Guidelines for the development and handling of research software at the AWI

1. Introduction

Software is an indispensable part of scientific work in almost all research disciplines, whether for simulations, for generating, processing, analyzing and visualizing research data or for controlling research devices and experiments. Many scientists develop their own software for their specific questions. Research software[1] is therefore not only a tool in the process of acquiring knowledge, but also represents an independent product of scientific work. Together with the research data, the associated software is the basis of good scientific practice[2].

The AWI would like to establish a sustainable approach to research software. For this purpose, the research software developed at the AWI is to be made accessible in the sense of Open Science[3] in order to enable the traceability and reproducibility of scientific results. At the same time, the reusability of research software is to be promoted. The AWI follows the "Recommendations for the Implementation of Guidelines and Policies on Research Software Management at the Helmholtz centers"[4] and takes into consideration impulses from the priority initiative "Digital Information" of the Alliance of Science Organizations in Germany[5].

The document is aimed at scientific project managers, software managers and developers at the AWI. However, the guideline also affects the area of responsibility of the technology transfer department and the legal department. The guideline is supplemented by a collection of materials for developers, in which some of the points formulated here in general are presented in more detail with regard to practice and are updated.

2. Development practice

The practice of software development and documentation should be coordinated and defined depending on the complexity of the software and the user group. For this purpose, application classes are defined:

- Application class 0: software with a small range of functions for personal use, that is not meant to be shared with others.
- Application class 1: Software that can be used and further developed to a limited extent by persons who were not engaged in the initial development
- Application class 2: Software for which long-term development and maintainability are to be ensured, but which has no product character.

[1] For definition of the term research software: "Today, software is used in all phases of scientific work, whether for the generation, processing, analysis, or visualization of research data. If this software is developed specifically for this purpose as part of research, it is referred to as research software." (translated from https://www.dfg.de/foerderung/info_wissenschaft/2019/info_wissenschaft_19_44/index.html)

[2] Deutsche Forschungsgemeinschaft DFG (2019): Guidelines for Safeguarding Good Research Practice. Code of Conduct.(translated) doi:10.5281/zenodo.3923601

[3] Commission's Directorate-General for Research and Innovation (2019), Factsheet Open Science. Online verfügbar unter https://ec.europa.eu/info/sites/info/files/research_and_innovation/knowledge_publications_tools_and_data/documents/ec_rtd_factsheet-open-science_2019.pdf.

[4] Recommendations for the Implementation of Guidelines and Policies on Research Software Management at the Helmholtz Centers (doi:10.48440/os.helmholtz.040) and Model Policy on Sustainable Software at the Helmholtz Centers, (doi:10.48440/os.helmholtz.041)

[5] Recommendations on the development, use and provision of Research Software (doi:10.5281/zenodo.1172988)

- Application class 3: Critical software and software with product character[6].

The AWI wants to create the following suggestions for action as practice for the various application classes, whereby development and documentation must be managed more strictly with expanding application classes.

- The program code should be managed in a version control system (e.g. Git). The associated repository should contain or reference all components of the research software that are necessary for its use.
- Tests (including the necessary reference and test data sets) should be prepared for the research software in order to be able to check the correctness of the software. The type and scope of the tests are adjusted depending on the software category.
- Stable versions (releases) of the software must be clearly identified for users, e.g. via version numbers or tags in the version control system.
- The research software must be documented. This includes the documentation of source code as well as the installation and user documentation.
- The program code should be implemented and structured in a comprehensible manner. The best practices applicable in the respective community must be observed.
- If the research software is developed by a group, the procedures for collaborative development should be specified and documented in writing, if feasible, so that new developers may rapidly learn and apply the coordinated methods. Documentation, test methodologies, development methods, collaborative tool use, development cycle definition, and communication routes are all covered.

To support developers, the AWI provides its own infrastructure and consulting services (such as version control systems, software publication options ...) or advises on the usage of relevant external infrastructures (e.g. Helmholtz-wide service offerings).

3. Software as a result and expression of scientific work

Development of research software is a creative process and research software may be thought of as executable knowledge. It is an intellectual and copyrighted achievement. AWI supports the activities in the HGF and beyond to document and acknowledge the work of software developers in scientific context for evaluation purposes. Software developers are encouraged to publish the software developed at the AWI in appropriate journals[7] making it citable. In exchange, used software must be adequately referenced (as much as practicable) in papers as part of good scientific practice.

