University of Reading

PUBLISHING RESEARCH SOFTWARE

A guide for researchers

As a researcher you may develop your own software to meet specific needs related to the generation, processing and analysis of research data and results. This software can be integral to the reliability and reproducibility of your research, and may need to be preserved and made available to others as well as your primary research outputs and data.

The University and many public funders, including the Research Councils, have policies requiring researchers to preserve and, where possible, share data collected or generated by them that support published research findings.[1] These requirements include any software created in the course of research that is necessary to replicate reported results.

Software can also be a valuable research output in its own right. By making your software available for others to use, you can help other researchers solve their own research problems, enable others to improve your software or adapt it for new purposes, and receive credit for the contribution you have made to your research community. Some software may even be viable for commercial exploitation.

But publication of software requires some effort, if the software is to be usable over the long term.

Research software is often written in response to specific research needs, and using whatever skills and resources may be to hand. Rarely is this software developed according to a rigorous engineering process, with formal specification, code reviews and testing. Code in its raw state may not be written, organised and documented in the most optimal way for long-term maintenance and use by others.

This guide highlights the things you need to consider when you are preparing to publish your software. It will help you to publish your software in a way that maximizes the benefits for both yourself and the wider communities of researchers and other users. It includes a selection of Useful links for further guidance and information, and a Software publishing checklist, to help you decide whether and how to publish your software.

What is software publishing and when should it be done?

Software publishing involves making available by online means the source code of software that has been developed, such that it can be re-used by others, subject to any restrictions specified in the terms of use under which it is published. The software published may be an integral software product, or, as is often the case, a segment of code that integrates with and builds on existing software or a script that automates the execution of a series of tasks in a given run-time environment.

[1] The University's Research Data Management Policy can be found at http://www.reading.ac.uk/reas-RDMpolicies.aspx. For information about funders' policies see http://www.reading.ac.uk/reas-FundersRDPolicies.aspx.

The following are typical research scenarios where software publishing would be expected to take place.[2]

Scripts written to perform analysis in widely-used scientific software

Many researchers collect experimental data and then use established scientific software, such as Matlab or R, to analyse and visualise the results. It may enable more effective validation of the reported results if you deposit in a data repository both the data themselves and the scripts that executed the analysis performed by the scientific software.

If the processing undertaken in the scientific analysis software is sufficiently well-described in the documentation accompanying the research data and/or in the paper reporting the research results, then further action may be unnecessary.

Scripts written to execute workflows

If you write a script to execute a series of operations that process data, for example by converting data from one format to another, it may not be necessary to make the script available, providing the data and metadata are accessible. But it may be of benefit to other researchers to execute the same workflow, so there is likely to be value in making the script available with the raw data.

Software to implement a new algorithm

Research may involve performing complex mathematical operations on existing data, as in climate modelling and analysis of large economic and financial datasets.

If you write a piece of software to implement an algorithm for performing calculations on an existing dataset, and then publish the algorithm and the results of its implementation, while this may be sufficient in principle for the results to be validated, in practice it may be of limited use to anyone wishing to replicate your results.

As algorithms and their implementation in computer code can be highly complex, if the results are to be effectively replicated, it will generally be necessary to make the software that implements the algorithm available. This will also enable others to vary or improve the implementation of the algorithm. Anyone wishing to validate the results would also need access to the existing dataset in the version used.

Developing existing commercial software

If you develop or improve commercial software in the course of conducting research, your freedom to publish the code you have developed is likely to be constrained by the commercial interests of the software owner.

Published findings may be able to describe the new version of the software and the results obtained from its application, but you may not be able to make the software itself openly available. The owner of the software may permit you to make the software available ad hoc, subject to a non-disclosure agreement, so that other researchers can verify the results reported in published findings. You should clarify what you will be permitted to do by the software owner at the outset of your research.

