Software Deposit: How to review a software deposit

Michael Jackson (ed.), The Software Sustainability Institute

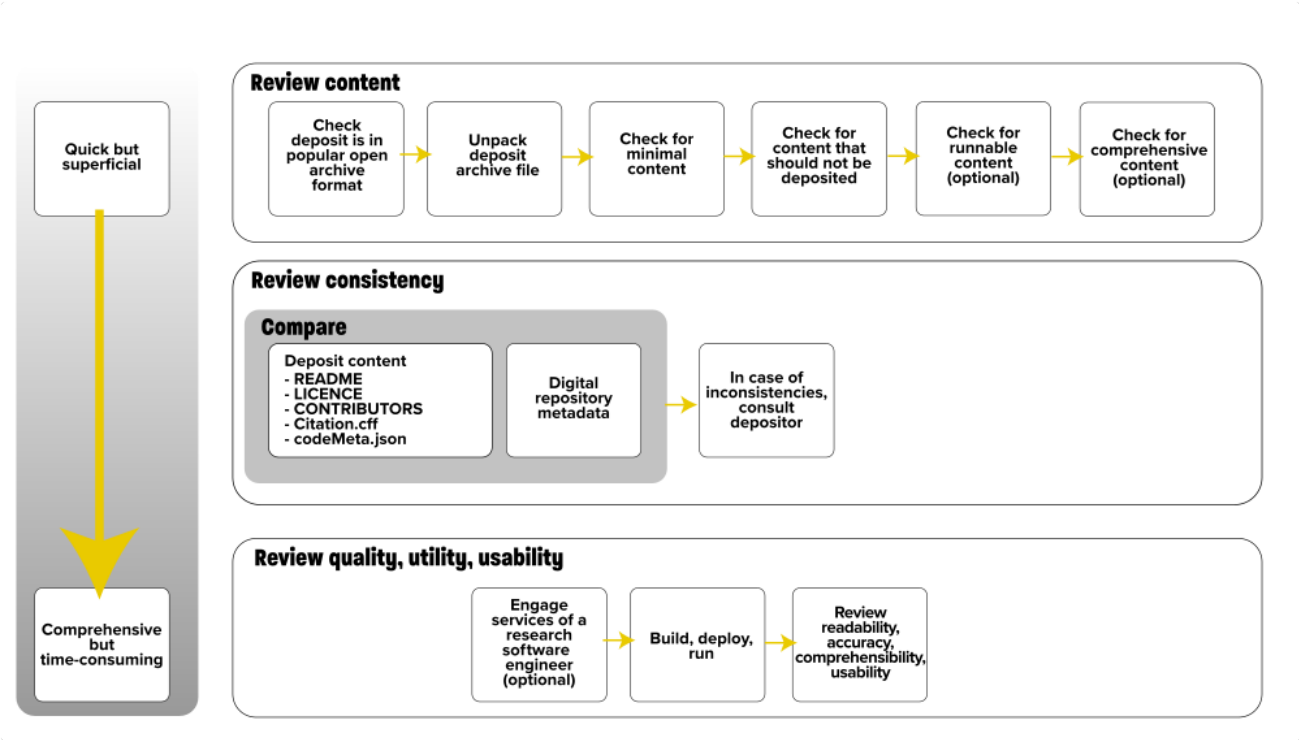
Version 1.0

doi:10.5281/zenodo.1327314

07 August 2018

Introduction

Digital repositories can differ in the deposits that they accept, the metadata requested from researchers, how researchers make their deposits and how these deposits are processed. This includes how deposits are reviewed post-deposit and pre-publication and the criteria that are used in these reviews. Despite the differences between individual digital repositories, there are some common checks that can be done for any software deposit. The nature and degree to which a software deposit can be reviewed depends upon the time, effort and expertise you have available. This guide describes three approaches to reviewing a deposit: a quick content review, a more detailed consistency review and a comprehensive review of quality, utility and usability.



How to review a software deposit

About this guide

This guide is one of a series of guides on software deposit, written by The Software Sustainability Institute [1], funded by Jisc [2]. For an overview of the series, see Michael Jackson (ed.) (07 August 2018). Software

Deposit: Guidance for Researchers (Version 1.0). Zenodo. doi:10.5281/zenodo.1327310. Online: https://softwaresaved.github.io/software-deposit-guidance/SoftwareDepositGuidance.html.

Review content

A content review has the advantage that it is quick to assess whether a deposit contains, or does not contain, content essential, or useful, for other researchers. However, the content review does not assess the quality, accuracy or consistency of these contents.

Check deposit archive

Check that the deposit is in a popular open archive format i.e. one of ".zip", ".tar", ".tar.gz", ".tgz", or ".tar.bz2". ".7z" or ".rar" should be avoided as support for these is not available by default in current versions of Linux, Windows or MacOS.

Unpack the deposit archive file to ensure that it has not been corrupted and can be unpacked successfully.

Check for minimal content

Check that the deposit includes the following minimal, essential, content:

- A simple README file which includes:
  - Name of the software.
  - Brief overview of the software, what it does (for example, what research problem it was written to solve) and what makes it novel (or different or distinct from similar software already available).
  - Contact: Researcher's name, affiliation, current email address and (optionally) ORCiD identifier.
  - Link to live software: for ongoing projects, links to their project web site and/or repository hosting service.
- Copyright and licence information. This could be in the README file, a LICENCE file (or a similarly named file e.g. LICENSE, LICENCE.txt, LICENSE.txt, LICENCE.md, LICENSE.md, Licence.txt etc.), a licences/ (or licenses/) sub-directory or in other documentation files.
- Authors and other contributors. This could be in the README file, a CONTRIBUTORS file (or a similarly named file e.g. CONTRIBUTORS.txt, CONTRIBUTORS.md, AUTHORS, CREDITS etc.) or in other documentation files.
- Source code.

