Persistent Identification and Citation of Software

Catherine Jones
B. Matthews, I. Gent, T. Griffin and J. Tedds
Feb 2015
Starting Point

- Software underpins research in many disciplines
- Data can be meaningless without the software which created, analysed or displays them
- Software is complicated with many dependencies
- Writing software is an intellectual endeavour in its own right

Jisc funded **Software Re-use, Repurposing and Reproducibility** project
Why am I interested?

I once wrote software....

Only a printout of photographs of the screen remain c.1990 – IBM 3090 VM & Rexx

Only printouts of a couple of programmes and the original specification documents remain, there may also be some postscript files of the documentation....

But I kept them because they were an important intellectual record for me....but they are not in a reusable or citeable form
What do we mean by software?

- Software is a general term, scale can vary from:
  - One off script (post-it note?)
  - Script used on a regular basis (technical report?)
  - Complete programme providing a set of functionality (journal article?)
  - Suite of programmes providing a wider range of functionalities (series? Journal issue?)
- It can be written from scratch or a local modification of existing code
- Don’t forget the environment and dependencies…
Why persistently identify software?

- The identification of the software used in a specific circumstance: **reuse and/or reproducibility**
- Citation of the software: **credit assignment**
- A preservation repository needs to know what is in the collection: **identification**
- To distinguish between different versions: **identification**
- To locate relevant software: **discovery**
What is being Identified?

Product is the concept. Version is a specific set of functionality. Variant is a version for a specific environment. Instance is a specific variant on a specific machine.

Any of these may be needed to be persistently identified depending on the situation.
Stakeholders & Motivation

• The further from the creation of the code, the greater the interest in preserving it is.
  – Research software engineers:
    • “Good software management practice is all that is needed”
    • We suspect those who need to reuse code may not agree …
  – Computational scientists who write code:
    • Haven’t thought about it but acknowledgement/credit and reproducibility are good in theory
  – Digital Preservation experts:
    • Very interested as they know they will have to do it
    • Recognised as a key trend in 2016 at iPres

• The idea of being able to prove reproducibility of results from software analysis is gaining traction

• There are overlaps with Continuous Integration tools or deployment processes
DataCite metadata

- Report: [http://purl.org/net/epubs/work/24058274](http://purl.org/net/epubs/work/24058274)
- Gave guidance on how to apply DataCite to software
  - Some suggestions may be adopted by DataCite
- Major issues were with
  - Relationship – too publication focused
  - ResourceType – Software too general a term for retrieval

<table>
<thead>
<tr>
<th>DataCite Property</th>
<th>&lt;name of the property&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataCite Description</td>
<td>Explanatory text provided by DataCite</td>
</tr>
<tr>
<td>Purpose</td>
<td>Discussion on how this applies to Software. This may include examples</td>
</tr>
<tr>
<td>Stakeholders</td>
<td>Who is involved in the decision and why they might add this metadata</td>
</tr>
<tr>
<td>Questions</td>
<td>Questions to help those who want to add DOIs to software to make the right decisions for their case</td>
</tr>
</tbody>
</table>
DataCite fields I

CREATOR

• To identify the people responsible for the software
• The creator may not be a straightforward item to ascertain as software can have a long life-span and may be worked on by many people.

TITLE

• Is the mandatory field with the most information.
  – Will the name used be unique enough for it to be found and distinguished from other search results?
  – Is the official name different from the common name?
  – What effect is versioning going to have on the name?
DataCite fields II

DESCRIPTION

• Used for extra information.

• **Abstract** and **Other** most commonly used
  – More information about the purpose of the software or releases or live repo

• Some information needed to understand the object doesn’t have an obvious field

• Report suggested new DescriptionType of TechnicalInfo
DataCite fields III

Relation type

• Relationships of particular relevance are:
  – RELEASES: IsNewVersionOf and IsPreviousVersionOf
  – MODULES IsPartOf and HasPart
  – ENVIRONMENT: IsVariantFormOf and IsOriginalFormOf
    (Different operating systems)
  – FORK: IsContinuedBy and Continues or perhaps IsSourceOf and IsDerivedFrom

• Make clear that IsCompiledBy and Compiles are not used in the computing sense

• Under consideration by the DataCite Metadata working group
Example: Mantid

- Mantid is an open source development for data analysis in the Neutron Scattering Community with a large distributed software development team.
- The software is used “as is” and there is no expectation that there will be local user modifications.
- Approach:
  - Product level DOI for the Mantid concept – lots of creators.
  - Each new version has its own DOI, crediting those who worked on that version.
    - Uses IsPartOf to link back to the Product and IsNextVersion/IsPreviousVersion to relate version levels.
- Users of the software can cite the software version used for the analysis.
Example: OpenSpecimen at Cancer Research Biobank

- A tailored version of OpenSpecimen was developed with plans for further use/adaption to remove reliance on project funded researchers.
- A new version of OpenSpecimen led to the potential for divergence with associated technical support issues.
- Having code in a managed repo doesn’t remove the need for technically proficient researchers.
- One resolution would be the creation of recomputation platforms to enable redeployment of code.
Conclusions and Next Steps

• For persistent identification and citation to become commonplace, **credit culture needs to change**
• Unresolved issues around the preservation and reuse of modified code

**Next Steps**
• Exploring persistent identification within computational science community (age of codes)
• Force11 Working Group on Software Citation
• Examining overlaps with Jenkins/Deployment
• Contributing to DataCite Metadata Working Group on relationships

Catherine.jones@stfc.ac.uk