Protecting or exploiting IP in software

You may develop software which is necessary for the validation of your published research results, but which you believe has commercial potential that would be jeopardised by its disclosure. The University and many research funders permit a delay in publication of research results while arrangements are made to protect valuable IP. In this case you should seek the support of the University's IP Management

[2] These scenarios are based on those identified in guidance developed by the Software Sustainability Institute in consultation with EPSRC, See http://www.software.ac.uk/resources/guides/epsrc-research-data-policy-and-software.

team. They can help you ensure that if necessary the intellectual property in the software is protected before your results are published. You will still need to take steps to make the code available to anyone who needs to validate the research once the results are published, and this can be done under a non-disclosure agreement if required.

Software with ongoing research value

The University and public funders of research expect researchers to provide the means by which others can replicate and validate results reported in published research findings. Where software code is integral to validation of research results, researchers must make the code accessible to other researchers. If there is a valid reason for not making software openly available, for example, where there has been significant intellectual investment in developing the software, and where open publication would seriously compromise your future research, it may be possible to make the software available on an ad hoc basis only, subject to a non-disclosure agreement, so that other researchers can validate the results reported in published findings.

Ownership and rights

Software is intellectual property, and as such attracts legal rights of ownership. It is not necessarily the case that you have the right to do whatever you want with software you develop:

- you may not own the software, or have the automatic right to distribute it, even though it is your work;
- even where you do have rights in the software, these may not be exclusive. Co-developers and owners of software that you develop may have rights that need to be taken into account.

Before you can consider publishing your software, you must be sure that you either own the software, or have been given permission to distribute it by any parties with rights in the software.

What rights exist in any software you develop will also determine the terms on which you can make the software available, and what uses others can be permitted to make of it. This will affect how your software can be licensed.

The University's Intellectual Property Management team, based in Enterprise Services, can help you make the most of your software while respecting the rights of other interested parties. They can advise on your publishing options and the most suitable licence for your software. You should contact the IP team for specific advice, but the following general considerations are worth bearing in mind.

University staff

Intellectual property created in the course of employment is generally owned by the employer (unless any contract stipulates otherwise). This applies to all intellectual property, including software, created by University staff in the course of their employment.

University researchers are encouraged to disseminate their research outputs, and are allowed discretion in choosing how and by what means to publish their work, but they are expected to do this responsibly, in such a way that will not contravene any law, breach any contract, prejudice any ongoing or intended exploitation of the intellectual property, breach any confidence, or in any other way interfere with the rights of any other parties with interests in the work. When publishing software, you are expected to ensure that it is covered by an appropriate licence agreement.

University requirements for the management of intellectual property are set out in the Code of Practice on Intellectual Property, Commercial Exploitation and Financial Benefits.[3] You should always

[3] For software, see clauses 7.7-7.12. https://www.reading.ac.uk/closed/reas-PP_IntellectualPropertyOwnership.aspx. Accessible to University staff only.

seek advice on the appropriate means of publishing your software, to ensure that it is published responsibly.

Students

Students by default own the intellectual property in their own creations, but this ownership may be shared, if they are part of a research team or collaboration, or it may be assigned to another party or be subject to the terms of any contract governing the student's work, such as a sponsorship or employment agreement.

Tips

Bearing in mind these considerations, you should consider the following when publishing software:

- If the software was written collaboratively by staff and students, any students involved will need to formally assign their IP to the University or give written permission for the software to be published, if they are free to do so, or seek permission for publication from any funding parties with contractually-established rights in their research outputs;
- If the software includes contributions from people who are neither staff nor students of the University, they or their employer will need to give written permission for the software to be published;
- If any part of the software code has been copied from a copyright source, such as a book containing programming recipes, then the permission of the copyright owner to reproduce the code will need to be sought, or the code will need to be removed from the published software;
- Code downloaded from the Web with no stated licence terms is still someone's intellectual property. If you want to use it, you must seek permission from the provider. If you are unable to secure this permission, you cannot assume you have the right to publish this code.

Where permission to publish software needs to be sought, the University's IP Management team can advise on the correct way to request this permission.

