/2011/run/run1/process/p15/> a prov:Activity ;
   prov:used <http://ns.taverna.org.uk/2011/data/run1/ref/168110bc-92d9-48f8> ;
   prov:used <http://ns.taverna.org.uk/2011/data/run1/ref/acc9aaa8-448c-484d> .
```

and generated new **entities**

```
<http://ns.taverna.org.uk/2011/run/run1/process/p15/> a prov:Activity ;
   prov:generated <http://ns.taverna.org.uk/2011/data/run1/ref/0c3dc264-6dcc-4b2d> ;
   prov:generated <http://ns.taverna.org.uk/2011/data/run1/ref/22ad1809-aeb5-4d1e> .
```

3.4.2.3 Agents

Agents are responsible for an **activity** taking place. The person who starts a workflow running would be an **agent**. **Agents** can act on behalf of others so a Taverna Server could be an **agent** acting on behalf of a person to carry out the **activity** of executing a workflow.

```
<http://ns.taverna.org.uk/2011/run/run1/> a prov:Activity ;
   prov:wasStartedBy :DavidWithers .

:DavidWithers a prov:Person, prov:Agent .
```

3.4.2.4 Roles

Entities are used by **activities** in particular **roles**. In a workflow, the **roles** are the input and output ports of the workflow and the processors. For this example, the **roles** are the workflow input and output ports:

```
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/in/directory_name> a prov:Role .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/out/path_from> a prov:Role .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/out/path_to> a prov:Role .
```

and processor input and output ports:

```
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/characterize_files/in/directory>
   a prov:Role .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/characterize_files/out/STDOUT>
   a prov:Role .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/convert_to_png/in/path_from>
   a prov:Role .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/convert_to_png/in/path_to>
   a prov:Role .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/convert_to_png/out/path_from>
   a prov:Role .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/convert_to_png/out/path_to>
   a prov:Role .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/extract_conversion_path/in/jpeg_path>
   a prov:Role .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/extract_conversion_path/out/jpeg_path>
```

```
  a prov:Role .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/extract_conversion_path/out/png_path>
  a prov:Role .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/find_jpegs/in/characterization>
  a prov:Role .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/find_jpegs/out/jpeg_files>
  a prov:Role .
```

Roles show how activities use or generate entities. For example, the output port *png_path* is the **role** in which the **activity** *extract_conversion_path* generates a data **entity**.

```
<http://ns.taverna.org.uk/data/run1/ref/175ab497-2fd9-4bde> a prov:Entity ;
  prov:wasGeneratedBy <http://ns.taverna.org.uk/run/run1/process/p11/> ;
  prov:qualifiedGeneration [  a prov:Generation ;
    prov:activity <http://ns.taverna.org.uk/run/run1/process/p11/> ;
    prov:hadRole
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/extract_conversion_path/out/png_path>
  ] .
```

The input port *jpeg_path* is the **role** in which activity *extract_conversion_path* used a data **entity**.

```
<http://ns.taverna.org.uk/2011/run/run1/process/p11/> a prov:Activity ;
  prov:used <http://ns.taverna.org.uk/data/run1/ref/568110bc-92d9-48f8> ;
    prov:qualifiedUsage [ a prov:Usage ;
    prov:entity <http://ns.taverna.org.uk/data/run1/ref/568110bc-92d9-48f8> ;
    prov:hadRole
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/extract_conversion_path/in/jpeg_path>
] .
```

3.4.2.5 Plans

The **plans** for this example are the workflow itself,

```
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/> a prov:Plan .
```

the processors in the workflow,

```
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/characterize_files/> a prov:Plan .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/convert_to_png/> a prov:Plan .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/extract_conversion_path/> a prov:Plan .
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/find_jpegs/> a prov:Plan .
```

Plans are followed by **agents** when associated with execution of an **activity**. For example, an **agent** follows the plan *characterize_files* when executing the **activity** *p2*.

```
<http://ns.taverna.org.uk/2011/run/run1/process/p2/> a prov:Activity ;
```

```
  prov:qualifiedAssociation [ a prov:Association ;
    prov:agent _:node174its72cx1 ;
    prov:hadPlan <http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/characterize_files/>
  ] .
```

3.4.2.6 Time

The **time** at which events occur is a very important part of provenance. PROV allows the timing of significant events to be described, including when an **entity** was generated or used, or when an **activity** started and finished. For example, invocation of a Taverna processor will have a *startedAtTime* and an *endedAtTime*.