For larger software projects that are made freely accessible, the number of website visits and software downloads, as well as other information, should be recorded and monitor systems should be installed (such as Matomo) in order to draw conclusions about the size and composition of the user group (important parameter for Helmholtz knowledge transfer).

3.1. Quality assurance

Since the quality of the research software has a direct impact on the quality of the scientific work in many cases, quality assurance is important in software development. The implementation and design of good software development and documentation practice

[6] Software with product character means software that is used in a productive context (e.g. on expeditions) and whose functionality is crucial.

[7] E.g.. Journal of Open Research Software, The R Journal, Computer & Geosciences, SoftwareX …

requires appropriate human and time resources, which must be considered by those responsible in the planning.

3.2. Training and career paths

The quality of the research software depends heavily on the existing knowledge and skills of the developers. Therefore, the AWI supports developers in their efforts for further training and networking by providing corresponding offers in the institute or across the centers in the Helmholtz Association (e.g. software carpentries) on the one hand and by enabling employees to participate in such regular offers on programming techniques, methods of software engineering and other aspects of software development on the other hand.

To be able to acquire and retain the essential expertise for software development in working groups over time, corresponding career pathways should be established, and employees' successes in software development should be acknowledged.

3.3. Reproducibility of results, making software accessible

Software must be made available in order to guarantee that scientific results are understandable, verifiable, and reproducible in line with the standards of good scientific practice. The AWI assists research software developers by providing corresponding in-house offers (publishing repository EPIC.awi.de) and advising on the usage of external services.

The European Commission-funded zenodo.org service provides the option of permanently archiving scientific software with a specified version (snapshot, version) as a ZIP archive. A citable DOI is assigned to software published on zenodo.org. Appropriate APIs may be used on zenodo to automatically publish releases from github or gitlab. For the software to be assigned to the AWI, the affiliation to AWI should be stated on Zenodo. The authors must import the publication's metadata into the AWI Research Information System elements.awi.de in order to enable the recording of citable software with a persistent identifier for the software publication indicator, which will be used beginning with the reporting year 2022 as part of the HGF's program-oriented research. Furthermore, the software published in this manner may be added to the personal CV.

4. Reuse of Research Software

High-quality software development necessitates the utilization of correspondingly large resources (time, staff). As a result, sustainable software development should allow for the reuse of research software and the ability for others to expand on current solutions.

Copyright law protects software as the outcome of creative activity. Third-party usage of software that is handed on or made accessible (i.e. application classes 1-3) must consequently be governed by a similar license. The selection of an appropriate license depends on a variety of criteria and should be considered as early as feasible in the development process to prevent limitations or conflicts created by the integration of third-party software.

As a result, the AWI supports the granting of third-party usage rights to software, with an open license advocated in the sense of Open Science, but commercial exploitation possibilities should not be ruled out. Developers can and should seek suitable guidance from the Technology Transfer Office early on regarding the alternatives, opportunities, risks, assistance possibilities, and issues to be addressed.

With open licenses, it is recommended to use existing (and proven) license models as far as possible, which are roughly divided into two categories:

- Copyleft:

  The licenses allow the code to be used, updated, and redistributed as long as the licensing terms are included in their whole. The most prominent representation is the GPL ("GNU General Public License")[8], which demands strong copyleft: if one uses GPL-licensed code in a program, the application must likewise be licensed under the GPL. It is not authorized to simply pass on a compiled program without access to the source code. A license of this type assures that the program stays open source and that all developers have access to future modifications and changes to the code.

  The "GNU Lesser General Public License" (LGPL)[9], which is mostly used for libraries, is less restrictive. She has a limited copyleft, allowing for the linking of libraries with both open source and proprietary software.

- Non-Copyleft:

  This collection of licenses, often known as permissive licenses, enable for additional improvements of the program to be issued under other licensing terms than the software itself, as long as the original copyright and license are provided. This means that the licensee is free to redistribute modified copies of the program under any licensing terms, including transferring them to proprietary software. The MIT license[10] and the Apache license[11] are the most important representatives of this group, with the MIT license giving more flexibility. The license's primary goal is to safeguard the software's creator in the case of faults in the software.

In the course of further digitization in science, research software is of great relevance. The present guidelines are designed to assure long-term quality of software development, particularly for increased cross-center collaboration, as well as to support all persons involved in the development process in order to continue to enable good research.

[8] https://opensource.org/licenses/GPL-3.0

[9] https://opensource.org/licenses/LGPL-3.0

[10] https://opensource.org/licenses/MIT

[11] https://opensource.org/licenses/Apache-2.0