Licensing

Software, like any intellectual property, should always be distributed under a licence. There are two very good reasons for this:

- A licence is an explicit statement of intellectual property rights, and makes it easier to protect these rights;
- It is provides a clear set of permissions and obligations under which the work may be used, so that any users of the work understand what they can and cannot do with it.

Licensing your work need not be difficult or complicated: in most cases the work can be straightforwardly licensed by the application of a standard licence statement. But it does require some forethought to get it right. Two principal considerations should be borne in mind when deciding how to licence your software:

- If you do not own the software or any other software with which it integrates, then your options are likely to be already pre-determined to a greater or lesser extent. You may be constrained in your licensing options by the requirements of any collaborating parties in the software or the terms of use of any existing software that you have used;
- Once you understand any operative constraints on your licensing options, you must then decide which among those options is most suitable for your software, given how you intend the software to be used.

Some general guidance on licensing follows, but you should seek advice on the most appropriate licence for your software. Contact the University's IP Management team if you require assistance.

Further guidance on licensing is also available from the Software Sustainability Institute.[4]

Licence options

There are a range of licensing options for software, as for other types of intellectual property, which may be more or less protective of the interests of rights-holders, and more or less restrictive in what they allow others to do with the software. Proprietary licences are always an option, and may be suitable for some purposes, typically to enable commercial exploitation. But much research software is licensed on an Open Source model, often using one of a number of standard licenses. Open Source licensing encourages collaborative development, re-use and distribution of software, and is not necessarily incompatible with commercial exploitation.

Amongst Open Source licences there is a distinction between *permissive* licences, which enable largely unrestricted re-use and re-licensing of the software, providing credit is given to the licensor, and *copyleft* licences, which require that any modifications of the software be shared under the same licence terms, thus effectively limiting the ability of others to commercially exploit the software.

When publishing Open Source software you are strongly advised to use a licence approved by the Open Source Initiative.[5]

Permissive licences

Apache 2.0[6] and MIT[7] are commonly-used permissive licences. Apache 2.0 is more explicit on certain points, such as that user contributions to the software are by default licensed under the same terms, so that it is not necessary to secure written permission from contributors to release software under the same licence. Apache 2.0 also includes a clause by means of which any contributor grants the licensor a patent licence.

These are probably the simplest license options for much research software, such as scripts and segments of code, where the principal interest is to make the software as openly available and re-usable as possible.

Copyleft licences

Copyleft licences are sometimes qualified as *strong copyleft* and *weak copyleft*. The distinction is largely operative in the case of composite software, where new code has been integrated with existing software code. Strong copyleft licences effectively require that the composite whole be licensed under the copyleft terms of the part, while weak copyleft applies only to the part and its derivatives, and not to the other parts with which the copyleft part was combined. GNU General Public License 3.0 (GPL)[8] is a widely-used strong copyleft licence. Commonly-used weak copyleft licences are the Mozilla Public License 2.0[9] and the GNU Lesser General Public License 3.0 (LGPL).[10]

Copyleft licences are more likely to be suitable where software may have commercial value, and you wish to limit the freedom of others to profit from the software, for example by patenting a derivative version.

Dual licensing

Dual licensing describes a model whereby software is released under both an Open Source licence (often GPL) and a proprietary licence. This is generally suitable for software products that have

[4] http://www.software.ac.uk/resources/guides/adopting-open-source-licence.
[5] https://opensource.org/licenses/category.
[6] http://choosealicense.com/licenses/apache-2.0/.
[7] http://choosealicense.com/licenses/mit/.
[8] http://choosealicense.com/licenses/gpl-3.0/.
[9] http://choosealicense.com/licenses/mpl-2.0/
[10] http://choosealicense.com/licenses/lgpl-3.0/.

commercial viability, in that the Open source licence encourages development and re-use of the software, while the commercial licence allows a version of the software and various add-on services to be sold to paying customers and incorporated in other commercial software applications. Examples of dual-licensed software include MySQL and FFTW.[11]

Dual licensing is most likely to be suitable for products with significant commercial potential, and may be rarely used for research software.