```
<http://ns.taverna.org.uk/2011/run/run1/process/p13/> a prov:Activity ;
 prov:startedAtTime "2012-08-14T16:59:54.372+01:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>;
 prov:endedAtTime "2012-08-14T16:59:54.383+01:00"^^<http://www.w3.org/2001/XMLSchema#dateTime> ;
```

3.4.3 Level 1 requirements

The following examples show how PROV can be used to satisfy the requirements specified in Section 2.2

- Entities used by a workflow - at least identifier/URL

```
<http://ns.taverna.org.uk/2011/run/run1/> a prov:Activity, wfprov:WorkflowRun ;
  prov:used <http://ns.taverna.org.uk/data/run1/ref/3b93a631-aaa0-4a26> .
```

- Entities generated by a workflow - at least identifier/URL

```
<http://ns.taverna.org.uk/2011/run/run1/> a prov:Activity, wfprov:WorkflowRun .

<http://ns.taverna.org.uk/2011/data/run1/ref/0c3dc264-6dcc-4b2d> a prov:Entity ;
  prov:wasGeneratedBy <http://ns.taverna.org.uk/2011/run/run1/> .
```

- For each process execution, a description of what entities it used (at least identifier/URL), and in what role

```
<http://ns.taverna.org.uk/2011/run/run1/process/p15/> a prov:Activity, wfprov:ProcessRun ;
  prov:used <http://ns.taverna.org.uk/data/run1/ref/168110bc-92d9-48f8> ;
  prov:qualifiedUsage [ a prov:Usage ;
    prov:entity <http://ns.taverna.org.uk/data/run1/ref/168110bc-92d9-48f8> ;
    prov:hadRole
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/convert_to_png/in/path_from>
  ] .
```

- For each process execution, a description of what entities it generated (at least identifier/URL), and in what role

```
<http://ns.taverna.org.uk/2011/run/run1/process/p15/> a prov:Activity, wfprov:ProcessRun .

<http://ns.taverna.org.uk/2011/data/run1/ref/0c3dc264-6dcc-4b2d> a prov:Entity ;
  prov:wasGeneratedBy <http://ns.taverna.org.uk/2011/run/run1/process/p15/> ;
  prov:qualifiedGeneration [ a prov:Generation ;
    prov:activity <http://ns.taverna.org.uk/2011/run/run1/process/p15/> ;
    prov:hadRole
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/convert_to_png/out/path_from>
  ] .

<http://ns.taverna.org.uk/2011/data/run1/ref/22ad1809-aeb5-4d1e> a prov:Entity ;
  prov:wasGeneratedBy <http://ns.taverna.org.uk/2011/run/run1/process/p15/> ;
  prov:qualifiedGeneration [ a prov:Generation ;
    prov:activity <http://ns.taverna.org.uk/2011/run/run1/process/p15/> ;
    prov:hadRole
<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/convert_to_png/out/path_to>
  ] .
```

- For each process execution (output) which failed, an indication of this error occurring. This could be as direct information about the execution, but also an indication if the generated entities are non-data-carrying errors (e.g. Taverna's error documents)

```
:error1 a prov:Entity, tavernaprov:Error;
  prov:wasGeneratedBy <http://ns.taverna.org.uk/2011/run/run1/process/p15/> .
```

3.4.4 Level 2 requirements

The following examples show how PROV can be used to satisfy the requirements specified in Section 2.3

- A reference to the workflow instance that defines a specific workflow execution (including inputs, options, etc.).

```
<http://ns.taverna.org.uk/2011/run/run1/> a prov:Activity, wfprov:WorkflowRun ;
  wfprov:describedByWorkflow <http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1>

<http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/> a wfdesc:Workflow, wfdesc:WorkflowInstance ;
  wfdesc:hasInput <in1.txt>, <in2.txt> .
```

- A unique identifier, or information from which a globally unique identifier can be constructed for any workflow execution (e.g. host name and timestamp for workflow execution)

```
<http://ns.taverna.org.uk/2011/run/run1/> a prov:Activity , wfprov:WorkflowRun .
```

- Date, time and time zone at which the workflow execution started

```
<http://ns.taverna.org.uk/2011/run/run1/> a prov:Activity, wfprov:WorkflowRun ;
 prov:startedAtTime "2012-08-14T16:59:52.412+01:00"^^<http://www.w3.org/2001/XMLSchema#dateTime> .
```

- Date, time and time zone at which the workflow execution completed

```
<http://ns.taverna.org.uk/2011/run/run1/> a prov:Activity, wfprov:WorkflowRun ;
 prov:endedAtTime "2012-08-14T17:00:07.661+01:00"^^<http://www.w3.org/2001/XMLSchema#dateTime> .
```

- Identification of the person or other agent that initiated the workflow execution

```
<http://ns.taverna.org.uk/2011/run/run1/> a prov:Activity, wfprov:WorkflowRun ;
  prov:wasStartedBy :DavidWithers .