Check that the deposit does *not* include the following files:

- Temporary directories and file. These can include: files starting or ending in "~"; XEmacs scratch files (e.g. "Dft.java~"); Microsoft Word, Excel and Powerpoint temporary files (e.g. "~wrcxxxx.tmp"); and files or directories called, or ending in "tmp".
- Source code repository configuration files. These can include ".git/" directories, ".hg/" directories, ".svn/" directories and "CVS/" directories.
- Binary files and executables. The only binary files and executables that should be present in a deposit are those that a researcher has explicitly and intentionally chose to include.

Check for runnable content (optional)

Check for the following content, if you want to check whether the deposit theoretically provides enough content to allow other researchers to build, install, configure and run the software on sample data:

- The README file also includes:
  - Version number of the software.
  - Copyright statement.
  - Name of their licence and third-party licences.
  - User documentation, or where in the deposit this can be found.

    - Summary of the other contents of the deposit with information on where in the deposit these can be found.
- User documentation and dependencies documentation. This could be in various formats e.g. text (".txt"), Markdown (".md"), reStructuredText (".rst"), HTML (".html"), Microsoft Word (".doc", ".docx"), OpenDocument Text Document (".odt"), Adobe Portable Document Format (".pdf"). It may be located in a subdirectory of the deposit.
- Sample input and output data.
- Recommended citation e.g., a CITATION.cff file.
- Metadata e.g., a codeMeta.json file.

Check for comprehensive content (optional)

Check for the following content, if you want to check whether the deposit theoretically provides enough content to allow other researchers to reuse, customise and modify the software and provides documentation allowing them to understand, in detail, both how the software implements the research and where it sits in its wider software ecosystem:

- Developer documentation. This could be in various formats e.g. text (".txt"), Markdown (".md"), reStructuredText (".rst"), HTML (".html"), Microsoft Word (".doc", ".docx"), OpenDocument Text Document (".odt"), Adobe Portable Document Format (".pdf"). It may be located in a subdirectory of the deposit.
- Narratives of what the code does and how it does it.
- Narratives of the software's ecosystem.
- Test code.
- Additional sample or test data.

Review consistency

Your digital repository will collect metadata about the researcher's deposit. The researcher's deposit itself will include metadata. This review checks that the metadata as recorded within your digital repository is consistent with that recorded within the researcher's deposit.

Compare your digital repository's metadata to metadata in the following files in the deposit:

- README file.
- LICENCE file (or a similarly named file e.g. LICENSE, LICENCE.txt, LICENSE.txt, LICENCE.md, LICENSE.md, Licence.txt etc.) and additional licence files in any licences/ (or licenses/) directory.
- CONTRIBUTORS file (or a similarly named file e.g. CONTRIBUTORS.txt, CONTRIBUTORS.md, AUTHORS, CREDITS etc.)
- CITATION.cff file (if present): This file lists information on how other researchers should cite use of the software deposit. It can contain metadata such as the software's name, description, authors, license and keywords.
- codeMeta.json file (if present): This file lists metadata about the software, however it may include a far richer set of metadata than captured by your digital repository.

If there are any discrepancies or inconsistencies then it would be expected that the metadata within the deposit takes precedence and the metadata within your digital repository should be updated accordingly. However, if in doubt, consult the researcher who made the deposit.

Review quality, utility and usability

A review of quality, utility and usability involves trying to build, deploy and run the software, following its user and developer documentation, reviewing these for readability, accuracy and usability. This review can help to determine whether another researcher could understand what has been deposited and whether they could replicate, reproduce and reuse the research, as manifested in the software, in the short term and inspect, for the historical record, in the long term, i.e., does the deposit now standalone as a useful resource independent of the researcher who deposited it. The disadvantage of such a review is that it requires a lot

more time to do. It may require you to consult the services of a research software engineer.

Find out more

Related Software deposit guides:

- Michael Jackson (ed.) (07 August 2018). Software Deposit: What to deposit (Version 1.0). Zenodo. doi:10.5281/zenodo.1327325. Online: https://softwaresaved.github.io/software-deposit-guidance/WhatToDeposit.html.
- Michael Jackson (ed.) (07 August 2018). Software Deposit: What not to deposit (Version 1.0). Zenodo. doi:10.5281/zenodo.1327323. Online: https://softwaresaved.github.io/software-deposit-guidance/WhatNotToDeposit.html.
- Michael Jackson (ed.) (07 August 2018). Software Deposit: How to describe a software deposit (Version 1.0). Zenodo. doi:10.5281/zenodo.1327321. Online: https://softwaresaved.github.io/software-deposit-guidance/HowToDescribeSoftwareDeposit.html.

Source code and data files:

- "List of filename extensions", Wikipedia, https://en.wikipedia.org/wiki/List_of_filename_extensions. A list of common filename extensions including those for source code and data files.

CodeMeta and codeMeta.json files:

- The CodeMeta Project, https://codemeta.github.io/. A machine-readable approach to documenting software metadata. Specific CodeMeta terms are listed at https://codemeta.github.io/terms/.

Software citation and CITATION.cff files:

- Citation File Format (CFF), https://citation-file-format.github.io/. A machine-readable approach to specifying the citation that others should use when citing your software.

Cite this guide

Please cite as: Michael Jackson (ed.) (07 August 2018). Software Deposit: How to review a software deposit (Version 1.0). Zenodo. doi:10.5281/zenodo.1327314. Online: https://softwaresaved.github.io/software-deposit-guidance/HowToReviewSoftwareDeposit.html.

---

1. The Software Sustainability Institute, https://www.software.ac.uk.↩
2. Jisc, https://www.jisc.ac.uk.↩