Incompatible licensing

Where code you write is integrated with existing software, you must beware of incompatible licensing risks. For example, if you integrate your code with existing software published under a copyleft licence, you would not be able to license your software under permissive or strong copyleft terms, as these would 'infect' the existing software in the version you have developed.

Publishing practicalities

It is important to attend to the practicalities of publishing software. As with any publishing activity, there are minimum editorial and production processes to be gone through if the work is to be published to an appropriate standard and distributed in a manner appropriate to its intended audience and use.

These processes can be considered in terms of *presentation and documentation*, so that the software is easy to read and understand; *packaging*, so that it is easy to use and execute; *preserving and sustaining the software*, so that it remains viable in the long term, and *publishing a software paper*, so that you can promote the software to the wider research community and receive academic credit for your work. Key considerations are outlined below.

Presentation and documentation

- Format your code clearly and consistently, so that it is straightforward to read and navigate.
- Include comments in your code, to explain what it is doing and why;
- Include at least one README file alongside your code files, with basic operational information, e.g. how to compile the software;
- Include a user guide for the software, explaining how it should be run and used. For simple software, this information can be included in a README file;
- Include references to any publications that explain the research problem the software was created to solve and the mathematical basis on which it was built.

For guidance on writing readable source code, consult the Software Sustainability Institute.[12]

Packaging

- Package your code in a directory that reflects its logical structure, and archive it as required in a Zip or tar.gz format that preserves the directory structure when unpacked;
- Ensure the code has a version number and maintain a version history;
- If the code needs to be compiled, can the program be built with a free compiler (e.g. GCC, G++,Gfortran)?[13]
- Does the code use a portable build system (e.g. Autotools, CMake)?[14]
- Does the package contain test data to demonstrate application of the program?

[11] http://www.mysql.com/; http://www.fftw.org/.
[12] http://software.ac.uk/resources/guides/writing-readable-source-code#node-131.
[13] https://gcc.gnu.org/, http://www.cprogramming.com/g++.html, https://gcc.gnu.org/wiki/GFortran.
[14] https://www.gnu.org/software/automake/manual/html_node/Autotools-Introduction.html; https://cmake.org/.

Preserving and sustaining software

You will need to consider whether your purpose in providing the software is solely to enable validation of your research results, or to allow the software to be actively re-used, maintained and developed. If your intention is to release software for active development, you must be prepared to support the software and your eventual user community. Code repositories can help you do this, but you should be prepared to offer some support in terms of handling user queries, resolving bugs, releasing updates, etc. The Software Sustainability Institute offers guidance on supporting Open Source software.[15]

If you simply intend to ensure the replicability of your research results, it may be sufficient to preserve the software code and documentation files alongside your data using a suitable repository, such as the University's Research Data Archive[16] or a general preservation and sharing service, such as Zenodo.[17]

If you want your software to be re-used, maintained and developed, you may prefer to use a code repository, such as GitHub or Bitbucket.[18] Code repositories enable the maintenance and development of software by multiple developers, and in addition to code hosting typically provide version control, bug tracking, release management, mailing lists and documentation functionality.

The two approaches can be combined, so that your software is made easily accessible for ongoing development via a code repository, while milestone versions of the code are archived for the long-term in a preservation service. For example, GitHub has an integration with Zenodo that enables users to archive a version of a code repository to Zenodo, where the files will be preserved and assigned a Digital Object Identifier (DOI), so that the given version can be easily cited and retrieved.[19]

The Software Sustainability Institute provides guidance on choosing a suitable repository for your software and lists a number of resources that can be considered.[20]

Publishing a software paper

Once you have published your software, you may want to use a peer-reviewed article to publish information about it, so that relevant user communities can learn about it and you can receive academic credit for your work. Where the software is a significant output of your research and has potential for re-use, it can be worth publishing a paper that is primarily focused on the software itself, rather than the research questions it was used to solve. There are a number of both general and subject-specific journals which accept submissions that are primarily about software: the Software Sustainability Institute maintains a list of these journals.[21]

[15] http://software.ac.uk/resources/guides/supporting-open-source-software.
[16] http://www.reading.ac.uk/reas-RDArchive.aspx.
[17] https://zenodo.org/.
[18] https://github.com/; https://bitbucket.org/.
[19] https://guides.github.com/activities/citable-code/.
[20] http://software.ac.uk/resources/guides/choosing-repository-your-software-project.
[21] http://www.software.ac.uk/resources/guides/which-journals-should-i-publish-my-software.