:DavidWithers a prov:Person, prov:Agent .
```

- Identification of the host (or cluster, or hosts) on which the workflow was run

```
:wfEngine a prov:Agent, wfprov:WorkflowEngine, tavernaprov:TavernaEngine ;
  prov:atLocation <http://myLaptop.example.com/>

<http://ns.taverna.org.uk/2011/run/run1/> a prov:Activity ;
  prov:qualifiedAssociation [ a prov:Association ;
    prov:agent :wfEngine;
  ] .
```

- Software or service(s) used to perform each process execution (possibly by reference to an element in the corresponding workflow instance)

```
<http://ns.taverna.org.uk/2011/run/run1/process/p2/> a prov:Activity, wfprov:ProcessRun ;
  prov:qualifiedAssociation [ a prov:Association ;
    prov:agent _:node174its72cx1 ;
    prov:hadPlan <http://ns.taverna.org.uk/wfBundle/wb1/workflow/Wf1/processor/characterize_files/>
  ] .
```

- Date, time and time zone at which any external services were invoked, or any software image was retrieved and loaded (if materially different from the time at which the process execution was started).

```
:toolExecution a prov:Activity ;
 prov:startedAtTime "2012-08-14T16:59:57.725+01:00"^^<http://www.w3.org/2001/XMLSchema#dateTime> .
```

- Date, time and time zone at which each process execution started

```
<http://ns.taverna.org.uk/2011/run/run1/process/p15/> a prov:Activity, wfprov:ProcessRun ;
```

```
 prov:startedAtTime "2012-08-14T16:59:56.905+01:00"^^<http://www.w3.org/2001/XMLSchema#dateTime> .
```

- Date, time and time zone at which each process execution completed

```
<http://ns.taverna.org.uk/2011/run/run1/process/p15/> a prov:Activity, wfprov:ProcessRun ;
 prov:endedAtTime "2012-08-14T16:59:58.230+01:00"^^<http://www.w3.org/2001/XMLSchema#dateTime> .
```

- Identification of the host (or cluster, or hosts) on which each process execution was run (WSDL endpoint, SSH node, REST URL)

```
<http://ns.taverna.org.uk/2011/run/run1/process/p15/> a prov:Activity, wfprov:ProcessRun ;
   prov:wasInformedBy :toolExecution .

:toolExecution a prov:Activity;
   prov: atLocation <http://server5.example.com> .
```

- For each process execution (output) which failed, details about the error, such as a message and stack trace

```
:error1 a prov:Entity, tavernaprov:Error;
   prov:wasGeneratedBy <http://ns.taverna.org.uk/2011/run/run1/process/p15/> ;
   tavernaprov:errorMessage “Example error message” ;
   tavernaprov:stackTrace “Example stack trace” .
```

- For each entity which can be represented as a binary object, a checksum (SHA-1 or SHA-256) of the binary value

```
<http://ns.taverna.org.uk/2011/data/run1/ref/22ad1809-aeb5-4d1e > a prov:Entity;
   tavernaprov:content [ a tavernaprov:Content
      tavernaprov:sha512
“07e547d9586f6a73f73fbac0435ed76951218fb7d0c8d788a309d785436bbb642e93a252a954f23912547d1e8a
3b5ed6e1bfd7097821233fa0538f3db854fee6” .
   ] .
```

- For each entity, the date, time and time zone it was created

```
<http://ns.taverna.org.uk/2011/data/run1/ref/22ad1809-aeb5-4d1e > a prov:Entity ;
   prov:generatedAtTime
      "2012-08-14T16:59:57.405+01:00"^^<http://www.w3.org/2001/XMLSchema#dateTime> .
```

- For each entity used in a process execution, the date, time and time zone it was accessed by that process execution. (This is particularly relevant for external entities.)

```
<http://ns.taverna.org.uk/2011/run/run1/process/p15/> a prov:Activity, wfprov:ProcessRun ;
   prov:used <http://ns.taverna.org.uk/data/run1/ref/168110bc-92d9-48f8> ;
```

```
  prov:qualifiedUsage [ a prov:Usage ;
     prov:entity <http://ns.taverna.org.uk/data/run1/ref/168110bc-92d9-48f8> ;
     prov:atTime  "2012-08-14T16:59:57.245+01:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
 ] .
```

- For each external entity accessed via the Web, the URI from which it was retrieved

```
<http://ns.taverna.org.uk/data/run1/ref/175ab497-2fd9-4bde> a prov:Entity ;
  prov:wasQuotedFrom <http://www.example.com/example.xml> .
```

- For each workflow input/output entity, the actual values as a binary object

```
<http://ns.taverna.org.uk/data/run1/ref/175ab497-2fd9-4bde> a prov:Entity ;
   prov:value “Example value” .
```

- Details of entity list/collection memberships (list of entities as another entity)

```
<http://ns.taverna.org.uk/data/run1/list/f9c889ac-0e17-488a/false/1> a prov:Collection ;
   prov:hadMember <http://ns.taverna.org.uk/2011/data/run1/ref/22ad1809-aeb5-4d1e > .
```

4 Acknowledgements

The authors gratefully acknowledge Khalid Belhajjame and Stian Soiland-Reyes from the Wf4Ever project for their input to this document.