Useful links

choosealicense.com. http://choosealicense.com/

A user-friendly guide to choosing the right licence for your software.

Code of Practice on Intellectual Property, Commercial Exploitation and Financial Benefits. https://www.reading.ac.uk/closed/reas-PP_IntellectualPropertyOwnership.aspx

The University's Code of Practice on Intellectual Property. Accessible to University members only.

Guide to the Commercial Exploitation of Intellectual Property. https://www.reading.ac.uk/closed/reas-IPMPoliciesandProcedures.aspx

University guidance on commercial exploitation of intellectual property. Accessible to University members only.

Intellectual Property Management. http://www.reading.ac.uk/closed/reas-IPMCommercialisation.aspx

The University's Intellectual Property Management team can help you to manage and protect the intellectual property in your software, and can advise on the most suitable licensing option. Accessible to University members only.

Open Source Initiative: https://opensource.org/

The stewards of the Open Source Definition (OSD) and the community-recognized body for reviewing and approving licenses as OSD-conformant.

Research Data Management Policy. http://www.reading.ac.uk/internal/reas-RDMpolicies.aspx

The University's Research Data Management Policy.

Software Sustainability Institute. http://software.ac.uk/

A national facility for cultivating and improving research software. The website provides guides and resources to assist with management and publication of research software, and includes information about training and support offered by the SSI.

TLDR Legal. https://tldrlegal.com/

A guide to software licences in plain English, provided by legal experts.

University of Reading Research Data Archive. http://www.reading.ac.uk/reas-RDArchive.aspx

The Archive can be used to publish software code using a range of standard software licences or under bespoke licence terms.

Contacts

Intellectual Property Management team

http://www.reading.ac.uk/res-contact.aspx?#IntellectualPropertyManagement

Robert Darby, Research Data Manager

researchdata@reading.ac.uk / r.m.darby@reading.ac.uk / x6161

Software publishing checklist

1. Is the software necessary to validate my published research findings?

Yes: you should preserve and publish the software. You can delay publication or restrict access if there is a need to protect the IP pending commercial exploitation or to protect your research advantage.

No: you do not need to preserve or publish the software.

2. Does the software have commercial value or potential?

Yes: contact the IP Management team to discuss commercial exploitation the software.

No: the software may be worth publishing if it has value for non-commercial research uses.

3. Could the software be of use to myself or other researchers in the future?

Yes: you should consider preserving and publishing the software.

No: it is not likely to be worth preserving or publishing the software.

4. Do I own the software or have the right to distribute it?

Yes: you can publish the software, subject to any conditions specified in the owner's grant of permission to publish (e.g. the University Code of Practice on Intellectual Property).

No: you must obtain permission from the owner(s) of the software before you can publish it.

5. Does anyone else hold ownership or rights in the software?

Yes: you must obtain permission from any co-owner(s) or other rights-holders in the software before it can be published, and/or clarify the terms on which the software can be made available.

No: you can proceed to publish.

6. Have I decided how to licence my software?

Yes: check with the IP Management team that this licence can be used for your software and is appropriate.

No: contact the IP Management team to discuss your licensing options.

7. Is my software code ready to publish/distribute?

Yes: you can proceed to publish.

No: make sure your code is clearly and consistently formatted, organised and documented, and is packaged with everything necessary to enable users to read and execute the software.

8. Do I want my code to be used and developed by other researchers or software developers?

Yes: use a suitable software development code repository to publish your software, such as GitHub or Bitbucket, and consider publishing an article about your software in a dedicated journal.

No: if the software is required solely to validate research results, use a suitable repository for preserving your software, such as Zenodo or the University of Reading Research Data Archive.